Хмельницький національний університет
Факультет програмування
та комп'ютерних і телекомунікаційних систем
Кафедра інженерії програмного забезпечення

ДИПЛОМНА РОБОТА

Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій

Назва теми

Рівень вищої освіти Другий (магістерський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДРІПЗ.150170.01.05.ПЗ

Виконав студент 2 курсу група ІПЗм-19-1 _____ Є. Е. Прейзнер
Підпис Ініціали, прізвище

Керівник канд. техн. наук, доцент _____ О. М. Яшина
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. техн. наук, доцент _____ Г. І. Радельчук
Підпис Ініціали, прізвище

До захисту допускаю:
Завідувач кафедри інженерії програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

7 грудня 2020 р.

Хмельницький 2020

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Програмування та комп'ютерних і телекомунікаційних систем
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Другий (магістерський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
Л. П. Бедратюк _____
02 09 2020 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Прейзнеру Євгенію Едуардовичу
Прізвище, ім'я, по батькові студента

1. Тема проєкту (роботи) Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій

Керівник проєкту (роботи) Яшина Оксана Миколаївна, канд. техн. наук, доцент
Прізвище, ім'я, по батькові, наукова ступінь, вчене звання

Затверджена наказом ректора університету від 01.09.2020 р. № 118
2. Строк подання студентом проєкту (роботи) на кафедру 01.12.2020 р.
3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1 Дослідження предметної області
2 Моделі та методи для вирішення проблеми
3 Проектування програмного забезпечення для вирішення проблеми
4 Реалізація програмного забезпечення для вирішення проблеми
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди)

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «01» вересня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1. Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження; визначення структури дипломної роботи	01.09 - 07.09.2020	
2 Визначення структурних і функціональних особливостей предметної області, аналіз відомих рішень поставленої проблеми, постановка задачі.	08.09 - 25.09.2020	
3 Розробка методів та моделей для вирішення поставленої задачі	26.09 - 10.10.2020	
4. Робота над науковими статтями	11.10 - 20.10.2020	
5 Проектування архітектури програмної системи, опис вимог до її функціонування, проектування бази даних, інтерфейсу та вибір технологій для реалізації	11.10 - 26.10.2020	
6 Детальний опис програмної реалізації системи та опис результатів тестування	27.10 - 15.11.2020	
7 Узгодження постановки задачі, отриманих результатів та висновків; написання вступу, загальних висновків, оформлення джерел посилання та додатків; оформлення пояснювальної записки згідно вимог стандартів	16.11 - 30.11.2020	
8 Попередній захист дипломної роботи	Листопад (згідно графіка)	
9 Перевірка роботи на наявність плагіату; норм контроль; брошурування пояснювальної записки; підготовка супровідних документів	01.12 - 04.12.2020	
10 Підготовка до захисту дипломної роботи	05.12 - 08.12.2020	

Студент _____ Є. Е. Прейзнер
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ О. М. Яшина
Підпис Ініціали, прізвище

РЕФЕРАТ

Тема роботи: Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій.

Автор роботи: Прейзнер Євгеній Едуардович

Керівник роботи: Яшина Оксана Миколаївна

Обсяг – 82 с., 13 рис., 2 табл., 7 додатків, 19 джерел.

СМАРТ-ТЕХНОЛОГІЯ, ШТУЧНИЙ ІНТЕЛЕКТ, ЕКСПЕРТНА СИСТЕМА, РАЦІОН ХАРЧУВАННЯ, ОПТИМІЗАЦІЯ, БАЗА ЗНАНЬ, JAVA, DROOLS.

Об'єктом дослідження є інтелектуальні інформаційні системи та засоби вибору оптимального раціону харчування.

Предметом дослідження є моделі вдосконалення вибору оптимального раціону харчування із врахуванням індивідуальних особливостей.

Метою роботи є удосконалення процесу вибору оптимального раціону харчування із використанням смарт-технологій.

Результати дослідження показують, що впровадження розроблених алгоритмічних моделей та програмного засобу дає можливість удосконалити процес формування оптимальних раціонів харчування із врахуванням індивідуальних особливостей та покращити якість результатів цього процесу.

Практичне значення отриманих результатів полягає в тому, що запропоновані моделі та реалізація повинні спростити отримання якісних безкоштовних індивідуальних планів правильного харчування для широкого кола людей, що, у свою чергу, має допомогти вирішити загальну проблему збалансованого харчування.

4.12.2020
(дата)

(підпис)

ABSTRACT

Topic: Technology of Developing a Software System for Personal Diet Guidance Based on Smart Technology.

Author of work: Preizner Evgeniy Eduardovich

Head of work: Yashyna Oksana Mykolayivna

Size of work – 82 p., 13 pic., 2 tab., 7 appx., 19 sources.

SMART TECHNOLOGY, ARTIFICIAL INTELLIGENCE, EXPERT SYSTEM, NUTRITION, OPTIMIZATION, KNOWLEDGE BASE, JAVA, DROOLS.

The object of this study are intelligent information systems and tools for choosing the optimal diet.

The subject of the research is the models of improving the choice of the optimal diet taking into account individual features.

The purpose of this work is to improve the process of choosing the optimal diet using smart technologies.

The results of the study show that the implementation of the developed algorithmic models and software makes it possible to improve the process of forming optimal diets taking into account individual characteristics and improve the quality of the results of this process.

The practical significance of the obtained results is that the proposed models and implementation should facilitate the receipt of high-quality free individual nutrition plans for a wide range of people, which in turn should help solve the overall problem of balanced nutrition.

4.12.2020
(дата)

(підпис)

ЗМІСТ

ВСТУП

Неправильне харчування у сучасному світі є досить глобальною та серйозною проблемою. Велика кількість людей не дбає про, власне, харчування, тому що вони не мають достатньої кількості знань, щоб створити свій власний раціон правильного харчування, а послуги фахівця в цій області не всі можуть собі дозволити. Крім того, багатьом людям дуже важко самостійно слідкувати за кількістю спожитих калорій і обраховувати їх для кожної приготованої страви. Однак, описані проблеми можна вирішити за допомогою розробок в сфері інформаційних технологій.

Так, сьогодні ми спостерігаємо постійне збільшення обсягів інформації, яка використовується у різних сферах діяльності людини, зокрема економіці, науці, виробництві. Причому тенденція на подальше збільшення обсягів даних, що обробляються, зберігаються, накопичуються та використовуються, постійно підтверджується науково-практичними дослідженнями.

Звичайно для обробки даних та задоволення інформаційних потреб користувачів призначені інформаційні системи. Але сучасний стан технологічного та інформаційного розвитку вимагає від таких програмних комплексів використовувати елементи штучного інтелекту, моделі подання знань, бази знань тощо. Одним із таких прикладів поєднання штучного інтелекту та інформаційних програмних продуктів є експертна система на основі правил, яка дозволяє забезпечити обробку корисних даних та спілкування з користувачем.

Аналіз останніх досліджень. Питання представлення знань, розробки інтелектуальних систем різного призначення висвітлювалися як вітчизняними, так і зарубіжними науковцями. В той же час потребують подальшого дослідження питання інформаційної підтримки напрямів охорони здоров’я, зокрема оптимізації раціону харчування, оскільки детальний аналіз відомих аналогів показав, що для вирішення проблем в цій сфері засоби штучного інтелекту до сих пір не використовуються.

Виходячи з вищесказаного, тема дипломної роботи є актуальною.

Формулювання задачі наукового дослідження. Задачею наукового дослідження є створення інформаційної технології вибору оптимального раціону харчування із використанням смарт-технологій.

Мета роботи полягає в удосконаленні процесу вибору оптимального раціону харчування із використанням смарт-технологій.

Об’єктом дослідження є інтелектуальні інформаційні системи та засоби вибору оптимального раціону харчування.

Предметом дослідження є моделі та методи вдосконалення вибору оптимального раціону харчування із врахуванням індивідуальних особливостей.

Виходячи з мети, об’єкту та предмету дослідження, можна виділити наступні завдання:

– аналіз та опрацювання предметної області;

– дослідження методів, інструментів та пакетів прикладних програм прийняття рішень при виборі оптимального раціону харчування;

– вдосконалення процесу вибору оптимального раціону харчування за допомогою засобів штучного інтелекту;

– вдосконалення методу вибору оптимального раціону харчування із врахуванням індивідуальних особливостей;

– розробка програмної системи з реалізацією запропонованого методу.

За темою і результатами дипломної роботи опубліковані тези доповіді на науково-практичній Інтернет-конференції та стаття у фаховому Міжнародному науковому журналі.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика структурних і функціональних особливостей предметної області

Харчування є важливою фізіологічною потребою організму, від якої значною мірою залежить стан здоров’я людини. Воно необхідне для побудови та оновлення клітин, тканин, поповнення енергозатрат, синтезу гормонів, ферментів та інших регуляторів обмінних процесів.

Нераціональне (або неправильне) харчування може призвести до порушення обміну та розладу функціонального стану систем організму, особливо травної, серцево-судинної та центральної нервової систем. Водночас нераціональне харчування може спричинити захворювання, які пов’язані з частковим недоїданням або з повним голодом, що, у свою чергу, може викликати білково-калорійну недостатність або надлишок деяких компонентів їжі.

Негативні наслідки нераціонального харчування найбільше проявляються у дітей та літніх людей, а також у людей при малій рухливості та недостатньому м’язовому навантаженні. А оскільки сьогодні, завдяки широкому розповсюдженню інформаційних технологій, все більше людей наражають себе на негативні наслідки неправильного харчування, проводячи майже весь свій вільний та робочий час в малорухливому положенні, проблема нераціонального харчування набуває особливої гостроти, що доводить необхідність розробки програмних інструментів для подолання подібних проблем.

У свою чергу, раціональне, тобто побудоване на науковій основі, харчування сприяє збереженню здоров’я, високій фізичній та розумовій діяльності, активному довголіттю, до того ж забезпечує нормальний ріст, фізичний і психофізіологічний розвиток організму, його високу працездатність та стійкість до несприятливих чинників навколишнього середовища. Основою такого харчування є збалансованість, тобто оптимальне співвідношення компонентів їжі. За такого харчування до організму надходять різноманітні поживні речовини в кількостях, необхідних для нормальної життєдіяльності людини.

Складання оптимального раціону харчування для звичайної людини може стати непростою задачею, виконання якої може зайняти досить багато часу і сил (якщо людина намагається зробити це без допомоги фахівців), або грошей (якщо людина звертається за допомогою до експерта в цій області).

Розглянемо обидва випадки. Для самостійної корекції раціону харчування людина може використати величезну кількість програм здорового харчування, які сьогодні можна знайти в Інтернеті, в книгах, в журналах, прочитавши які, вона може вибрати для себе потрібну програму. Але це може бути шкідливо для організму, оскільки самостійні спроби корекції раціону без наявності необхідних знань можуть стати причиною вітамінної або мінеральної надмірності.

З іншого боку, існує більш безпечна, надійна і одночасно з цим більш затратна альтернатива цьому способу − це похід до лікаря-дієтолога, який вивчить організм людини і запропонує індивідуальну програму харчування. Кожна програма орієнтована на досягнення цілей, які необхідні клієнту. Наприклад, хтось хоче схуднути, хтось хоче бути в тонусі, комусь потрібно правильно набрати вагу для збільшення м’язової маси, а комусь необхідна програма, яка заповнить нестачу необхідних елементів в організмі, або програма, що виключає елементи, які організм погано переносить. Інакше кажучи, лікар-дієтолог на основі інформації про організм клієнта запропонує програму, яка допоможе клієнту якнайшвидше отримати бажані результати. Однак, що перший спосіб, що другий вимагають від людини чималих часових і фінансових витрат, тому доцільно було б розглянути створення іншого способу отримання індивідуальної програми раціонального харчування на основі сучасних засобів.

Для цього потрібно розглянути тенденції сучасного стану розвитку прогресу. Сьогодні ми живемо у світі інформаційних технологій, у світі масової автоматизації, де всі паперові і рутинні роботи, самостійні прийняття рішень і всі розрахунки вручну переходять в комп’ютерний вигляд. Це значно спрощує користувачам виконання великої кількості різних задач у різних сферах діяльності людини, зокрема економіці, науці, виробництві тощо, але разом з тим це створює тенденцію постійного збільшення обсягів інформації, яка використовується, що, у

свою чергу, зумовлює збільшення попиту на створення нових більш сучасних програм для вирішення задач в цих галузях.

На даний момент найбільш сучасне програмне забезпечення у розглянутій предметній області засноване на використанні методів пошуку оптимального рішення; до них відносять метод Нелдера-Міда, метод Хука-Дживса, симплекс-метод тощо (при тому, що нині у світі існують більш новітні технології).

Так, у сучасному світі спостерігається активне використання інформаційних систем з елементами штучного інтелекту. Це можна пояснити колосальною різницею ефективності цих систем у порівнянні зі звичайними інформаційними системами при вирішенні певних задач. З кожним днем все більшої популярності набуває сучасна побутова, комп'ютерна та портативна електронна техніка, що містить елементи штучного інтелекту. Основними відмінностями штучного інтелекту від звичайних інформаційних систем є формування високоякісного рішення, яке не поступається рішенню фахівця-людини; можливість пояснення користувачу сформованих рішень на основі бази знань; здатність системи до автоматичного виявлення певних закономірностей у накопичених фактах та збереження їх у базі знань; можливість працювати з інформацією, яка характеризується неповнотою або неточністю.

На основі вищесказаного можна зробити висновок про те, що було б логічно об'єднати і покращити процес оптимізації програми харчування за допомогою інформаційних технологій. Інакше кажучи, для розв'язання цієї проблеми доцільно буде розробити інформаційну технологію вибору оптимального раціону харчування із використанням найсучасніших наукових досягнень у сфері інформаційних технологій.

Таким чином, робота програми буде ґрунтуватися на основі штучного інтелекту, навченого на професійних знаннях експертів у цій галузі. Водночас всі раціони харчування будуть створені з врахуванням індивідуальних особливостей організму людини.

База знань цієї системи за вибором програми харчування буде розроблятися на основі моделі аналізу виключно експертних рекомендацій в цій області, а це

зводить до мінімуму ризик того, що отриманий результат не зможе задовольнити користувача або завдасть йому шкоди.

Результатом роботи програми буде отримання користувачем індивідуальної раціону харчування, який найбільше підходить для його цілей, минаючи всі довготривалі пошуки схожого раціону в різних джерелах і особисте відвідування фахівця в цій галузі.

Тобто будь-якій людині досить буде занести необхідні критерії в програму замість того, щоб займатися пошуками найбільш відповідної програми в Інтернеті або йти на особисту консультацію до фахівця. Після занесення обов’язкових критеріїв програма, аналізуючи отримані дані та використовуючи модель аналізу експертних рекомендацій, зможе видати необхідний для користувача оптимальний результат.

У свою чергу, під оптимальним результатом мається на увазі таке рішення, яке приймається на основі того, що воно має найбільшу корисність, є найкращим за всі інші рішення та буде мінімізувати або максимізувати ті критерії, які повинні призвести до досягнення поставленої мети користувача.

Також доцільно зазначити, що пропонований підхід на основі штучного інтелекту відмінний від раніше відомих розробок в цій області, оскільки може дозволити не тільки підрахувати калорії, які вжив користувач, і розрахувати норми споживання цих продуктів, а ще й надає можливість отримати повністю готову програму харчування індивідуально для кожного. Отже, користувачеві вже не доведеться витрачати час на компонування з випадкового набору продуктів свого власного раціону, програма зробить це за нього.

Очікується, що розроблений продукт буде мати великий попит, адже не у всіх вистачає часу кардинально зайнятися покращенням свого харчування. Тому розроблювана система значно спростила б життя людей, які хочуть вести здоровий спосіб життя, залучаючи своєю доступністю і простотою використання, а також поповнила б їх ряди та поліпшила якість життя в цілому.

1.2 Аналіз відомих моделей, методів та засобів

Оскільки розроблювана технологія буде створюватись з використанням смарт-технологій, тому доцільно буде описати що вони представляють собою та проаналізувати деякі методи створення подібних технологій.

Взагалі поняття інтелект (Intelligence) походить від латинського терміну intellectus – розум, розумові здібності людини, відповідно під штучним інтелектом (Artificial Intelligence – AI) мається на увазі здатність автоматичних систем вибирати і приймати оптимальні рішення на основі раніше отриманого життєвого досвіду і аналізу зовнішніх впливів, тобто брати на себе функції людини. А під смарт-технологіями розуміються засоби створення штучного інтелекту.

На даний момент найважливішою сферою досліджень інформаційних технологій (ІТ) є штучний інтелект (ШІ), а розробки в цій області вважаються найбільш сучасними та перспективними технологіями. І це можна легко пояснити, адже вони представляють собою неймовірно потужні інструменти, які значно полегшують життя та роботу людей і можуть використовуватись в різних галузях суспільної діяльності.

Сьогодні існує велика кількість методів та підходів до розробки штучного інтелекту. Таким чином, вся методологія штучного інтелекту базується на двох школах мислення [7]:

– звичайний ШІ;

– обчислювальний ШІ.

Розглянемо кожен тип ШІ більш детально. Так, звичайний ШІ розуміється як ШІ, який представляє сукупність компонентів і методів, що базуються на логічній відповіді на проблеми. До цього типу ШІ належать такі методи машинного навчання, які включають формалізм та статистичний аналіз. Звичайний ШІ також іноді називають символічним або акуратним ШІ.

До основних методів, що пов’язаних зі звичайним ШІ, належать:

– експертні системи: програми, які, діючи за певними правилами, обробляють велику кількість інформації, і в результаті видають висновок або

рекомендацію на її основі;

– міркування за аналогією (Case-Based Reasoning);

– байєсові мережі довіри;

– поведінковий підхід.

Проаналізуємо кожен з наведених методів. Так, експертна система – це комп'ютерна програма, яка на основі певних правил обробляє велику кількість інформації в певній предметній області та виробляє знання, операції з якими дозволяють розв'язувати проблеми в цій галузі або виробляти рекомендації і обґрунтовувати їх [8]. Кожне правило таких систем містить лише дві частини: умову та дію.

Ця програма може повністю виконувати функції людини-експерта і тим самим замінити його, або ж використовуватись в якості його помічника для підвищення ефективності роботи.

Таким чином, експерті системи вирішують наступні задачі:

– прогнозу;

– планування;

– конструювання;

– контролю;

– інтерпретації;

– діагностики;

– підтримки прийняття рішень;

– управління;

– оптимізації;

– інструктажу.

Система моделює механізм мислення людини при вирішенні задач в певній проблемній області, замість того щоб моделювати безпосередньо проблемну області або її частини. Втім, програма не відтворює психологічну модель експерта, вона лише повторює експертні методики розв'язання проблем за

допомогою комп’ютерних засобів, тобто програма вчиться виконанню деякої частини задач так само (або навіть краще), як це робить експерт.

Система створює певні висновки і міркування, опираючись на базу знань, яку вона має в своєму розпорядженні. Знання завзичай представлені в системі на деякій спеціальній мові і зберігаються окремо від, власне, програмного коду, який відповідає за формування міркувань та висновків.

Отже, основними перевагами експертних систем як перед людиною-оператором, так і перед звичайними алгоритмічними базами даних є:

– інтегрованість: існують інструментальні засоби, які легко входять до складу інших інформаційних технологій і засобів;

– відкритість і портативність;

– відсутність поспішних висновків: експертні системи не мають упередженого ставлення і вони стійкі до різних перешкод;

– видача оптимального рішення;

– необмежені розміри бази знань: експертна система може зберігати набагато більше знань, ніж фахівець;

– постійне зберігання даних: на відміну від експерта, система нічого не може забути.

Далі розглянемо системи міркувань на основі аналогічних випадків (Case Based Reasoning – CBR). Методика формулювання рішення на основі міркування за аналогією дозволяє вирішити нове, невідоме завдання, використовуючи або адаптуючи рішення вже відомої задачі, тобто використовуючи вже накопичений досвід вирішення подібних завдань. Суть цього підходу спирається на той факт, що фахівцю, який приймає рішення, властиво на першому етапі пошуку рішення поточного (нового) завдання намагатися використовувати рішення, які приймалися раніше в подібних ситуаціях, і, при необхідності, адаптувати їх до проблеми, що виникла (поточної проблемної ситуації) [9].

Іншими словами, ідея систем CBR полягає в тому, щоб знайти у минулому близькі аналоги ситуації, яку потрібно вирішити і обрати таку ж відповідь, яка

була правильною для них, тобто зробити прогноз на майбутнє або вибрати правильне рішення на основі вже існуючих рішень.

CBR-системи сильно відрізняються від класичних експертних систем, які діють на основі логічних правил, адже вони зберігають успішні рішення низки реальних проблем, які називаються прецедентами, і при появі нової проблеми за певним алгоритмом знаходять найбільш схожі та підходящі прецеденти, після чого пропонують відповідну модифіковану комбінацію рішень.

Якщо нова проблема таким чином буде успішно вирішена, тоді це рішення заноситься в базу прецедентів з метою підвищення ефективності роботи системи під час вирішення подібних проблем у майбутньому.

До числа основних переваг такого підходу можна віднести:

– можливість використання досвіду, який був накопичений системою, без інтенсивного залучення фахівця в тій чи іншій предметній області;

– можливість запам’ятовування невдалих рішень нарівні з успішними, що дає змогу уникнути потенційних проблем у майбутньому;

– можливість скорочення часу пошуку рішення завдяки використанню існуючого рішення для такого завдання;

– можливість виключення повторного отримання помилкового рішення;

– відсутність необхідності використання всіх існуючих знань в предметній області та їх поглибленого вивчення, оскільки можна обмежитися урахуванням тільки істотних особливостей предметної області;

– невеликий об’єм розрахунків у порівнянні з традиційним підходом, оскільки розрахунки використовуються тільки на етапі адаптації уже вирішеного прецеденту.

Маючи стільки переваг, CBR-системи показують досить пристойні результати при вирішенні великої кількості різних завдань, тим не менш вони мають декілька суттєвих недоліків. Так, основним їх недоліком вважають той факт, що вони не створюють ніяких правил або моделей, які узагальнили б накопичений досвід, а у виборі рішення вони ґрунтуються на всьому масиві доступних даних, тому важко сказати, на основі яких конкретно чинників подібні системи виводять свої відповіді.

Ще один недолік полягає у самоправстві, яке допускають системи CBR при виборі міри «близькості» прецедентів. Від цього головним чином залежить розмір множини прецедентів, які потрібно зберігати в пам’яті для досягнення задовільної класифікації або прогнозу.

Крім того, до мінусів CBR-систем ще відносять:

– зниження продуктивності системи при великій кількості прецедентів;

– проблематичність визначення критеріїв для порівняння та індексації прецедентів;

– складність у створенні алгоритмів визначення подібних (аналогічних) прецедентів;

– неможливість отримання рішення для завдань, які не мають прецедентів або ступінь схожості яких менше заданого порогового значення [11].

Наступними проаналізуємо байєсові мережі довіри.

Байєсові мережі (БМ) представляють собою графічні моделі подій і процесів, які створюються на основі об’єднання деяких результатів теорії ймовірностей і теорії графів. Вони тісно пов’язані з діаграмами впливу, які можна використовувати для прийняття рішень. Незважаючи на свою назву, ці мережі не обов’язково мають на увазі тісний зв’язок з байєсовськими методами. Назва пов’язана, перш за все, з Байєсовим правилом ймовірного виведення [10].

Тотожним найменуванням методу байєсових мереж є поняття «байєсової класифікації». Спочатку цей метод застосовувався в експертних системах для формалізації знань, проте у сучасному світі байєсову класифікацію використовують в якості одного з методів глибинного аналізу даних.

Байєсові мережі також є чудовим інструментом для опису досить складних процесів і подій з невизначеностями. Вони особливо корисні при розробці та аналізі машинних алгоритмів навчання. Основною ідеєю побудови графічної моделі є розкладання складної системи на прості елементи. Для об’єднання окремих елементів в систему використовуються результати теорії ймовірностей, які забезпечують моделі спроможність в цілому, а також дають можливість, графічні моделі з базами даних.

Такий граф-теоретичний підхід до побудови графічної моделі надає людині можливість будувати моделі процесів з безліччю сильно взаємодіючих змінних, а також створювати структури даних для подальших розробки ефективних алгоритмів їх обробки і прийняття рішень [11].

Так, байєсові мережі належать до методів інтелектуального аналізу даних та можуть бути застосовані для пошуку закономірностей між ними.

Основними перевагами мереж Байєса є наступні:

– зв’язки в моделі встановлюються між усіма змінними, що дає можливість легко обробляти ті випадки, в яких відсутні значення тих чи інших параметрів;

– мережі Байєса досить легкі в розумінні, що дає можливість проведення аналізу за сценарієм «що буде, якщо..» при прогностичному моделюванні;

– байєсові мережі можуть поєднувати знання експертів та закономірності, виведені зі статистичних даних;

– практичне застосування байєсових мереж допомагає уникнути зайвого ускладнення моделі.

Проте байєсові мережі також мають деякі недоліки, а саме:

– перемножувати умовні ймовірності потрібно тільки в тому випадку, якщо всі вхідні змінні справді статистично незалежні;

– безпосередня обробка безперервних змінних неможлива: їх необхідно перетворити в інтервальну шкалу, щоб атрибути стали дискретними, однак подібні перетворення можуть призвести до втрати значущих закономірностей;

– у строго-баєсовому підході на результат класифікації не впливає комбіноване значення пар різних атрибутів; впливають лише індивідуальні значення вхідних змінних.

Використання моделей байєсових мереж допомагає:

– визначати фактори, які впливають на процес реалізації поставленої мети;

– виявити зв’язки між знайденими факторами;

– проаналізувати степінь впливу для досягнення бажаного сценарію розвитку.

Наступними розглянемо методи обчислювального ШІ. Так, на відміну від звичайного ШІ, під обчислювальним штучним інтелектом розуміється такий ШІ,

основна мета якого полягає у практичному підході до інтеграції машинного навчання, а також штучного інтелекту в різних практичних сферах у всьому світі. Тобто це найцінніше застосування інтелекту, яке може бути здійснено машинами на даний момент. Ця форма ШІ передбачає ітеративну розробку і навчання. Навчання ґрунтується на досвіді і тому засноване на емпіричних даних, часто пов'язаних з несимвольним ШІ та нечіткими системами.

Виділяють наступні методи обчислювального ШІ:

– нейронні мережі: цифрові аналоги біологічної нервової системи, які показують, зокрема, високі здібності до розпізнавання образів.

– нечіткі системи: методики для міркування в умовах невизначеності.

– еволюційні обчислення: моделі, які засновані на основі поняття природного відбору, яке забезпечує відсіювання найменш оптимальних рішень згідно заданого критерію.

Сьогодні найбільш поширеним підходом до створення штучного інтелекту є використання нейронних мереж, тому з них і почнемо. Взагалі нейронна мережа, а точніше штучна нейронна мережа (ШНМ), представляє собою спрощений цифровий аналог біологічної нейронної мережі. Іншими словами, це обчислювальна мережа, що складається з великої кількості паралельно працюючих штучних нейронів. Штучний нейрон – елементарна одиниця обробки інформації, що накопичує експериментальні знання і надає їх для подальшої обробки [12]. Мережа обробляє вхідну інформацію і в процесі зміни свого стану в часі формує сукупність вихідних сигналів. Крім того, оброблювана інформація має чисельний характер, що дозволяє використовувати нейронну мережу в якості моделі об'єкта з абсолютно невідомими характеристиками.

Нейронні мережі мають цілу низку переваг, зокрема вони:

– надають можливість вирішення задач при невідомих закономірності: на відміну від традиційний математичних методів та експертних систем нейронні мережі здатні, використовуючи здатність навчання на безлічі прикладів, вирішувати завдання, в яких невідомі закономірності розвитку ситуації і залежності між вхідними та вихідними даними;

– мають можливість роботи при великій кількості неінформативних вхідних даних: нейронна мережа може самостійно визначити степінь придатності даних для вирішення завдання і відкинути непридатні або малопридатні дані;

– вміють адаптуватися до змін навколишнього середовища;

– мають потенційно надвисоку швидкодію: це досягається завдяки використанню масового паралелізму при обробці інформації;

– мають високу відмово стійкість: якщо при несприятливих умовах був пошкоджений нейрон або його зв'язок, тоді ускладнюється процес витягання запам’ятованої інформації тільки з цього нейрона, проте оскільки вся інформація, яка зберігається в нейронній мережі розподіляється між нейронами, то можна зробити висновок, що тільки серйозні пошкодження структури нейронної мережі істотно вплинуть на її працездатність.

Дивлячись на переваги нейронних мереж, стає очевидним той факт, що сфера їх застосування є дуже широкою. Так, ШНМ використовуються для:

– класифікації та розпізнавання об’єктів;

– управління та прийняття рішень;

– кластеризації;

– прогнозування;

– асоціативна пам’ять та стиснення даних;

– аналізу даних;

– задач оптимізації.

При класифікації та розпізнаванні в ролі об’єктів можуть бути концептуально різні предмети, наприклад зображення, звуки, символи тексту тощо. Щоб навчити мережу виконувати задачу класифікації в якості вхідних даних використовують навчальну вибірку різних образів з інформацією про те, до яких класів ці образи відносяться. У процесі навчання всі приклади навчальної вибірки від початку до кінця обробляються мережею (кожен такий цикл називається епохою), з кожним разом підвищуючи точність класифікації. Це дасть змогу мережі після навчання розпізнавати раніше невідомі образи і відносити їх до певного класу. Зазвичай зразки зберігаються у вигляді векторів значень ознак,

за якими можна визначити клас. Водночас ознаки повинні бути такими, щоб їх сукупність дозволяла однозначно визначати клас, до якого належить приклад, проте бувають ситуації, коли ознак для класифікації образу недостатньо. У такому випадку один і той же зразок може бути віднесений до декількох класів.

Задачі управління та прийняття рішень у нейронних мережах дуже схожі до задачі класифікації, оскільки при класифікації мережа приймає рішення про належність образу до того чи іншого класу. А це означає що аналогічним чином мережа може класифікувати ситуації, характеристики яких були введені до нейронної мережі в якості вхідних даних.

Процес кластеризації ШНМ являє собою розбиття вхідних даних на певні класи без наявності заздалегідь відомої інформації про кількість цих класів та їх ознаки. Таким чином, мережа, яка була навчена для вирішення задачі кластеризації здатна визначати, до якого класу можна віднести вхідні дані. Крім того, подібна мережа може легко визначити що вхідні дані не належать ні до одного з виділених класів і належать до нових даних, які були відсутні в навчальній вибірці. А це означає, що подібна мережа вміє виявляти нові, невідомі раніше класи даних. Відповідність між класами, які існують в предметної області, та класами, які виділила мережа може бути встановлена тільки людиною.

Ще однією здатністю ШНМ є змога робити прогнози, яка випливає з її здібностей до узагальнення та виділення прихованих залежностей між вхідними та вихідними даними. Так, мережа, яку навчили робити прогнози, здатна спрогнозувати майбутнє значення деякої послідовності ґрунтуючись на кількох попередніх значеннях та (або) на факторах, які існують в даний момент. Однак прогнозування можливо тільки тоді, коли попередні зміни справді в деякій мірі характеризують майбутні. Наприклад, прогнозування цін акцій на основі інформації про зміни цін за минулий тиждень може виявитися успішним, в той час як вдало спрогнозувати результати майбутньої лотереї на основі даних за останні 50 років майже неможливо.

Крім описаних застосувань, нейронні мережі ще можуть відновлювати вихідний набір даних з частини інформації (ця здатність називається

асоціативною пам’яттю). Ця здібність дозволяє відновлювати пошкоджені вхідні дані та викладати дані великої розмірності більш компактно, при умові якщо ці дані тісно взаємопов’язані між собою. Це досягається за допомогою здатності мережі виявляти взаємозв’язки між різними параметрами.

З описаного вище можна зробити висновок, що нейронні мережі вирішують велику кількість задач, які досить помітно відрізняються одна від одної. Це досягається за допомогою використання різних моделей для окремих задач. Так, в сучасному світі нейронні мережі мають велику кількість архітектур, зокрема виділяють наступні:

– мережі прямого поширення: всі зв’язки в таких мережах поширюються строго в одному напрямку, починаючи від вхідного шару нейронів, через приховані шари до вихідного шару;

– рекурентні нейронні мережі: зв’язки з яких складаються подібні мережі характеризуються утворенням спрямованої послідовності, водночас сигнал з нейронів прихованого шару або з вихідних нейронів частково передається назад на входи нейронів вхідного шару;

– радіально базисні функції: ця мережа досить компактна, швидко навчається і характеризується наявністю прихованого шару з радіальних елементів та вихідного шару з лінійних елементів;

– самоорганізаційна карта Кохонена – такий клас мереж, як правило, успішно застосовується в задачах розпізнавання і має всього два шари, кожен з яких складений з радіальних елементів.

Далі проведемо аналіз нечітких систем. Вони ґрунтуються на теорії нечітких множин та принципах нечіткої логіки. Так, нечітка логіка представляє собою набір нестрогих правил, в яких для досягнення поставленої мети можуть використовуватися радикальні ідеї, інтуїтивні здогадки, а також досвід фахівців, накопичений у відповідній області. Нечіткій логіці властива відсутність суворих стандартів. Найчастіше вона застосовується в експертних системах, нейронних мережах і системах штучного інтелекту. Замість традиційних значень «Істина» та «Брехня» в нечіткій логіці використовується більш широкий діапазон значень,

серед яких «Істина», «Брехня», «Можливо», «Іноді», «Не пам’ятаю» («Як би так», «Чому б і ні», «Ще не вирішив», «Не скажу» тощо).

Нечітка логіка просто незамінна у тих випадках, коли на поставлене запитання немає чіткої відповіді (так чи ні; «0» або «1») або наперед невідомі всі можливі ситуації. Наприклад, в нечіткій логіці висловлювання виду «X є велике число» інтерпретується як таке, що має неточне значення і характеризується деякою нечіткою множиною.

У загальному випадку механізм логічного висновку в рамках нечіткої логіки складається з чотирьох етапів:

– введення нечіткості;

– нечіткий висновок;

– композиція;

– приведення до чіткості.

Алгоритми нечіткого виведення в основному розрізняються видом правил, які вони використовують, різновидом логічних операцій та методом приведення до чіткості. Також варто зазначити, що нечітка логіка вплинула на інші парадигми штучного інтелекту. Так, об’єднання її принципів з методами інших напрямків породило нові напрямки, серед яких виділяють:

– адаптивні нечіткі системи;

– нечіткі асоціативні правила;

– нечіткі когнітивні карти;

– нечіткі запити;

– нечіткі нейронні мережі;

– нечітку кластеризацію.

Альтернативні методи штучного інтелекту використовуються у різних комбінаціях для створення гібридних інтелектуальних систем.

Основними перевагами нечітких таким систем можна виділити:

– можливість нечіткої формалізації критеріїв оцінки і порівняння: оперування неоднозначними критеріями («переважно», «більшість» тощо);

– можливість оцінки вхідних даних та виведених результатів;

– можливість оперування нечіткими вхідними даними;

– можливість виконання порівняльного аналізу складних динамічних систем із заданим ступенем точності, а також можливість швидкого моделювання подібних систем.

Останнім серед методів обчислювального ШІ розглянемо еволюційні обчислення. Так, розділ еволюційних обчислень (або еволюційного моделювання) ґрунтується головним чином на використанні еволюційних алгоритмів і застосовується, як правило, для вирішення задач комбінаторної оптимізації, планування, складання розкладів, обчислення маршрутів, а також задач розташування і транспортування.

У свою чергу, еволюційні алгоритми представляють собою такі алгоритми, які засновані на концепції біологічної еволюції, а саме на процесах відбору, мутації та відтворення. Основним прикладом подібних алгоритмів можна назвати генетичний алгоритм.

Суть подібних алгоритмів полягає у наступному. Задача, яку потрібно вирішити, інтерпретується так, щоб її рішення можна було представити у вигляді масиву. Далі у створеному масиві випадковим чином створюється певна кількість початкових елементів, які згодом оцінюються за допомогою функції пристосованості на можливість виживання, в результаті чого кожному вектору присвоюється відповідне значення пристосованості.

Після цього створюється наступне покоління елементів на основі використання «генетичних операторів» схрещування і мутації до елементів минулого покоління, які були допущені до наступного етапу (за допомогою оператора селекції). Елементи нового покоління також оцінюються, потім до них застосовується селекція, оператор схрещування, оператор мутації тощо. Так виглядає процес моделювання еволюційного процесу. Він триває декілька життєвих циклів (такі цикли називають поколіннями), поки не буде виконано умову зупинки алгоритму.

Генетичні алгоритми в основному застосовуються для:

– складання розкладів;

– оптимізації функцій;

– налагодження і навчання нейронних мереж;

– завдання компонування;

– апроксимації функцій.

Основними перевагами генетичних алгоритмів є наступні:

– відсутність необхідності в специфічних знаннях про задачу, яку потрібно вирішити;

– прозорість реалізації та ідейна простота;

– легкість кодування вихідних та вхідних даних;

– варіативність при виборі виду параметрів систем, які досліджуються;

– можливість використання алгоритму до широкого кола задач без значних змін у його будові;

– можливість розпаралелювання [13].

Аналогічним чином проведемо аналіз існуючих засобів у предметній області. Так, мережа Інтернет може запропонувати багато сервісів, які пропонують розробити індивідуальну програму харчування (втім, вони мають низку недоліків).

По-перше, користуватися цими ресурсами менш зручно, ніж використовувати готове програмне забезпечення. Майже всі подібні сервіси пропонують реєстрацію, що змушує користувачів витрачати досить багато часу для отримання можливості ввести дані про свій організм, не кажучи вже про те, щоб дочекатися отримання розробленої програми.

По-друге, надійність подібних сервісів може бути сумнівною, тобто їх можуть використовувати шахраї та продавати одну програму харчування як індивідуальну для кожного користувача, що може мати негативні наслідки для здоров’я користувачів.

По-третє, сервіси, що пропонують розробити індивідуальну програму харчування, також поступаються іншим програмним засобам в тому, що для користувача індивідуальна програма харчування пропонується лише одна, без будь-яких альтернатив та можливості вибору. Крім того, в подібних сервісах

послуга розробки індивідуальної програми харчування може мати дуже високу вартість, що робить її недоступною для широкого кола людей.

Таким чином, через наявність великої кількості значних недоліків на зміну Інтернет-сервісам приходять мобільні додатки.

Одним з таких засобів є додаток для операційної системи iOS під назвою «Диеты – правильное питание» від розробника RARUS-SOFT, який пропонує функції створення особистого раціону харчування на основі інформації про загальні показники організму користувача. Він створений на базі платформи «1С:Предприятие», має зручний та приємний інтерфейс, а також локалізацію на шість різних мов, у перелік яких українська мова не входить. Дослідити цей продукт більш детально не вдалося, оскільки абсолютно весь функціонал додатку є платним, що підтверджує актуальність роботи та розробки аналогів.

Крім мобільних додатків, в цій сфері існують ще настільні, одним з яких є російськомовний безкоштовний додаток під назвою «Здоровое питание» версії 1.13. З її допомогою людина може провести оцінку поточного режиму харчування та відкорегувати його, додавши або виключивши певні продукти з раціону.

До складу програми входить перелік продуктів харчування і медикаментів, а також є можливість пошуку продуктів за назвою та вибір по групах.

До основних функцій програми відноситься:

– автоматичний розрахунок калорійності, основного обміну і складання діаграми споживаних калорій на основі введених даних;

– оптимізація раціону для запобігання набору зайвої маси тіла;

– оптимізація раціону при вагітності;

– складання раціону для лікувально-відновлюваного харчування;

– складання своєї програми тренувань в залежності від маси тіла і статі;

– експорт результатів роботи програми в Excel- та RTF-файли;

– можливість корегувати харчування для дітей дошкільного віку.

За допомогою цієї програми користувач може скласти правильне меню і підібрати програму тренувань для підтримки гарної форми, відновлення після хвороби, при вагітності тощо. Крім того, ця програма має зрозумілий інтерфейс і

вона проста у використанні. Таким чином, на рисунку 1.1 зображено головну сторінку програми «Здоровое питание», яка призначена для зібрання основної інформації про користувача.

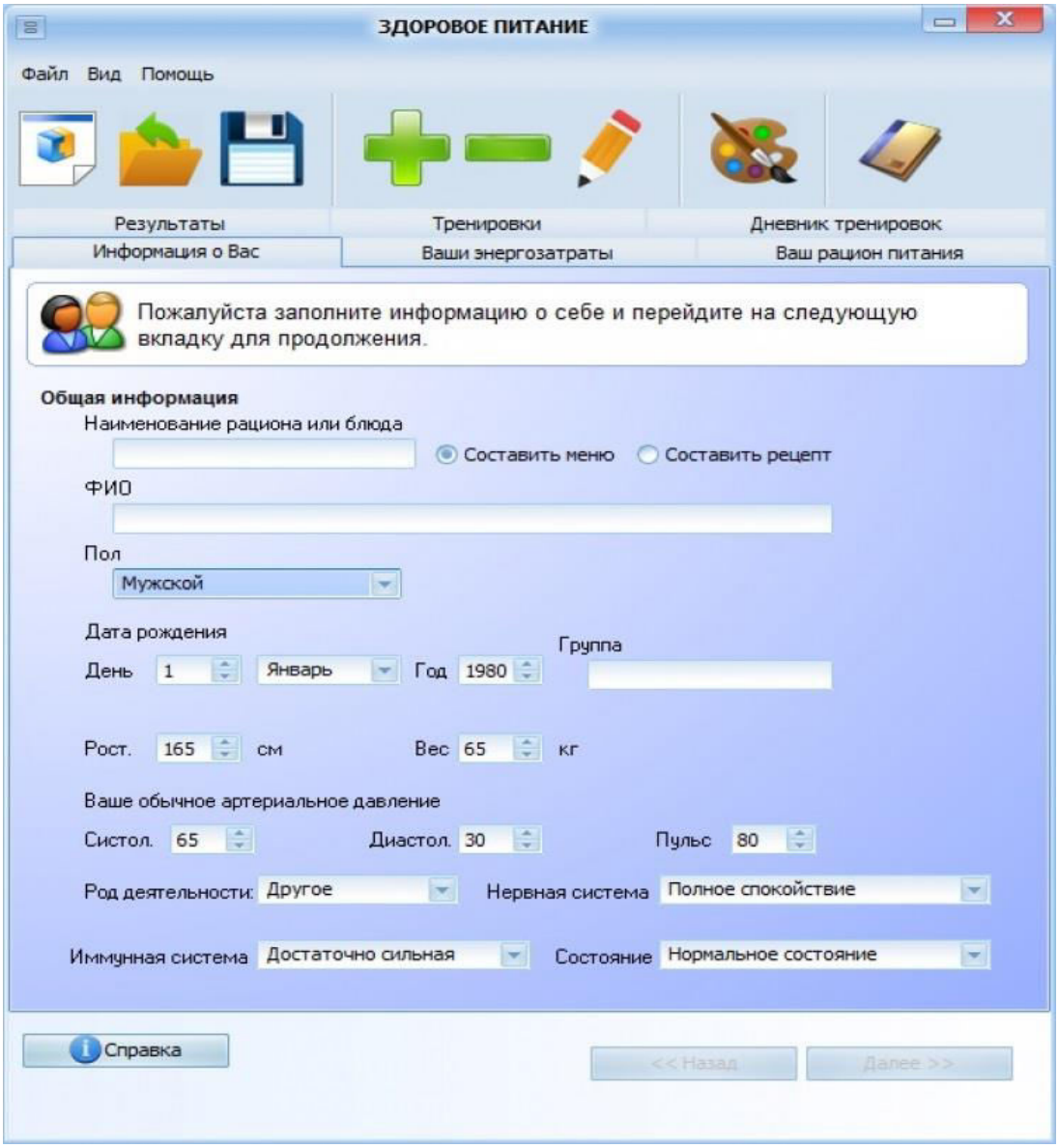


Рисунок 1.1 – Сторінка введення інформації про користувача

На рисунку 1.1 видно, що вікно містить поле для введення назви раціону, щоб його можна було потім зберегти, поле для повного імені користувача, статі, повної дати народження, групи, росту, ваги. Крім того, на цій сторінці розташовані поля для введення повної інформації про артеріальний тиск

користувача, а також засоби для вибору роду діяльності, стану нервової та імунної системи та загального стану користувача.

Основними перевагами цієї програми є:

– наявність великого каталогу продуктів харчування і медикаментів;

– детальний збір інформації про користувача;

– підтримка мови сценаріїв;

– докладний та обґрунтований список рекомендацій.

Однак програма також має деякі недоліки, а саме:

– неправильно підраховується ідеальне співвідношення росту та ваги;

– результат визначення біохімічного складу крові неправильний, так як введених вхідних даних недостатньо;

– тривалий розрахунок результатів;

– раціон харчування розраховується тільки на один день;

– немає підтримки української мови.

Отже, виходячи з вище написаного можна зробити висновок, що робота є актуальною, оскільки переважна більшість сучасних засобів в сфері харчування мають велику кількість недоліків, основними з яких є невисока якість результатів та відсутність української локалізації. У свою чергу, програмні засоби з більш-менш якісними результатами є платними.

Крім того, на даний момент жодна з програм для оптимізації раціону харчування не використовує найновіші програмні технології, а саме, штучний інтелект, що можна легко помітити якщо поглянути на невелику точність результатів у деяких додатках.

1.3 Постановка задачі

З огляду на все сказане вище, можна зробити висновок, що недооцінювати важливість правильного харчування не варто, проте велика кількість людей все ж

таки ставляться до цього питання зі зневагою. Причинами цього є сучасний темп життя, некомпетентність у питаннях культури харчування, а також брак часу.

Крім того, в сучасних умовах пандемії питання здорового харчування набуває особливої актуальності, оскільки велика кількість людей веде малорухливий спосіб життя, що негативно позначається на обміні корисних речовин і відповідно призводить до нераціонального харчування. Втім, існують різні шляхи вирішення цієї проблеми:

– використання Інтернет-сервісів або програмних засобів, більшість з яких можуть створити індивідуальні раціони тільки низької якості або надають лише платні послуги, а також не враховують всі важливі особливості організму користувача;

– використання послуг фахівця в цій області; цей шлях вирішення проблеми відрізняється високою якістю створених індивідуальних раціонів, а також високою ціною, яку не всі можуть собі дозволити.

Таким чином, для вирішення описаної проблеми доцільно вдосконалити процес вибору оптимального раціону харчування із використанням найновіших технологій, а саме, засобів штучного інтелекту.

Крім того, на основі опрацювання предметної області та дослідження методів, моделей та пакетів засобів прийняття рішень при виборі оптимального раціону харчування сформулюємо задачі, котрі в процесі роботи повинні бути вирішені, щоб досягти поставленої мети.

Так, до основних завдань роботи можна віднести:

– вдосконалення методу вибору оптимального раціону харчування із врахуванням індивідуальних особливостей користувача;

– вдосконалення процесу вибору оптимального раціону харчування за допомогою засобів штучного інтелекту;

– розробка програмної системи з реалізацією запропонованого методу.

1.4 Висновки

Підводячи підсумок, слід зазначити, що в межах розділу було здійснено докладний аналіз предметної області, а саме: проведено змістовний аналіз процесу харчування людини; пояснено, що являє собою і чим відрізняється раціональне та нераціональне харчування; зазначено негативні наслідки неправильного харчування та позитивні сторони правильного; обґрунтовано актуальність та новизну роботи за темою дослідження.

Крім того, детально розглянуто, проаналізовано та описано найбільш популярні, потужні та перспективні підходи до розробки штучного інтелекту, виділено концепції роботи та переваги кожного підходу, а також зазначено, у яких галузях вони зазвичай використовуються.

Після чого було проаналізовано наявне програмно-технічне забезпечення у предметній області, описано методи, на яких засновані ці засоби, а також описано їх основні функції, переваги та недоліки.

В результаті проведення дослідження визначено задачі, які потрібно виконати для досягнення мети роботи, а саме, удосконалення процесу вибору оптимального раціону харчування із використанням смарт-технологій.

2 МОДЕЛІ ТА МЕТОДИ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ

2.1 Вдосконалення методу вибору оптимального раціону харчування із врахуванням індивідуальних особливостей

На основі проведеного аналізу предметної області, а також дослідження методів, моделей та засобів, які використовуються при виборі оптимального раціону харчування, було вирішено розглянути та вдосконалити математичний метод вибору оптимального раціону харчування. Роботу цього методу пропонується покращити за рахунок часткової модифікації формул, а також використання експертної системи, яка повинна враховувати основні аспекти харчування та покращити результати оптимізації харчування.

Як вже згадувалось, під оптимальним харчуванням мається на увазі таке харчування, яке найкраще підходить для досягнення цілі користувача (схуднення, набір ваги, підтримка правильного харчування) із врахуванням стану його здоров’я.

Таким чином, в якості методу для вибору оптимального раціону харчування вирішено розглянути алгоритм, запропонований у 2010 році [14], математичну модель якого можна описати наступним чином:

$$\mu_k = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \tag{1}$$

де μ_k – змінна включення до складу раціону при наступних обмеженнях:

1) по не повторюваності страв в добовому раціоні:

$$\mu_k = 1\ ;\ k \in\ k_a, k_b, k_c \tag{2}$$

2) на граничну калорійність одного прийому їжі:

$$\sum_k \mu_k Q_k \leq Q_{\text{доб}} \eta_l \quad (3)$$

де Q_k – нормативна калорійність *k*-тої страви, ккал;

$Q_{\text{доб}}$ – гранична калорійність добового раціону, ккал;

η_l – частка добової калорійності, яка припадає на сніданок, обід і вечерю, таким чином, що:

$$\sum_l \eta_l = 1 \quad (4)$$

3) на граничний обсяг сніданку, обіду і вечері:

$$\sum_k \mu_k V_k \leq V_{\text{доб}} \omega_l \quad (5)$$

де $V_{\text{доб}}$ – граничний добовий обсяг раціону, г;

V_k – об'єм *k*-тої страви,г;

ω_l – частка добового обсягу, що припадає на сніданок, обід і вечерю, таким чином, що:

$$\sum_l \omega_l = 1 \quad (6)$$

4) по верхньому і нижньому кордонах вмісту *i*-го хімічного елемента:

$$B_i^{min} \leq \mu_k \sum_J \alpha_{ij} Z_{jk} \leq B_i^{max} \quad (7)$$

де B_i^{min} та B_i^{max} – відповідно нижні і верхні обмеження на вміст у добовому раціоні білків, жирів та вуглеводів, мг;

α_{ij} – вміст *i*-го елемента хімічного складу в одиниці *j*-го продукту;

Z_{jk} – рекомендований вміст *j*-го продукту в *k*-ій страві, г;

5) по масі *k*-ої страви, г:

$$\sum_J Z_{jk} = Z_k \quad (8)$$

де Z_{jk} – рекомендований вміст *j*-го продукту в *k*-ій страві, г;

Z_k – маса *k*-ої страви, г.

Вказаний алгоритм було обрано через те, що при роботі він враховує більшість основних аспектів харчування, зокрема, повторюваність споживання однієї страви на добу, калорійність кожного прийому їжі та його частку в добовому раціоні і, аналогічним чином, обсяг кожного прийому їжі та його частку в добовому раціоні. Крім того, він перевіряє загальну кількість білків, жирів та вуглеводів у всіх продуктах, з яких складається добовий раціон, і відповідність рекомендованого вмісту кожного продукту у страві до маси цієї страви.

Алгоритм працює наступним чином:

– випадково обирається перша страва, друга страва та напій для кожного прийому їжі;

– перевіряється, чи містяться обрані страви у добовому раціоні. Якщо повторювана страва є, тоді вона замінюється на іншу випадкову страву відповідного типу поки не буде підібрана страва, якої ще немає в раціоні.

– перевіряється, чи сума калорій всіх обраних страв не перевищує граничну калорійність добового раціону;

– перевіряється, чи сума обсягу всіх обраних страв не перевищує граничний добовий обсяг раціону;

– перевіряється, чи відповідає сумарна кількість кожного хімічного елемента обмеженням на вміст в добовому раціоні білків, жирів та вуглеводів

– якщо всі попередні перевірки пройшли успішно, формується добовий раціон з підібраних страв.

Так, під час роботи алгоритму перевіряється відповідність підібраних страв описаним обмеженням. Якщо в процесі перевірки виявиться, що страва (або комбінація страв) не відповідає хоча б одному з перерахованих обмежень, тоді ця страва замінюється на іншу.

На основі вищесказаного можна зробити висновок, що деякі обмеження описаного алгоритму, а саме обмеження 4 та 5, що описані формулами (7) та (8), є надлишковими для вирішення поставленої задачі. Достатньо буде лише перевіряти загальний вміст білків, жирів та вуглеводів у кожному прийомі їжі. Тому модифікуємо раніше описаний алгоритм, а саме приберемо формулу (8) та змінимо обмеження 4 таким чином:

4) по верхньому і нижньому кордонах вмісту *i*-го хімічного елемента у кожному прийомі їжі:

$$B_i^{min} \leq \mu_k \sum_k \alpha_{ik} \leq B_i^{max} \quad (9)$$

де B_i^{min} та B_i^{max} – відповідно нижні і верхні обмеження на вміст у добовому раціоні білків, жирів та вуглеводів, г;

α_{ik} – вміст *i*-го елемента хімічного складу в *k*-ій страві, г;

Варто зазначити, що навіть після модифікацій ресурсозатратність алгоритму напряму залежить від кількості обмежень, які враховуються. Відповідно, щоб

покращити цей показник, доцільно замінити деякі з описаних обмежень математичними обчисленнями у наступний спосіб:

– прибрати обмеження на граничний обсяг сніданку, обіду і вечері;

– замість обмеження використати розрахунок обсягу кожного прийому їжі на основі граничної кількості калорій цього прийому.

В результаті замість формул (5) і (6) будуть використовуватись наступні:

$$V_{nk} = \left(\frac{Q_{\text{доб}}\eta_l\beta_{mk}}{Q_k}\right)100 \tag{10}$$

де V_{nk} – обсяг *n*-ї страви в *k*-му прийомі їжі, г;

$Q_{\text{доб}}$ – гранична калорійність добового раціону, ккал;

η_l – частка добової калорійності, яка припадає на сніданок, обід і вечерю;

Q_k – калорійність *k*-тої страви, ккал.

β_{mk} – частка страви *m*-го типу у *k*-му прийомі їжі, така що:

$$\sum_m \beta_{mk} = 1 \tag{11}$$

Таким чином, у формулі (10) введено коефіцієнт β, який вказує частку першої, другої страви та напою у кожному прийомі їжі та дозволяє збільшити точність обрахунків обсягів страв.

Варто зазначити, що для використання описаної математичної моделі необхідно в якості вхідних даних мати не лише інформацію про вік, стать, зріст та вагу користувача, а ще й інформацію про його ціль (схуднення, набір ваги, підтримка здорового харчування) та обраховану на основі цих даних граничну добову кількість калорій, яку користувачу рекомендовано вживати для досягнення своєї цілі.

Для врахування індивідуальних особливостей при використанні наведеного вище алгоритму необхідно обрахувати граничну калорійність добового раціону для кожного окремого користувача. Щоб виконати цю задачу, пропонується використати формули Міффліна-Сан-Жеора [15], які видають кращі результати підрахунку добової кількості калорій, ніж інші формули [16] і, до того ж, враховують фізичну активність людини:

$$Q_{\text{доб}} = (10w + 6{,}25h - 5a - 161)F \quad (12)$$

$$Q_{\text{доб}} = (10w + 6{,}25h - 5a + 5)F \quad (13)$$

де $Q_{\text{доб}}$ – добова норма калорій, ккал;

w – вага, кг;

h – зріст, см;

a – вік, р;

F – коефіцієнт активності людини, який приймає значення 1,2 при мінімальній активності; 1,35, при низькі активності; 1,55, якщо активність людини середня; 1,75 при високій і відповідно 1,95 при екстремально високій активності.

За формулою (12) обраховується добова норма калорій жінки, а за формулою (13) – норма калорій чоловіка. Ці формули будуть використовуватись для визначення граничної калорійності добового раціону користувача перед використанням математичної моделі формування раціону харчування.

Отже, вдосконалений алгоритм при формуванні меню враховує доволі багато обмежень, які базуються на інформації про індивідуальні особливості організму користувача, але, на думку автора, навіть цього недостатньо, оскільки тут не враховується такий важливий аспект харчування, як наявність у користувача алергій та захворювань. Для усунення цього недоліку було вирішено вдосконалити описаний алгоритм вибору оптимального раціону харчування за рахунок використання засобів штучного інтелекту.

2.2 Вдосконалення процесу вибору оптимального раціону харчування за допомогою засобів штучного інтелекту

Перед тим, як переходити до безпосередньо вдосконалення процесу вибору оптимального раціону харчування, доцільно буле обрати методологію розробки штучного інтелекту для вирішення поставленої задачі.

Таким чином, поглянемо на головні особливості описаних раніше підходів до створення штучного інтелекту з точки зору задачі вибору оптимального раціону харчування.

Розпочнемо з методики міркувань за аналогією. Цей підхід не підходить для вирішення поставленої задачі, оскільки при його використанні програма повинна передбачати наявність надзвичайно великої кількості прецедентів, що, у свою чергу, значно збільшить складність та час розробки та призведе до значного зниження продуктивності.

Крім того, при використанні цього підходу є вірогідність виникнення ситуацій, які не матимуть прецедентів у базі системи, що, у свою чергу, не дозволить системі сформувати користувачу індивідуальний раціон. Така особливість підходу значно знижує його корисність і може погіршити процес вибору оптимального раціону харчування замість того, щоб покращити його.

Далі розглянемо байєсові мережі довіри. На відміну від міркувань за аналогією, байєсові мережі у сучасному світі використовуються тільки для задач глибинного аналізу даних. Це зумовлено низькою ефективністю мереж в інших сферах, зокрема для вирішення задач оптимізації.

Нейронні мережі представляють собою надзвичайно потужний інструмент, який може вирішити широке коло задач, у тому числі задачі класифікації, кластеризації, аналізу даних, оптимізації тощо. Наша проблема відноситься до задачі оптимізації, а це означає, що нейронні мережі можна використати як засіб вдосконалення процесу вибору оптимального раціону харчування.

Проте для того, щоб таким чином отримати пристойні результати, нейронну мережу необхідно спочатку навчити на величезній кількості специфічних

навчальних даних, що робить використання цієї методики неможливим у нашому випадку, оскільки автору не вдалося знайти у широкому доступі необхідної кількості даних для вирішення поставленої задачі. До того ж, нейронні мережі мають відносно високу обчислювальну вартість процесу навчання як за обсягом пам’яті, яку вони займають, так і за часом.

Серед підходів, які однозначно не підходять для вирішення задачі вибору оптимального раціону харчування, варто виділити нечіткі системи, адже неточностей при роботі в сфері охорони здоров’я не повинно бути, а це означає, що принципи роботи таких систем не відповідають специфіці предметної області.

Таким чином, проаналізувавши основні концепції створення інтелектуальних технологій, можна зробити висновок, що серед усіх розглянутих підходів для вирішення задачі оптимізації раціону харчування найкраще підходять експертні системи, оскільки вони:

– можуть зберігати знання експертів-дієтологів протягом довгого часу;

– надають можливість одержання й об’єднання експертних знань з багатьох джерел, що є вкрай важливим фактором при відсутності безпосередньо фахівця у необхідній сфері;

– можуть надавати об’єктивний результат за будь-яких обставин і при необхідності пояснити його.

Тому доцільно буде розібрати структуру безпосередньо експертних систем детальніше. Так, у додатку А на рисунку А.1 зображено структуру роботи експертної системи у загальному випадку, а також основні компоненти, з яких складаються подібні системи, та яким чином вони пов’язані між собою.

На рисунку А.1 можна побачити, що експертна система у загальному випадку складається з чотирьох основних компонентів: модуля висновків (МВ), бази знань, інтелектуального редактора бази знань та інтерфейсу користувача.

Розглянемо детально кожен із компонентів. Всі складові системи можна поділити на два типи:

– компоненти, які не використовуються під час роботи системи, а є необхідними лише в процесі створення;

– компоненти, які безпосередньо беруть участь у роботі системи.

До перших відноситься лише інтелектуальний редактор бази знань. Цей компонент надає можливість людині (зазвичай, такою людиною є фахівець предметної області або інженер із знань) створювати та доповнювати базу знань у режимі діалогу.

Всі інші складові належать до другого типу компонентів і відіграють важливу роль при роботі системи. Так, модуль висновків – це основний компонент роботи системи, який моделює механізм мислення експерта у предметній області на основі знань, які розміщуються у базі знань.

У свою чергу, під базою знань розуміється така база даних, яка зберігає не лише довгострокові дані, які описують предметну область, а також перелік правил, що описують дії, які необхідно проводити над даними цієї області.

Робоча пам’ять призначена для зберігання всіх вхідних даних експертної системи (фактів) в процесі роботи модуля висновків.

Наступним компонентом, який потрібен для роботи системи, називається підсистемою пояснень (або блоком пояснень). Він представляє собою програму, яка пояснює користувачу, яким чином були зроблені ті чи інші висновки при формуванні вирішення проблеми. Зокрема, цей блок може вивести повний ланцюг умовиводів, які були зроблені при формуванні рішення.

Інтерфейс користувача – це єдиний компонент, який взаємодіє безпосередньо з користувачем, тобто надає йому можливість вводити інформацію в експертну систему, отримувати результати її роботи, а також переглядати інформацію, яку надає підсистема пояснень.

Отже, на основі сказаного вище, можна зробити висновок, що серед найбільш популярних та потужних засобів штучного інтелекту для вибору оптимального раціону харчування найбільше підходять експертні системи.

Таким чином, пропонується дещо покращити процес вибору оптимального раціону харчування завдяки використанню експертної системи з метою інтелектуального підбору страв для кожного окремого користувача.

Крім того, на думку автора, доцільно дещо модифікувати описану раніше структуру експертної системи для конкретної задачі наступним чином:

– прибрати інтелектуальний редактор знань;

– прибрати підсистему пояснень;

– розбити базу знань на базу даних та базу правил.

У цьому методі пропонується використовувати інтелектуальний редактор знань в залежності від кваліфікації людини, яка буде створювати та редагувати базу знань. Таким чином, цей компонент доцільно реалізовувати та застосовувати тільки у випадках, коли базу знань формує або редагує людина, яка не володіє мовами правил та програмування. Це посприяє зменшенню часу на розробку програмного забезпечення, яке реалізує запропонований в роботі метод, у випадках, коли база знань формується самим розробником на основі інформації з різних джерел.

Крім того, видалення підсистеми пояснень допоможе збільшити швидкодію роботи системи та зменшити ресурсозатратність.

Базу знань пропонується поділити на два окремі компоненти, кожен з яких буде виконувати свої функції. Таким чином, база даних буде використовуватись в якості контейнера даних, інформація з якого буде подаватись в робочу пам’ять системи і оброблятись у вигляді фактів, а база правил буде зберігати правила обробки фактів.

Отже, для вдосконалення процесу вибору оптимального раціону харчування пропонується дещо покращити структуру експертної системи, використавши модифікований раніше алгоритм.

Так, у програмному продукті експертна система буде, використовуючи продукційні правила, реалізовувати раціональний вибір продуктів та страв на основі властивостей продуктів і раціону в цілому за критеріями харчової та біологічної цінності в залежності від стану здоров’я людини (зріст, вага тіла, стать, наявність алергій та хвороб), її щоденної активності та цілей, які вона хоче досягти. Відповідно, після вибору продуктів буде використано описаний раніше алгоритм формування раціону, щоб підрахувати розміри порцій добового меню,

враховуючи поживну цінність, кількість мінералів, обмеження не повторюваності цих страв, граничний обсяг кожного прийому їжі тощо.

Структура вдосконаленої експертної системи з використанням продукційної моделі представлена у додатку Б на рисунку Б.1.

На рисунку Б.1 можна помітити, що представлена модель має змінену структуру експертної системи і складається з шести компонентів:

– інтерфейс користувача;
– модуль висновків;
– підсистема формування раціону;
– база правил;
– робоча пам’ять;
– база даних.

Розглянемо детально кожен з цих компонентів.

Інтерфейс користувача, як і в загальній структурі експертної системи, головним чином призначений тільки для введення вхідних даних про користувача та відображення результатів роботи.

Аналогічним чином призначення робочої пам’яті не відрізняється від описаного раніше, тому розглянемо лише типи даних, які потрапляють до робочої пам’яті при роботі системи.

На рисунку Б.1 зображено, що всі необхідні для роботи системи дані потрапляють у робочу пам’ять з інтерфейсу користувача (інформація про стан здоров’я організму), а також з бази даних (інформація про страви).

База даних у програмній системі потрібна лише для збереження детальної інформації про страви, а саме: поживну цінність, вміст білків, жирів, вуглеводів, дані про алергени, які містяться у страві.

База правил, відповідно до своєї назви, зберігає продукційні правила, за допомогою яких будуть оброблятися факти. Як вже згадувалося раніше, правило представляє собою деяку пару умови та дії («якщо… то …»), на основі виконання якої можна робити певні висновки.

Цей компонент доцільно розглянути більш детально, оскільки саме в ньому буде реалізована покращена модель експертних рекомендацій, яка буде містити знання фахівців предметної області, на основі яких буде виконуватись інтелектуальний підбір страв для кожного окремого користувача.

У запропонованому методі за рахунок правил експертної системи буде вдосконалено підбір страв для конкретного користувача на основі інформації про особливості його організму. Досягатись це буде наступним чином: на основі існуючих у базі продукційних правил мають перевірятись всі страви з бази даних і видалятись тільки такі, які протипоказані конкретному користувачу за станом здоров'я. Така перевірка забезпечить безпечність розроблюваних раціонів харчування та не дозволить нашкодити людині.

Головним компонентом експертної системи є модуль висновків, оскільки саме в цій частині системи використовуються знання експертів з бази правил та формується результат підбору продуктів та страв для користувача.

Наостанок доцільно описати підсистему формування раціону. Як вже згадувалось раніше, ця частина програми реалізує модифікований у попередньому підрозділі алгоритм та призначена для підрахунку порцій і періодичності вживання страв, які найкраще підходять для досягнення цілі користувача на основі введених ним даних до своє здоров'я.

Реалізація всіх описаних модулів у вихідному програмному продукті повинна відбуватися послідовно від компонентів, які потрібні для зберігання інформації, до реалізації інтерфейсу користувача та модулів, що описують логіку роботи з вхідними даними.

Таким чином, підбір оптимального раціону харчування буде виконуватись з врахуванням всіх перерахованих обмежень в кожному конкретному випадку, що дозволить створювати оптимальні рішення, пов'язані з вибором і корекцією кількості калорій, складу раціонів та продуктів харчування, згідно з медико-біологічними вимогами користувача, які формуються на основі індивідуальних особливостей організму користувача та обмежень, які вони накладають.

Зокрема, запропонований метод, на відміну від існуючих розробок в предметній області, буде, на основі вдосконаленої продукційної моделі, підбирати для користувача продукти та страви, враховуючи наявність алергій та захворювань, при яких обмеження у харчуванні є обов’язковим.

2.3 Висновки

Отже, у підрозділі 2.1 було вдосконалено математичну модель алгоритму вибору оптимального раціону харчування, робота якого базується на використанні формул (1) – (8), визначено його переваги та недоліки, а також запропоновано способи модернізації та усунення слабких сторін. Крім того, детально описано зміни, які на думку автора, доцільно внести в алгоритм, та показано, як саме ці нововведення повинні покращити роботу алгоритму.

У підрозділі 2.2 вдосконалено процес вибору оптимального раціону харчування за допомогою використання модифікованого алгоритму у поєднанні з експертною системою. Крім того, покращено структуру експертної системи для формування оптимального раціону харчування за рахунок використання вдосконаленої продукційної моделі та описано переваги запропонованих нововведень.

3 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ

3.1 Проектування архітектури програмної системи

Для того, щоб розробити якісний програмний продукт, перед програмною реалізацією необхідно спроектувати продуману архітектуру, тобто визначити, з яких елементів система буде складатися і як ці елементи будуть взаємодіяти між собою. Оскільки у попередньому розділі було зроблено висновок, що найкращим методом для вирішення поставленої проблеми є експертна система (ЕС), а також проаналізовано, з яких частин складається загальна структура таких систем (рисунок А.1), то доцільно буде розглянути класифікацію експертних систем.

Експертні системи поділяються за наступними ознаками.

За рівнем складності структури ЕС розрізняють:

– поверхневі системи;

– глибинні системи.

Поверхневі системи інтерпретують знання про предметну область у вигляді правил, які складаються з двох частин: умови та дії. Умова правила встановлює зразок деякої ситуації, при дотриманні якої правило може бути виконано. Пошук рішення в таких системах залежить від виконання таких правил, зразки яких зрівнюються з поточними даними.

Глибинні системи мають всі можливості поверхневих систем, а також можуть при виникненні невідомої ситуації визначити за допомогою певних загальних принципів предметної області які дії доцільно виконати.

За зв’язком з реальним часом ЕС поділяють на:

– статичні;

– квазідинамічні;

– динамічні.

Статичні експертні системи використовуються для сфер діяльності, в яких база знань та інтерпретовані дані не змінюються під час роботи системи.

Квазідинамічні ЕС використовуються в ситуаціях, в яких дані змінюються з деяким фіксованим інтервалом часу.

Динамічні ЕС розробляються та використовуються для ситуацій, коли потрібна безперервна обробка та інтерпретація даних у режимі реального часу.

За ступенем інтеграції з іншими програмами розрізняють наступні ЕС:

– автономні експертні системи, які працюють безпосередньо у режимі консультацій з користувачем для допомоги у вирішенні специфічних «експертних» завдань, які не вимагають застосовування методів обробки даних.

– гібридні експертні системи; це, як правило, програмну система, яка об'єднує стандартні пакети прикладних програм та засоби роботи зі знаннями.

За завданням, що вирішується, виділяють ЕС для:

– інтерпретації даних;

– діагностики;

– моніторингу;

– проектування;

– прогнозування;

– планування;

– навчання.

Під інтерпретацією даних при використанні експертної системи мається на увазі формування опису за результатами спостережень або даними, отриманими від різних джерел. При вирішенні такого завдання зазвичай передбачається багатоваріантний аналіз даних.

При вирішенні задачі діагностики експертні системи використовують для виявлення несправностей на основі результатів спостережень за поведінкою контрольованих систем.

Експертні системи моніторингу аналізують поведінку контрольованої системи і. порівнюючи отримані дані з критичними точками заздалегідь складеного плану, прогнозують імовірність досягнення поставленої мети. Прикладами використання таких системи є: контроль руху повітряного транспорту, спостереження за станом енергетичних об'єктів тощо.

Проектуючі експертні системи призначені для структурного синтезу конфігурації об’єктів із заздалегідь визначеними властивостями (компонентів проектованої системи) при заданих обмеженнях, водночас прогнозуючі системи використовуються для логічного дослідження можливих наслідків заданих подій або ситуацій.

Планувальні системи призначені для підготовки планів проведення послідовності операцій, що призводить до заданої мети.

Експертні системи навчання аналізують знання «учнів», відшукують пробіли в знаннях і пропонують засоби для їх ліквідації.

За способом формування рішення:

– аналізуючи системи; рішення в таких системах обирається серед множини відомих рішень на основі аналізу знань;

– синтезуючі системи, які створюють рішення на основі окремих фрагментів знань [17].

Після детального опису всіх можливих класифікацій можна обрати тип експертної системи, яка буде розроблятися. За рівнем складності наша система буде відноситись до поверхневих систем, оскільки в процесі роботи системи не очікується виникнення невідомої ситуації.

Крім того, розроблюваний продукт буде відноситись до статичних автономних ЕС, оскільки при роботі програми вхідні дані змінюватись не будуть (ні з фіксованим інтервалом часу, ні в режимі реального часу), а необхідність у використанні методів обробки даних відсутня.

Наостанок необхідно зазначити, що розроблювана система буде планувально-аналізуючою, оскільки головне завдання, яке повинна виконувати система, полягає у формуванні плану здорового харчування (планування) на основі аналізу індивідуальних особливостей користувача (аналізуюча).

Наступним кроком при проектуванні архітектури програмного продукту буде детальний аналіз та опис принципів роботи найважливіших частин системи, а саме модуля висновків.

На рисунку 3.1 представлено загальний механізм роботи модуля висновків експертної системи.

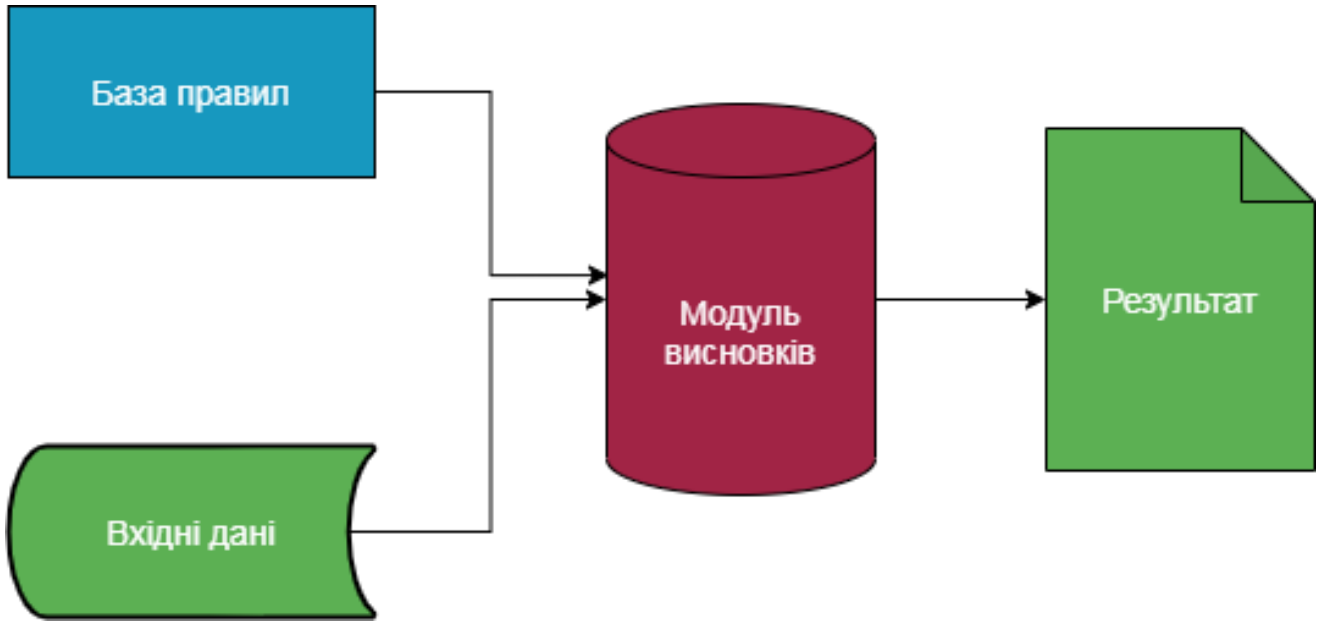


Рисунок 3.1 – Загальний механізм роботи модуля висновків

На рисунку 3.1 видно, що в загальному випадку модуль висновків формує результат (рішення) на основі вхідних даних, які можуть бути отримані від користувача або записані в базі даних, за допомогою використання правил з бази правил (або бази знань).

Цей принцип роботи прекрасно підходить для програмних систем, які отримують статичні дані в процесі роботи.

Проте, алгоритм роботи цього компонента та вхідні дані, які він використовує, визначають декілька концепцій вирішення задач.

Існує дві стратегії міркувань при роботі модуля висновків:

– зворотній ланцюг міркувань;

– прямий ланцюг міркувань.

Розглянемо спочатку стратегію зворотного ланцюга міркувань. Ця стратегія, по суті, є методом отримання висновку, який працює у зворотному напрямку від мети, тобто вхідними даними при реалізації цієї стратегії виступають результати

деяких дій. За допомогою цієї стратегії експертна система з’ясовує відповідь на запитання: «Чому це сталося?».

На основі того, що вже сталося, механізм міркувань намагається з’ясувати, які умови могли бути е минулому, щоб цей результат відбувся. Ця стратегія використовується для з’ясування причин (наприклад, для діагностики виникнення хвороби або неполадок у системі).

Крім того, цю стратегію застосовують в автоматизованому доведенні теорем, машинному формування висновків та інших напрямках штучного інтелекту. Принцип роботи цієї стратегії представлений на рисунку 3.2.

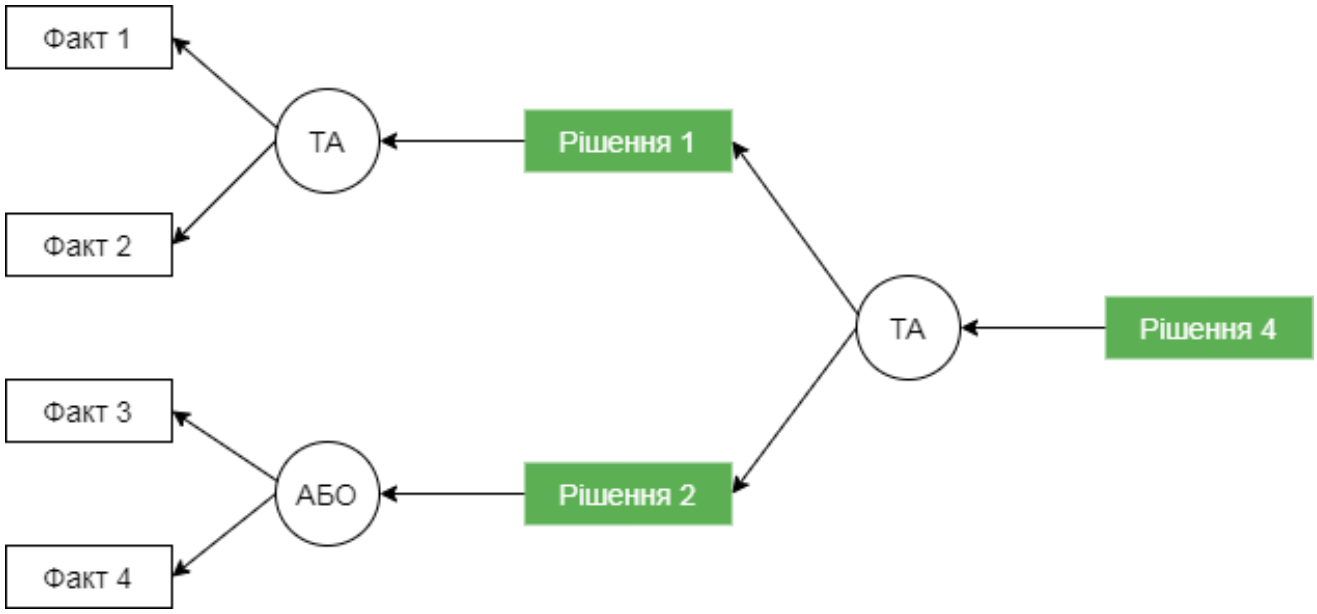


Рисунок 3.2 – Механізм роботи зворотного ланцюга міркування

Протилежним до методу зворотного ланцюга є метод прямого ланцюга міркувань; вхідними даними при роботі цього підходу виступають факти (наприклад, інформація про користувача, база даних страв тощо). Ця стратегія експертної системи відповідає на запитання: «Що може бути далі?».

Принцип роботи цієї стратегії представлений на рисунку 3.3.

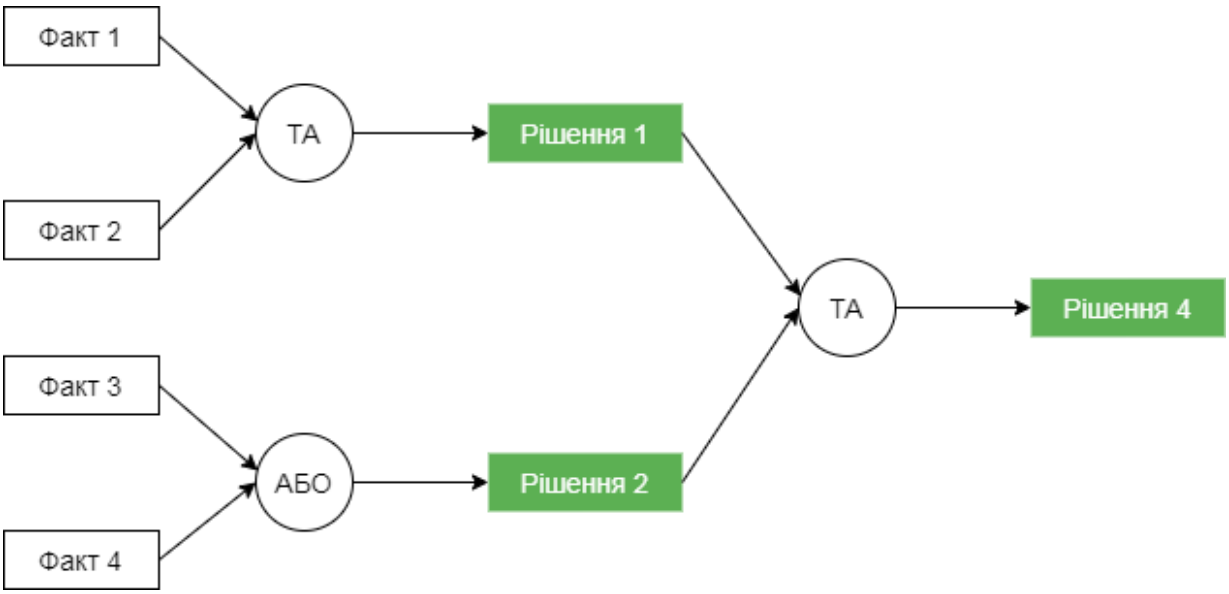


Рисунок 3.3 – Механізм роботи прямого ланцюга міркування

Тут механізм міркувань слідує за ланцюгом умов та висновків і, нарешті, виводить результат. Він розглядає всі факти і правила та сортує їх, перш ніж приймати рішення. Ця стратегія використовується для роботи над висновком, результатом чи ефектом. Прикладом використання цієї стратегії є прогнозування статусу ринку акцій.

Таким чином, з двох існуючих підходів до розробки експертних систем для вирішення поставленої проблеми може бути використана лише стратегія прямого ланцюга міркувань, адже при роботі системи не потрібно з’ясовувати причин виникнення того чи іншого результату, натомість потрібно визначити, яким буде результат при поглинанні тієї чи іншої страви та вибрати серед усіх результатів найбільш оптимальний для поставленої цілі.

3.2 Вимоги до програмно-технічного середовища функціонування програмної системи

Вимоги до функціональних характеристик

Програмна система повинна забезпечувати виконання наступних функцій:

– створення раціону для запобігання набору зайвої маси тіла;

– створення раціону для схуднення;

– створення раціону для набору ваги;

– створення раціону для лікувально-відновлюваного харчування;

– складання індивідуальної програми харчування на вибраний користувачем проміжок часу, в залежності від маси тіла і статі;

– можливість корегувати харчування для дорослих;

– збереження даних про створений раціон.

Вимоги до надійності

Для забезпечення надійного функціонування програма повинна мінімізувати кількість помилок зроблених людиною за рахунок врахування у вихідному коді виняткових ситуацій та інформування про них користувача. Крім того, програма повинна повідомляти користувача при виникненні помилок і надавати йому інформацію про причини їх виникнення.

Таким чином, надійність роботи програмного продукту залежить від надійного функціонування персонального комп'ютера та встановленої на ньому операційної системи. Тобто надійне функціонування системи забезпечується шляхом врахування усіх можливих некоректних дій користувача.

Вимоги до складу та параметрів технічних засобів

Для використання програмного забезпечення необхідні такі мінімальні характеристики:

– процесор з тактовою частотою не менше 2 ГГц;

– оперативна пам'ять 4 Гб;

– об'єм жорсткого диску має бути не менше 512 Гб;

– операційна система – Windows;

Вимоги до інформаційної та програмної сумісності

Системні програмні засоби, що використовуються програмним комплексом, мають бути представлені операційною системою Microsoft Windows 7.

Вихідні коди програмного комплексу повинні бути реалізовані мовою Java в інтегрованому середовищі NetBeans IDE.

Для збереження даних повинна використовуватись реляційна система для управління базами даних MySQL.

Вимоги до захисту інформації і програм не пред’являються.

Вимоги до транспортування та зберігання

Програма поставляється на лазерному носії інформації.

Програмна документація надається в електронному та друкованому вигляді.

Умови експлуатації програмного забезпечення збігаються з умовами експлуатації ПК.

Вимоги до програмної документації

Склад програмної документації повинен включати:

– технічне завдання;

– пояснювальна записка до дипломної роботи;

– керівництво користувача;

– програму та методику випробувань.

Техніко-економічні показники.

Орієнтовна економічна ефективність не розраховується. Аналогія не проводиться через унікальність вимог, що пред’являються до розробки вибраного програмного продукту.

3.3 Проектування програмного забезпечення для вирішення задачі

3.3.1 Аналіз та автоматизація обробки потоків даних

Наступним кроком після детального проектування архітектури та розробки вимог до програми необхідно описати яким чином рухаються та перетворюються дані при роботі з системою. Для цього використаємо графічне представлення цих процесів у вигляді діаграм потоків даних (Data Flow Diagram).

Базуючись на функціональних вимогах, розробимо загальну модель потоків даних при роботі з програмною системою (рисунок 3.4).

Рисунок 3.4 – Загальна діаграма потоків даних

На рисунку 3.4 видно, що з розроблюваною системою взаємодіє тільки користувач (тут немає адміністраторів, модераторів, диспетчерів тощо). Він вводить у систему всі необхідні дані про себе та отримує результат, а саме – сформований індивідуальний раціон харчування.

Аналогічним чином створимо більш детальне представлення інформаційних потоків, які циркулюватимуть в розроблюваному програмному забезпеченні. Так, діаграма декомпозиції першого рівня зображена на рисунку 3.5.

На рисунку 3.5 представлено повний шлях перетворення даних.

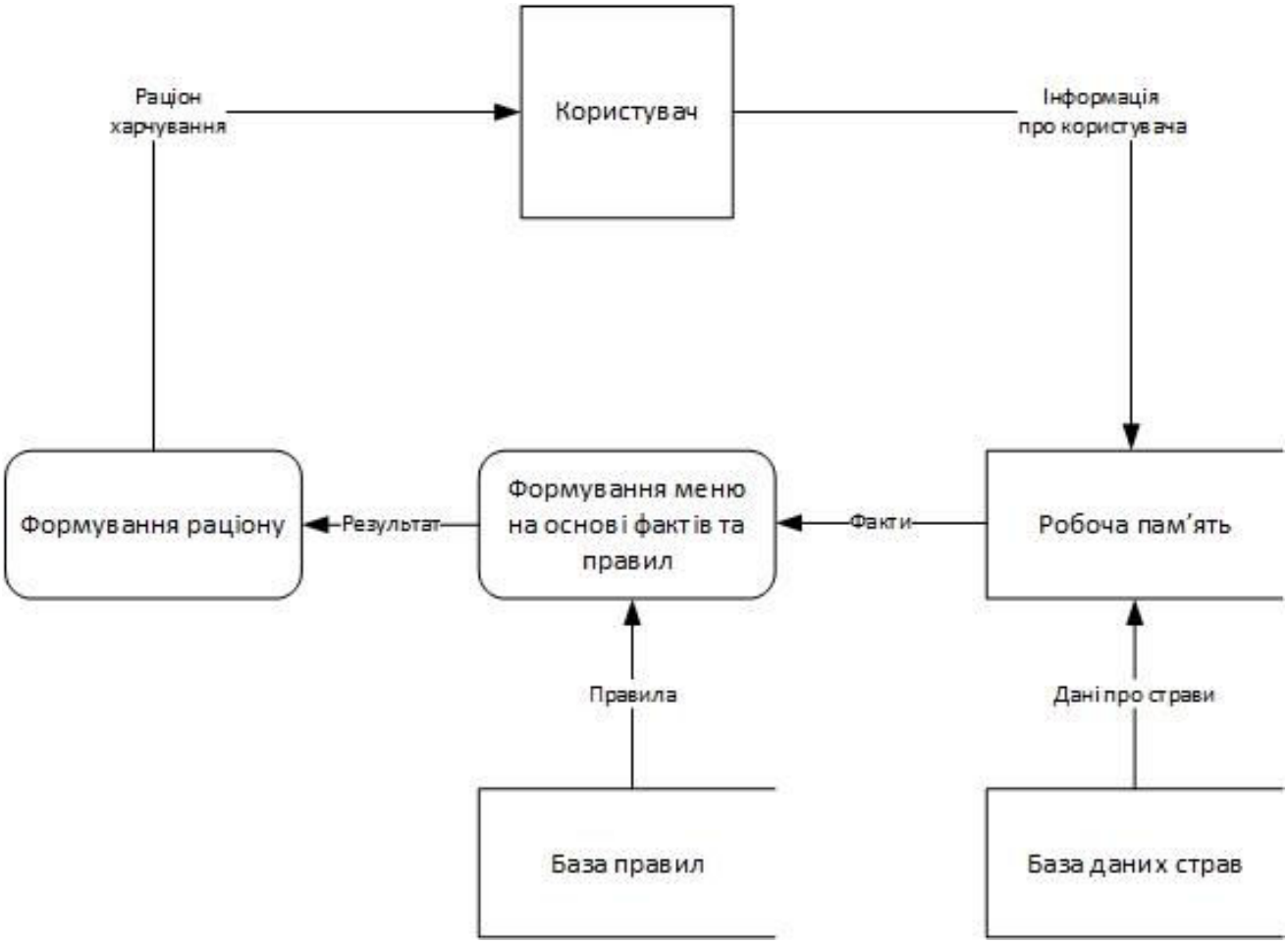


Рисунок 3.5 – Діаграма декомпозиції першого рівня

Таким чином, вхідні дані після введення користувачем потрапляють у робочу пам’ять разом з інформацією про страви, яка зберігається у базі даних страв. Далі, в ході формування меню, вся інформація з робочої пам’яті у вигляді фактів поступово завантажується у модуль висновків і обробляється, використовуючи всі існуючі правила із бази правил.

Після цього список найбільш оптимальних страв для досягнення цілі користувача з врахування особливостей його організму, а також інформація про рекомендовану кількість калорій, розмір порцій тощо, потрапляють у підсистему формування раціону, де на основі отриманої з модуля висновків інформації, базуючись на формулах, описаних у попередньому розділі, формується оптимальний для користувача раціон харчування. Далі весь сформований раціон надходить до інтерфейсу користувача, де відображається у зручній формі.

Таким чином, в ході аналізу системи розглянуто основні інформаційні потоки та створено відповідні діаграми для кращого розуміння процесів, які будуть відбуватися в системі.

3.3.2 Проектування схеми бази даних

На основі спроектованої архітектури та сформульованих вимог до функціональності розроблюваної системи можна розпочати створення бази даних.

Перш за все для роботи експертної системи потрібні вхідні дані (факти) про користувача та загальний список страв з інформацією про поживну цінність, вміст білків, жирів та вуглеводів. Крім того, потрібно також зберігати дані про вміст алергенів у стравах, адже це вкрай важливий аспект, який потрібно враховувати, щоб не погіршити стан здоров’я користувача.

Всі ці дані можна зберігати у базі даних, втім створення окремої сутності і, відповідно, збільшення обсягів бази даних для збереження інформації про користувача не є обов’язковими, оскільки кожен раз при використанні системи інформація про користувача вводиться окремо і вона не має безпосереднього зв’язку з іншими даними системи. Інакше кажучи, зберігати дані про користувача немає сенсу, тому що вони, по суті, є одноразовими.

Таким чином, в базі даних доцільно зберігати лише інформацію про страви. Її можна знайти у відкритому доступі у мережі, наприклад, в базі даних міністерства сільського господарства США під назвою USDA National Nutrient Database. Втім, вказана база даних містить надто мало страв та продуктів, які є актуальними для нашого регіону, а це означає, що її використання є неефективним. Тому скористаємось інформацією, яка знаходиться у відкритому доступі на одному із популярних сайтів здорового харчування у нашому регіоні.

Вся отримана таким чином інформація містить дані про кількість калорій, вміст білків, жирів та вуглеводів з розрахунку на 100 грам страви, а це означає,

що після завантаження бази даних потрібно буде ще виділити наявність алергенів у кожній страві.

Отже, для розроблюваної програми достатньо буде створити базу даних лише з однією сутністю «Страва», проте для зручності роботи доцільно поділити всі страви за їх групою і відповідно створити шість різних сутностей:

- перше;
- друге;
- напій;
- салат;
- гарнір;
- каша.

Таким чином, первинні ключі та атрибути кожної з перерахованих сутностей наведені у таблиці 3.1.

Таблиця 3.1 – Сутності, їх атрибути та первинні ключі

Сутність	Атрибути	Первинний ключ
Перше	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви
Друге	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви
Напій	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви
Салат	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви

Кінець таблиці 3.1

1	2	3
Гарнір	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви
Каша	Назва, калорійність (ккал/100г) вміст білків (г/100г), вміст жирів (г/100г),вміст вуглеводів (г/100г), алергени	Код страви

Як видно з таблиці 3.1, всі сутності мають однакові атрибути та первинні ключі, тому зупинятися на структурі окремих сутностей немає сенсу. Переходимо до наступного частини проектування системи.

3.3.3 Проектування інтерфейсу користувача

Одним з головних етапів при розробці програмного засобу є проектування інтерфейсу користувача, оскільки це єдиний засіб взаємодії з користувачем, який безпосередньо впливає на зручність роботи з програмою.

По-перше, розроблювана програмна система повинна містити лише одне вікно, на якому мають бути такі вкладки, як: вкладка для введення загальної інформації про користувача, вкладка для введення інформації про особливості організму користувача (наявність алергій і різних захворювань), а також вкладка для виведення результату роботи програми.

По-друге, основне вікно системи повинно містити кнопки для переходу між вкладками, а також кнопки для відображення довідки роботи з програмою.

По-третє, одразу після запуску програмної системи повинно з’явитися вікно з активною вкладкою введення загальної інформації про користувача, на якій мають бути розташовані контейнери для введенні відповідних даних.

Таким чином, вкладка введення загальної інформації має містити контейнери для даних про:

– стать;
– вік;
– зріст;
– вагу;
– систолічний тиск та діастолічний тиск (у стані спокою);
– пульс (у стані спокою);
– рівень активності;
– ціль використання програми.

Однак, для деяких з цих даних доцільно буде використовувати різні контейнери, такі, як:

– поля зі списком;
– лічильники.

Це потрібно для того, щоб користувач міг вибрати дані зі списку, замість того, щоб кожен раз вводити однотипні дані, такі як стать, рівень активності та ціль використання програми. Аналогічним чином всі числові дані зручніше вводити в поле-лічильник, тому для віку, зросту, ваги, систолічного та діастолічного тиску, а також пульсу повинні бути окремі лічильники.

Отже, після введення даних на першій вкладці при переході на наступну вкладку повинна бути перевірка, чи всі необхідні дані введено і чи вони є коректними. Відповідно, користувач має побачити діалогове вікно помилки, якщо хоча б одна з перерахованих умов не була виконана.

Якщо ж всі необхідні поля заповнені адекватними даними, тоді при натисканні на відповідну кнопку має відбутися перехід на наступну вкладку. На цій вкладці має знаходитися міні-опитування, в якому необхідно вказати, чи є в користувача алергія на:

– молочні продукти;
– морепродукти;

– злаки;

– цитрусові;

– горіхи;

– гриби;

– бобові

– мед.

Для цього у вкладці має бути створена група перемикачів з назвами «Так» і «Ні» для кожного алергенна. Аналогічним чином у вкладці має бути подібне міні-опитування про наявність таких хвороб, як:

– виразка шлунка або гастрит (обмеження раціону харчування при цих хворобах співпадають);

– захворювання нирок;

– ожиріння;

– цукровий діабет;

– серцево-судинні захворювання;

– панкреатит;

– захворювання печінки;

– захворювання нервової системи.

Після відповіді на всі питання та натисканні на відповідну кнопку має відбутися перехід на останню вкладку, де має відобразитися результат роботи програми у текстовому форматі.

3.4 Аналіз та вибір технологій і методів реалізації проектованої системи

Останнім кроком перед програмною реалізацією системи буде визначення інструментарію за допомогою якого буде виконуватись процес реалізації. Головним інструментом для реалізації будь-якого програмного продукту є мова програмування, тому почнемо з вибору мови.

Серед всіх існуючих мов для розробки системи вибору оптимального раціону харчування було обрано мову програмування Java. Щоб обґрунтувати цей вибір перелічимо переваги цієї мови програмування:

– незалежність від платформи, на якій виконуються програми;

– легкість розробки складних проектів;

– наявність широкого вибору інструментів;

– висока надійність.

Далі оберемо систему управління базою даних (СУБД). Для цього нам необхідно визначити вимоги до СУБД, таким чином вона повинна:

– бути безкоштовною або мати некомерційну версію;

– бути з відкритим вихідним кодом;

– підтримувати обсяги даних до 1 Тб.

Із рішень з відкритим вихідним кодом, що відповідають вказаним вимогам, найбільш популярною та зручною є MySQL, тому оберемо саме її.

Разом з тим необхідно розглянути і обрати інструмент для розробки експертних систем, написаний під обрану нами мову Java, або який можна використовувати при створенні програм на Java. До таких інструментів відносять систему для розробки експертних систем Jess (Java Expert System Shell), системи управління бізнес-правилами OpenL Tablets та JBoss Drools, фреймворки Java під назвами Easy Rules та RuleBook та інші. Кожен з перерахованих інструментів має свої переваги та недоліки, які перелічувати немає сенсу, тому зупинимось тільки на обраній технології.

На сьогодні найкращим інструментом для створення експертних систем на основі правил є система управління бізнес-правилами (або система створення правил) JBoss Drools.

Основними особливостями цього інструменту є:

– підтримка можливості створення правил на основі прямого ланцюга висновків;

– підтримка можливості створення правил на основі зворотного ланцюга висновків;

– підтримка стандарту Java Specification Request 94;

– використання алгоритму Rete, який забезпечує вищу ефективність роботи експертної системи.

Крім того, ця система є кросплатформенною, вона добре підтримується, має вичерпну документацію (на відміну від інших розглянутих технологій) та зрозумілу мову написання правил, що дозволяє легко її використовувати.

Отже, для того, щоб перейти до створення спроектовано програмного продукту, залишається обрати середовище розробки.

NetBeans IDE – це вільне інтегроване середовище розробки для мов програмування Java, Python, JavaFX, PHP, C/C++, HTML5, JavaScript та інших. Середовище можна встановити як з підтримкою окремих мов, так і у повній конфігурації. Середовище розробки NetBeans за замовчуванням підтримує розробку для платформ J2SE і J2EE.

За якістю і можливостями останні версії NetBeans IDE входять у список найкращих середовищ розробки для мови Java, підтримуючи рефакторинг, профілювання, виділення синтаксичних конструкцій кольором, авто-доповнення мовних конструкцій на льоту, шаблони коду тощо.

3.5 Висновки

Підводячи підсумок, слід сказати, що у розділі було спроектовано і описано архітектуру та структуру майбутньої програмної системи.

Перш за все було проведено аналіз існуючих підходів до розробки обраної архітектури, яка зможе задовольнити всі сформульовані в попередніх розділах вимоги до створюваної програмної системи. Проаналізувавши всі існуючі на даний момент стратегії рішення, вивчивши їх принципи роботи, а також переваги та недоліки, було встановлено, що експертна система на основі правил зможе найкраще задовольнити вимоги до проектованої програмної системи.

Наступним кроком було описано вимоги до функціональних характеристик і надійності програмного продукту, до складу технічних засобів та їх параметрів, до інформаційної і програмної сумісності, а також до програмної документації.

Далі було здійснено аналіз рухів і перетворень даних при роботі з системою, а також створено загальну діаграму потоків даних та діаграму декомпозиції першого рівня, кожна з яких була детально описана.

Крім того, було описано мету створення бази даних, обґрунтовано доцільність та спроектовано базу даних для збереження інформації про страви.

Також було спроектовано і детально описано з яких компонентів повинен складатися користувацький інтерфейс, що ці компоненти повинні містити і які функції мають виконувати.

У останньому підрозділі було проаналізовано можливі варіанти технологій для створення експертної системи, а також обрано мову програмування, інтегроване середовище програмування та технології, які найкраще підійдуть для реалізації програмної системи.

Результатом написання цього розділу є виокремлення всіх характеристик, необхідних для реалізації програмної системи.

4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ

4.1 Детальна реалізація

Як було зазначено у попередньому розділі, розроблена програмна система для вибору оптимального раціону харчування складається з шести частин:

- інтерфейс користувача;
- модуль висновків;
- підсистема формування раціону;
- база правил;
- робоча пам’ять;
- база даних.

Всі описані компоненти доцільно поділити за призначенням на функціональні та нефункціональні.

До перших відносяться лише три модулі, а саме:

- інтерфейс користувача;
- модуль висновків експертної системи;
- модуль формування порцій.

Відповідно, всі інші компоненти є нефункціональними, а оскільки вони теж є необхідними для роботи програми, тому для роботи з ними теж необхідно створити пакети допоміжних класів.

Так, для роботи з базою даних створено пакет під назвою db_models. До цього пакету входять наступні класи:

- Beverage;
- Dish;
- FirstCourse;
- Getter;
- Porridge;
- Salad;

– SecondCourse;

– SideDish.

Діаграму класів пакета db_models представлено на рисунку 4.1.

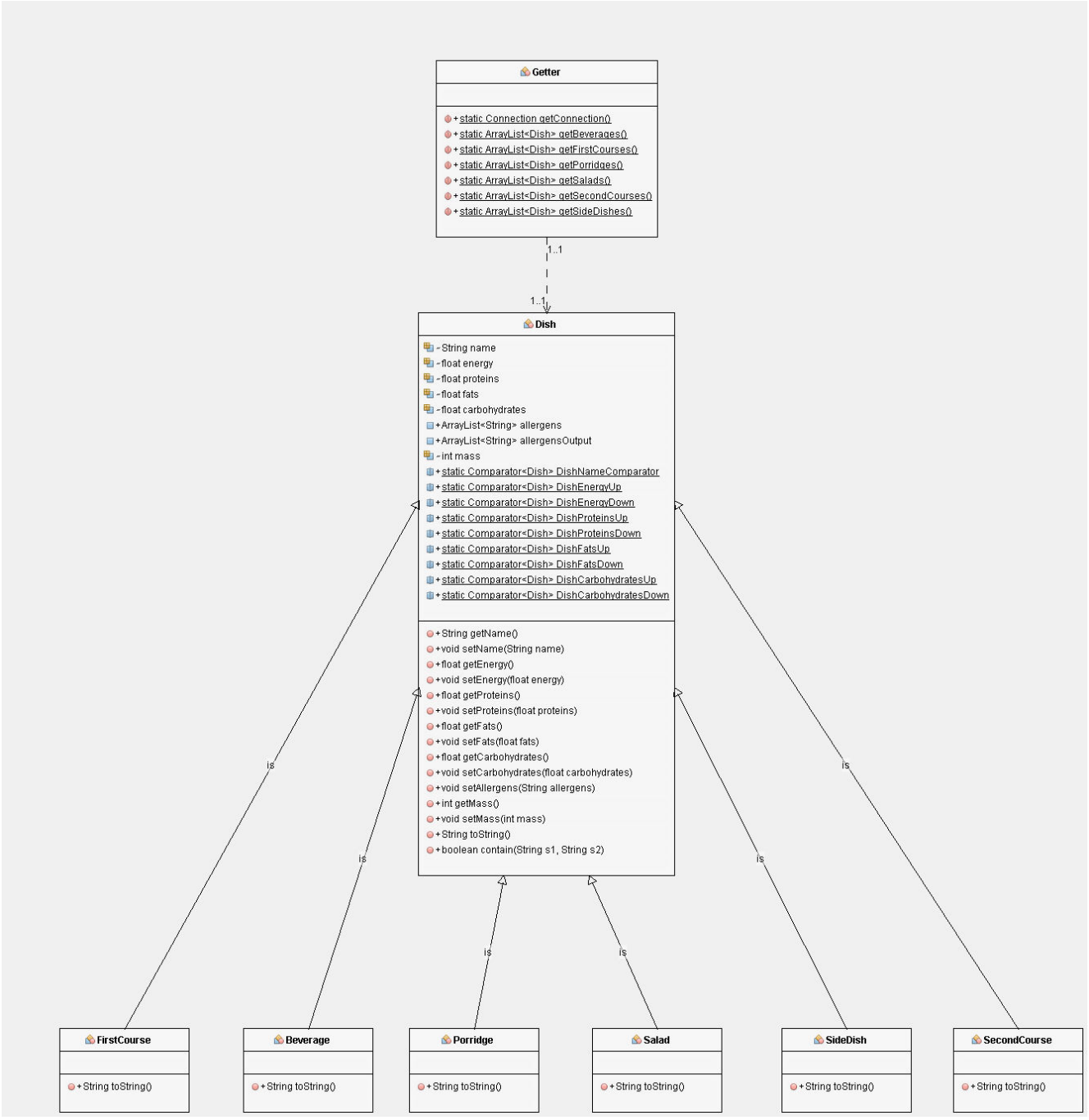


Рисунок 4.1 – Діаграма класів пакета db_models

Розглянемо кожен з перерахованих класів детальніше.

Всі класи в пакеті db_models, крім Getter та Dish, представляють собою модель даних окремої сутності бази даних. У свою чергу, клас Dish представляє собою загальну модель страви, яку наслідують класи Beverage, FirstCourse, Porridge, Salad, SecondCourse та SideDish, а клас Getter реалізує всю логіку отримання інформації з бази даних.

Відповідно, об’єкти кожного з перерахованих класів-спадкоємців необхідні для збереження даних про різні типи страв.

Клас Beverage зберігає дані про напої, клас FirstCourse містить дані про перші страви, клас Porridge необхідний для збереження даних про каші, об’єкти класу Salad зберігають дані про салати, клас SecondCourse містить дані про другі страви, а клас SideDish зберігає дані про гарніри.

Дослідимо батьківський клас Dish. Він містить наступні поля:

– поле типу String для назви страви,

– чотири поля типу float для збереження харчової цінності страви, кількості білків, жирів та вуглеводів;

– два типізованих списки ArrayList<String> для інформації про алергени, які містяться в страві;

– поле типу int для маси страви, яке буде потрібне при формуванні порцій;

– вісім типізованих статичних полів типу Comparator, які потрібні для сортування списків страв за різними критеріями.

Цей клас, крім перерахованих полів, містить лише гетери та сетери, тому зупинятися на ньому більше немає сенсу.

Останнім розглянемо клас Getter. Він не має полів, виступає у ролі контакту між системою та базою даних і містить лише сім статичних методів, один з яких призначений для створення підключення до бази даних і шість інших для отримання інформації з БД про різні видів страв.

Перейдемо до аналізу наступного пакету. Єдиною метою існування пакету person_models є представлення даних про користувача системи. Він містить лише один клас Person і п’ять перерахувань для зручності збереження інформації про стать користувача, його ціль, рівень активності, а також захворювання і алергії.

Діаграму класів цього пакету зображено на рисунку 4.2

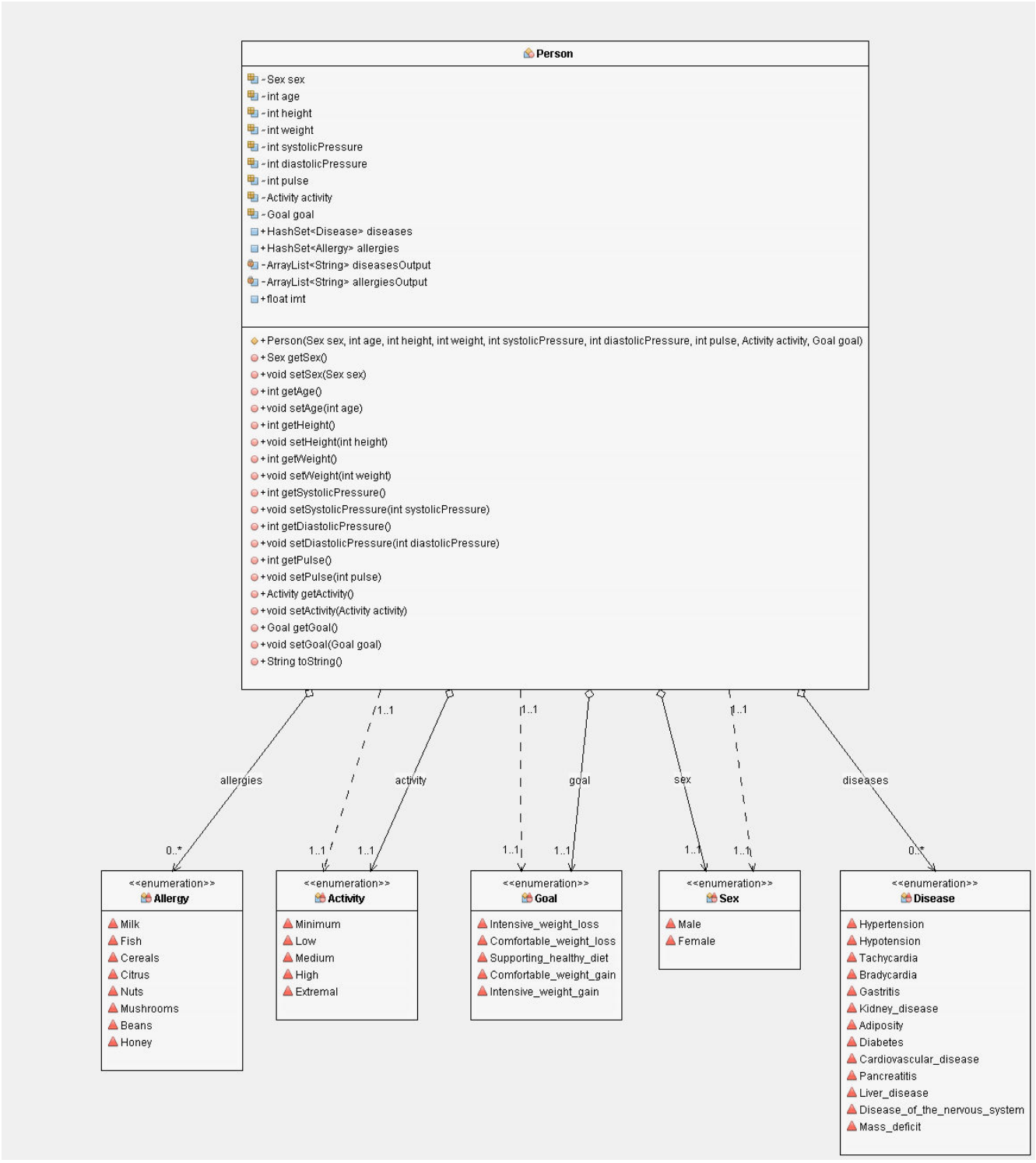


Рисунок 4.2 – Діаграма класів пакета person_models

Розберемо клас Person детальніше.

Він містить наступні поля:

– шість полів типу int для інформації про вік, вагу, зріст, артеріальних тиск (систолічний та діастолічний) і пульс в стані спокою;

– дві типізованих множини HashSet<Disease> і HashSet<Allergy>, які містять інформацію про захворювання і алергії користувача відповідно

– два типізованих списки ArrayList<String> для виведення інформації про захворювання і алергії;

– поле типу float для зберігання Індексу маси тіла користувача.

Оскільки вказаний клас необхідний тільки для збереження даних про користувача, то, крім описаних полів, він має лише гетери та сетери.

Таким чином, ми розглянули всі допоміжні класи нефункціональних компонентів системи, втім розглянуті пакети не єдині. В проекті існують ще три менші пакети, які виконують інші функції та містять лише по одному класу.

Повна структура проекту представлена на рисунку 4.3.

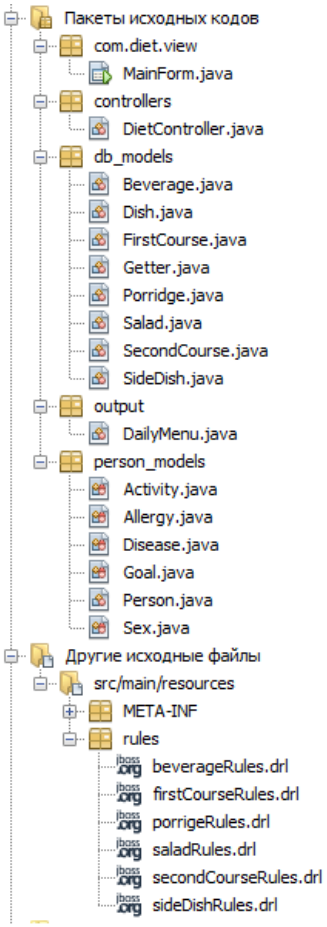


Рисунок 4.3 – Повна структура проекту

Детальна діаграма класів представленого проекту представлена у додатку В на рисунку В.1.

Тому детально розглянемо ці пакети, розпочавши з пакету, який містить основну логіку роботи програми, а саме controllers.

В цьому пакеті знаходиться лише один клас під назвою DietController. Він реалізує всю логіку роботи з модулем висновків експертної системи та формування порцій.

Безпосередньо модуль висновків експертної системи представлений двигуном правил Drools, вся робота з яким виконується за допомогою класів, що знаходяться в бібліотеці kie-api, а саме KieServices, KieContainer та KieSession.

Перші два класи необхідні лише для створення об’єкта класу KieSession і завантаження правил в нього. У свою чергу, клас KieSession являє собою найпоширеніший і найзручніший спосіб взаємодії з двигуном правил. Він дозволяє програмі встановити ітераційну розмову з механізмом, де стан сеансу зберігається у всіх викликах. Процес міркування може запускатися кілька разів для одного і того ж набору даних. Однак після того, як програма закінчить використовувати сеанс, вона повинна викликати метод dispose(), щоб звільнити ресурси та використовувану пам’ять.

Приклад створення об’єкта класу KieSession наведено нижче:

```
KieServices ks = KieServices.Factory.get();
KieContainer kContainer = ks.getKieClasspathContainer();
KieSession kSession = kContainer.newKieSession("ksession-rules");
```

Таким чином, при створенні об’єкту класу KieSession спочатку ініціалізуються об’єкти класів KieServices та KieContainer, після чого виконується завантаження правил у робочу пам’ять двигуна правил з бази знань, яка знаходиться в пакеті rules. Далі створений об’єкт kSession використовується для завантаження фактів за допомогою методу insert() і встановлення у двигуні правил глобальних змінних за допомогою методу setGlobal().

Глобальні змінні необхідні для того, щоб можна було відсортувати всі отримані з бази даних страви та повернути їх для подальшого використання в модулі формування раціону.

Після завантаження всіх правил, фактів та глобальних змінних необхідно запустити виконання правил за допомогою методу fireAllRules(). Використання цього методу запустить двигун правил і надасть можливість отримати змінені двигуном глобальні змінні за допомогою методу getGlobal().

Метод використання експертної системи в класі DietController представлений нижче:

```
public String useExpertSystem() {
    try {
        KieServices ks = KieServices.Factory.get();
        KieContainer kContainer = ks.getKieClasspathContainer();
        KieSession kSession = kContainer.newKieSession("ksession-rules");
        kSession.insert(person);
        kSession.setGlobal("allBeverages", beverages);
        kSession.setGlobal("allFirstCourses", firstCourses);
        kSession.setGlobal("allPorriges", porriges);
        kSession.setGlobal("allSalads", salads);
        kSession.setGlobal("allSecondCourses", secondCourses);
        kSession.setGlobal("allSideDishes", sideDishes);
        kSession.fireAllRules();
        beverages = (ArrayList) kSession.getGlobal("allBeverages");
        firstCourses = (ArrayList) kSession.getGlobal("allFirstCourses");
        porriges = (ArrayList) kSession.getGlobal("allPorriges");
        salads = (ArrayList) kSession.getGlobal("allSalads");
        secondCourses = (ArrayList) kSession.getGlobal("allSecondCourses");
        sideDishes = (ArrayList) kSession.getGlobal("allSideDishes");
        kSession.dispose();
        return formOutput();
    } catch (Throwable t) {
        t.printStackTrace();
    }
    return "Експертна система не запустилась";
}
```

В представленому фрагменті коду видно, що в методі useExpertSystem() перш за все створюється єдиний об’єкт класу KieSession, за допомогою якого

виконується вся робота з експертною системою. Далі у робочу пам’ять завантажуються факти про користувача системи і шість глобальних змінних для кожного типу страви. Після чого виконується запуск обробки фактів правилами, отримуються набори страв кожного типу, які підходять конкретному користувачу, а також звільняються ресурси та використовувана пам’ять.

Метод формування раціону наведено нижче:

```
public String formOutput() {
      String output = "";
      int personCalories = (int) calculateCalories(person);
      output += "Добова норма калорій: " + personCalories + "\n";
      for (int i = 0; i < 7; i++) {
         DailyMenu day = new DailyMenu();
         day.number = i + 1;
         day.dailyCalories = personCalories;
         formBreakfast(day, (personCalories / 3));
         formLunch(day, (personCalories / 3));
         formDinner(day, (personCalories / 3));
         output = output + day.toString() + "\n";
      }
      return output;
}
```

В цьому методі використовується клас з іншого пакету, а саме DailyMenu з пакету output, його ми розглянемо дещо пізніше. А зараз опишемо метод підрахунку добової норми калорій.

Метод підрахунку добової норми калорій наведено нижче:

```
public int calculateCalories(Person person) {
   if (null == person.getGoal()) {
      return (int) calculateBasalMetabolicRate(person);
   } else {
      switch (person.getGoal()) {
         case Intensive_weight_loss:
            return (int) ((int) calculateBasalMetabolicRate(person) * 0.8);
         case Comfortable_weight_loss:
            return (int) ((int) calculateBasalMetabolicRate(person) * 0.9);
         case Supporting_healthy_diet:
```

```
            return (int) calculateBasalMetabolicRate(person);
        case Comfortable_weight_gain:
            return (int) ((int) calculateBasalMetabolicRate(person) * 1.1);
        case Intensive_weight_gain:
            return (int) ((int) calculateBasalMetabolicRate(person) * 1.2);
        default:
            return (int) calculateBasalMetabolicRate(person);
      }
   }
}
```

Метод формування сніданку наведено нижче:

```
public void formBreakfast(DailyMenu menu, double breakfastCalories) {
   int firstCourse_rnd = getRandomInt(0, firstCourses.size() - 1);
   int secondCourse_rnd = getRandomInt(0, secondCourses.size() - 1);
   int beverage_rnd = getRandomInt(0, beverages.size() - 1);
   int firstCourse_mass;
   int secondCourse_mass;
   int beverage_mass;
   while (menu.breakfast.size() < 3) {
      if (!menu.allDishes.contains(firstCourses.get(firstCourse_rnd))
            && !menu.allDishes.contains(secondCourses.get(secondCourse_rnd))
            && !menu.allDishes.contains(beverages.get(beverage_rnd))) {
         firstCourse_mass = (int) ((breakfastCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
         Dish dish1 = firstCourses.get(firstCourse_rnd);
         dish1.setMass(firstCourse_mass);
         menu.breakfast.add(dish1);
         secondCourse_mass = (int) ((breakfastCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
         Dish dish2 = secondCourses.get(secondCourse_rnd);
         dish2.setMass(secondCourse_mass);
         menu.breakfast.add(dish2);
         beverage_mass = (int) ((breakfastCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
         Dish dish3 = beverages.get(beverage_rnd);
         dish3.setMass(beverage_mass);
         menu.breakfast.add(dish3);
         menu.allDishes.add(dish1);
         menu.allDishes.add(dish2);
         menu.allDishes.add(dish3);
      } else {
```

```
            firstCourse_rnd = getRandomInt(0, firstCourses.size() - 1);
            firstCourse_mass = (int) ((breakfastCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
            Dish dish1 = firstCourses.get(firstCourse_rnd);
            dish1.setMass(firstCourse_mass);
            menu.breakfast.add(dish1);
            menu.allDishes.add(dish1);
            secondCourse_rnd = getRandomInt(0, secondCourses.size() - 1);
            secondCourse_mass = (int) ((breakfastCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
            Dish dish2 = secondCourses.get(secondCourse_rnd);
            dish2.setMass(secondCourse_mass);
            menu.breakfast.add(dish2);
            menu.allDishes.add(dish2);
            beverage_rnd = getRandomInt(0, beverages.size() - 1);
            beverage_mass = (int) ((breakfastCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
            Dish dish3 = beverages.get(beverage_rnd);
            dish3.setMass(beverage_mass);
            menu.breakfast.add(dish3);
            menu.allDishes.add(dish3);
        }
    }
}
```

Аналогічним чином також формується раціон обіду та вечері.

Повний програмний код основних модулів представлений у додатку Г.

Перейдемо до розгляду наступного пакету, а саме пакету output, який містить всього один клас.

Клас DailyMenu з пакету output необхідний для представлення моделі добового раціону.

Він містить наступні поля:

– поле типу int для збереження номеру дня і відображення його при виводі;

– три типізованих множини HashSet<Dish>, які містять страви сніданку, обіду та вечері відповідно;

– одна типізована множина HashSet<Dish>, яка містить страви всіх трапез добового раціону і необхідна для перевірки не повторюваності страв;

– поле типу int для зберігання денної кількості калорій, яку повинен вживати користувач.

Серед методів цей клас має лише метод під назвою sumCalories(), який необхідний використовується для підрахування загальної кількості калорій у всіх стравах добового раціону.

Останнім пакетом, котрий залишилось розглянути, є пакет views. Він містить тільки один клас під назвою MainFrom, який є головним класом проекту і реалізує інтерфейс програми, з яким взаємодіє користувач. Розглянемо всі вікна інтерфейсу детальніше.

При запуску програмної системи користувач побачить вікно, яке зображено на рисунку 4.4.

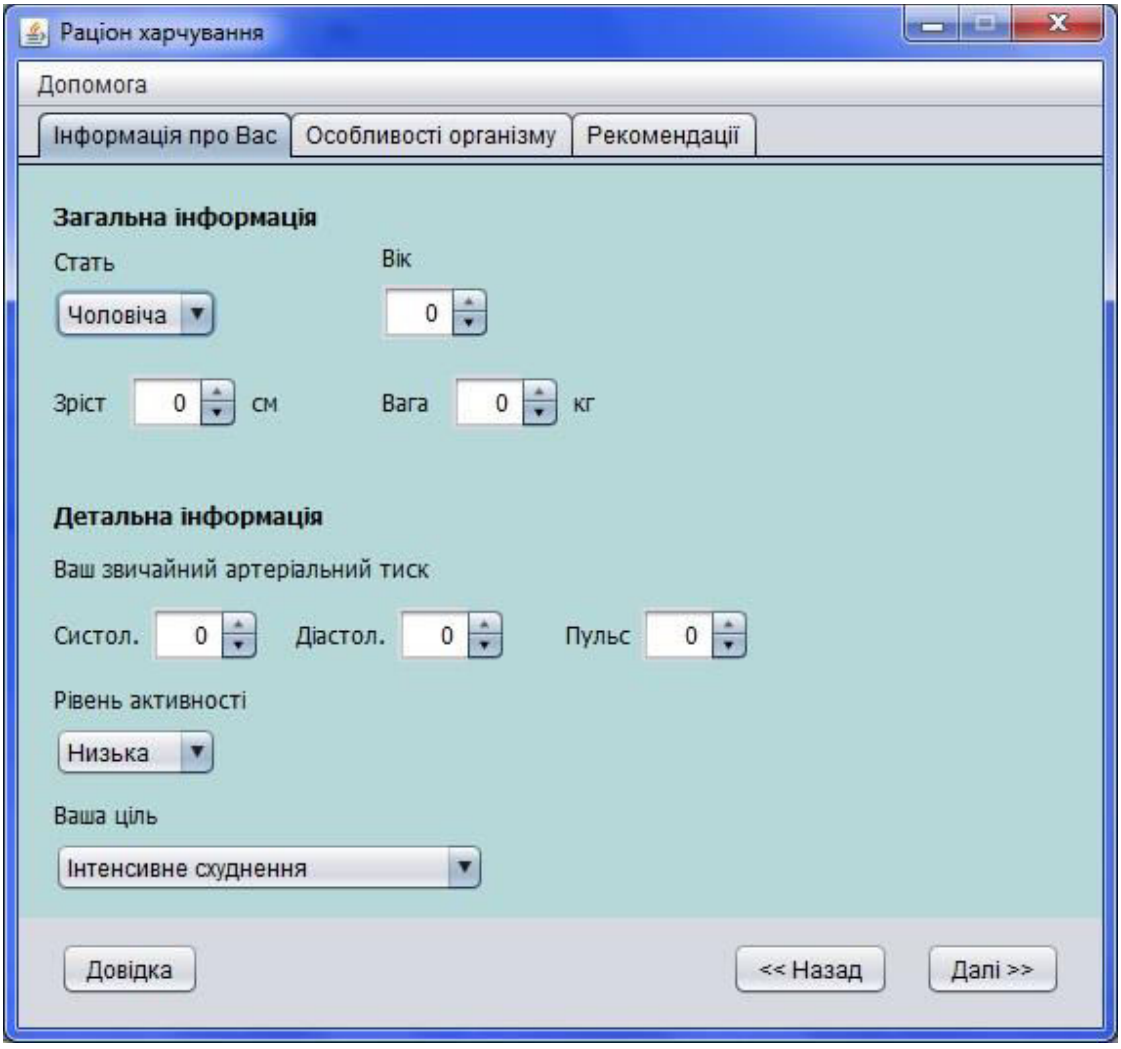


Рисунок 4.4 – Інтерфейс початкового вікна програми

Зображене вікно є єдиним функціональним в проекті, всі інші вікна програми є діалоговими. Так, вікно, яке представлено на рисунку 4.4, містить три вкладки з назвами «Інформація про Вас», «Особливості організму» та «Рекомендації» відповідно.

При запуску програми активною є вкладка інформації про користувача, яка відповідно до описаних в минулому розділі деталей містить необхідну кількість полів для введення всіх важливих даних. Перелічувати всі поля немає сенсу, оскільки вони всі зображені на рисунку 4.4, тому зупиняти на цьому не будемо і перейдемо до опису наступних вкладок.

Наступною вкладкою після «Інформації про Вас» є вкладка «Особливості організму». Вона зображена на рисунку 4.5.

Рисунок 4.5 – Вкладка особливостей організму

Ця вкладка головним чином призначена для визначення алергій і захворювань у користувача. Цей крок є вкрай важливим при розробці раціону харчування, адже алергічні реакції можуть негативно позначитися на здоров'ї та навіть призвести до анафілактичного шоку або смерті. Крім того, не варто забувати про обмеження раціону при наявності різних захворювань.

Таким чином, для використання програми користувач обов'язково повинен відповісти на всі питання, які знаходяться на вкладці. Для цього він має активувати одну з кнопок кожної пари «Так/Ні». Без заповнення цієї вкладки користувач не зможе продовжити використовувати програму. Це зроблено для того, щоб не допустити у сформованому раціоні небезпечних для користувача продуктів та елементів.

Далі розглянемо останню вкладку, яка необхідна для відображення рекомендацій по зміні раціону. Вона представлена на рисунку 4.6.

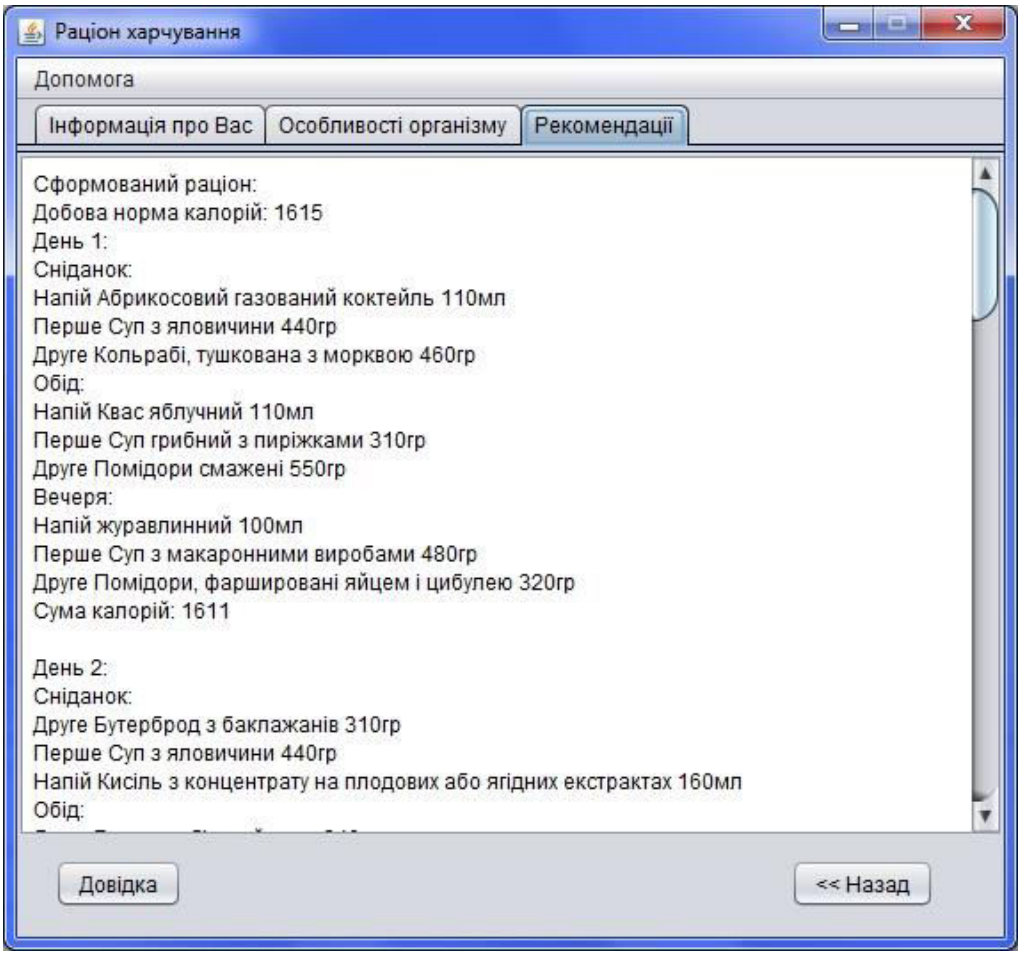


Рисунок 4.6 – Результат роботи програми

Ця вкладка містить лише текстове поле для відображення результатів формування раціону харчування користувача, тому зупиняти на ній ми не будемо і одразу перейдемо до розгляду всіх кнопок, котрі розміщені у вікні.

У головному вікні, крім кнопок навігації («Назад» і «Далі»), є кнопка «Довідка», яка відкриває діалогове вікно довідки, а також невелике меню допомоги з кнопками «Як це працює?» і «Про автора», які відкривають вікна інформації про роботу програми і про користувача відповідно. Ці вікна необхідні тільки для відображення інформації у текстовому вигляді, тому розглядати їх більш детально немає сенсу. Інструкцію користувача розробленої системи представлено у додатку Д.

4.2 Тестування програмного забезпечення

Останнім кроком при реалізації програмного продукту є тестування коректності роботи кожного розробленого модуля. Для детальної перевірки системи необхідно розробити план тестування основних частин, а саме:

- клас логіки роботи з експертною системою під назвою DietController;
- клас виводу даних під назвою DailyMenu;
- клас інтерфейсу користувача MainForm.

Тому було створено загальний план тестування для всіх методів перерахованих класів, який наведено у таблиці 4.1.

Таблиця 4.1 – План тестування

Назва тесту	Клас, який перевіряють	Опис тесту
testUseExpertSystem	DietController	Отримання сформованого меню
testFormOutput	DietController	Отримання сформованого раціону

Кінець таблиці 4.1

1	2	3
testCalculateBasalMetabolicRate	DietController	Отримання базового метаболізму користувача
testCalculateCalories	DietController	Отримання добової норми калорій для користувача
testFormBreakfast	DietController	Отримання сформованого раціону сніданку
testFormLunch	DietController	Отримання сформованого раціону обіду
testFormDinner	DietController	Отримання сформованого раціону вечері
testToString	DailyMenu	Отримання добового раціону в текстовому форматі
testSumCalories	DailyMenu	Отримання суми калорій добового раціону
testMain	MainForm	Перевірка роботи інтерфейсу

У зазначеному плані тестування головним чином перевіряється правильність роботи методів класу DietController, а вже потім методи класу DailyMenu та інтерфейсу.

Крім того, всі перераховані тести для класу DietController будуть повторені тричі з різними наборами вхідних даних, тобто вони будуть запускатись на основі даних про користувачів з різними особливостями організму і цілями. Це необхідно для детальнішої перевірки роботи цього класу.

Таким чином, результати та час проходження всіх зазначених тестів зображені на рисунку 4.7.

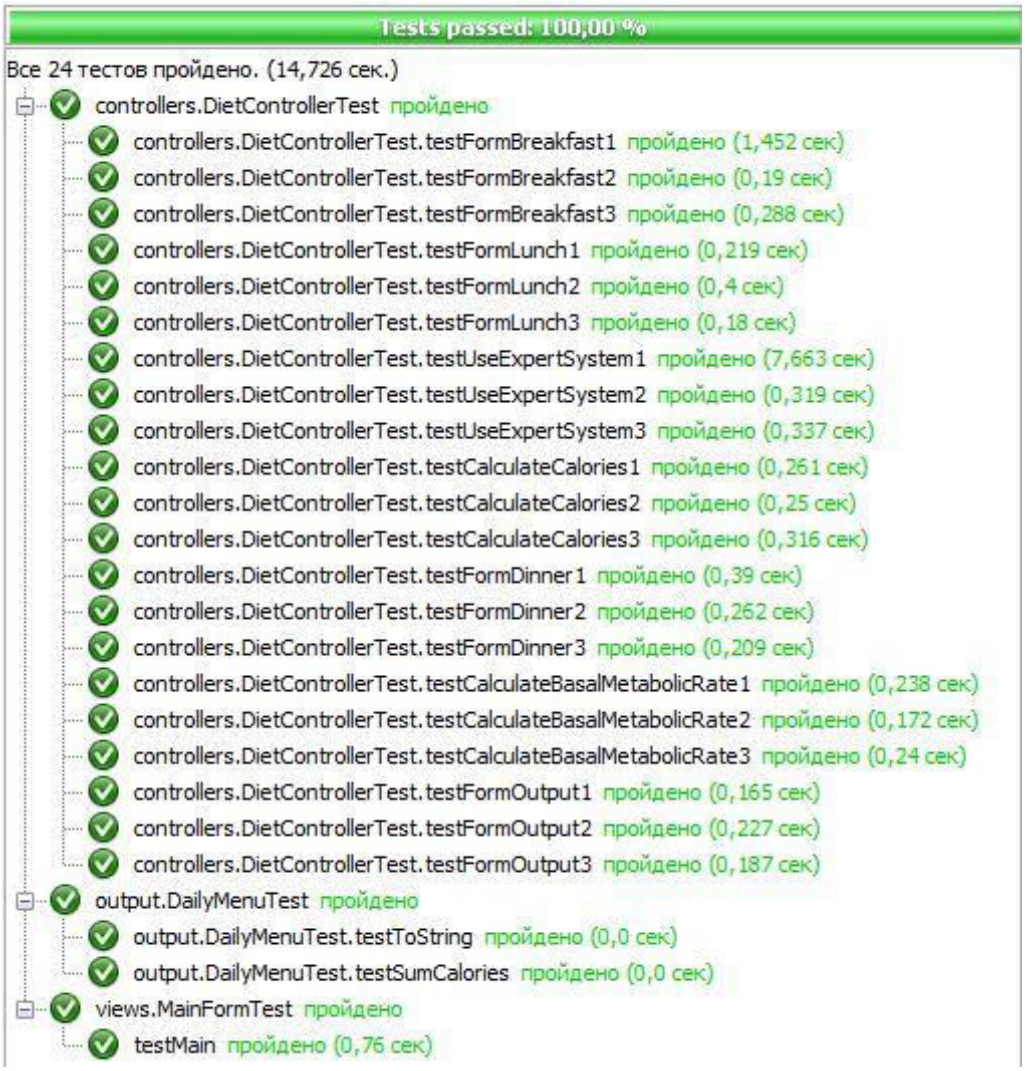


Рисунок 4.7 – Результати тестів

Як видно на рисунку 4.7, всі 24 тести були пройдені успішно, а помилок у функціонуванні модулів системи виявлено не було. Водночас тестові методи, які було створено з метою перевірки належного функціонування засобів використання експертної системи, доступу до бази даних, завантаження фактів до робочої пам’яті, підрахунку добової норми калорій користувача, формування добового меню та інших, показали, що кожен компонент системи працює правильно, а всі помилки, яку можуть виникнути, обробляються належним чином.

Крім того, слід зазначити, що при тестуванні було виявлено невеликий недолік програми, а саме довгий запуск експертної системи при першому використанні у порівнянні з наступними, проте цей факт ніяким чином не впливає на якість формування раціонів.

4.3 Висновки

У цьому розділі дипломної роботи описано реалізацію програмної системи вибору оптимального раціону харчування, яка була детально спроектована та описана у третьому розділі. Розроблена програмна система реалізує спроектовану архітектуру для вирішення завдань, описаних і поставлених у першому розділі.

Після детального опису призначення і функціоналу всіх складових реалізованого проекту був проведений процес тестування, який дозволив переконатись у правильності роботи системи.

Створене програмне забезпечення відповідає визначеним у першому розділі функціональним та нефункціональним вимогам. В той же час система є досить простою та зручною у використанні, а також має достатню кількість функціоналу для отримання користувачем раціону харчування, який найкраще відповідає його потребам та вимогам.

Вдосконалення та подальший розвиток системи є можливим та перспективним, оскільки однією з переваг реалізованої методології є розширюваність бази знань за допомогою додання нових правил.

ВИСНОВКИ

Отже, на основі проведеного аналізу технічної літератури та здійсненого дослідження в ході роботи, можна зробити наступні висновки.

У першому розділі було здійснено опис та аналіз предметної області сфери застосування розроблюваного програмного забезпечення, проведено аналіз існуючих рішень та знайдено їх недоліки, на основі яких було сформовано вимоги і складено технічне завдання.

У другому розділі вдосконалено математичну модель алгоритму вибору оптимального раціону харчування, зокрема визначено його переваги та недоліки, запропоновано способи модернізації та усунення слабких сторін. Крім того, було вдосконалено процес вибору оптимального раціону харчування за допомогою використання модифікованого раніше алгоритму в поєднанні з експертною системою, а також покращено структуру експертної системи для формування оптимального раціону харчування за рахунок використання вдосконаленої продукційної моделі.

У третьому розділі спроектовано архітектуру системи, визначено вимоги до програмно-технічного забезпечення, описано рух та перетворення даних при роботі системи, представлено спроектовану структуру бази даних. Крім того, описано всі деталі інтерфейсу користувача, які мають бути реалізовані, та обґрунтовано вибір технологій, за допомогою яких виконана програмна реалізація.

У четвертому розділі детально описано структуру системи в цілому та її модулів зокрема, а також класів, з яких ці модулі складаються. Подано програмний код найважливіших модулів системи, зазначено інформацію, достатню для розуміння функцій ПЗ, описано всі компоненти інтерфейсу користувача, протестовано систему на основі створеного плану тестування. В кінці розділу зроблено висновки щодо роботи системи на основі результатів виконання цих тестів.

Таким чином, результатом виконання дипломної роботи є створена технологія розробки програмної системи вибору оптимального раціону

харчування із врахуванням індивідуальних особливостей користувача на основі смарт-технологій та розроблена відповідна програмна система. У майбутньому розроблену систему можна вдосконалити шляхом додання правил в базу знань.

Практичне значення роботи полягає у покращенні результатів розробок персональних раціонів харчування та спрощенні отримання якісних безкоштовних індивідуальних планів правильного харчування для широкого кола людей, що, у свою чергу, повинно допомогти вирішити загальну проблему збалансованого харчування.

Таким чином, мети роботи з удосконалення процесу вибору оптимального раціону харчування із врахуванням індивідуальних особливостей користувача за допомогою смарт-технологій досягнуто.

За темою і результатами дипломної роботи опубліковані тези доповіді на науково-практичній Інтернет-конференції [18] та стаття у фаховому Міжнародному науковому журналі [19].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Колбасина К. Ю. Социальные представления о здоровом питании: опыт эмпирического исследования / К. Ю. Колбасина. – М.: Вестник Балтийского федерального университета им. И Канта, – 2013. – 3 с.

2. Шаров С. В. Сучасний стан розвитку інтелектуальних інформаційних систем / С. В. Шаров // Вісник Чернігівського національного педагогічного університету. Серія: Педагогічні науки. – 2015. – №130.

3. Шаров С. В. Розробка інтелектуальної інформаційної системи для птахівництва / С. В. Шаров, Д. В. Лубко // Системи обробки інформації. – 2017. – Т. 4. – № 150. – С. 170-174.

4. Петровский К. С. Гигиена питания: Учебник для сан.-гиг. фак. мед. ин-тов. – 2-е изд., перераб. и доп. / К. С. Петровский. – М.: Медицина, – 1975. – 400 с.

5. Сусліков В. Л. Харчування і здоров'я / В. Л. Сусліков. – Чебоксари, 1990. – 48 с.

6. Башмаков А. И. Интеллектуальные информационные технологии: учеб. пособ. / А. И. Башмаков, И. А. Башмаков. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2005. – 304 с.

7. How Practical and Conventional Artificial Intelligence are different [Електронний ресурс] / Веб-сайт компанії New Generation Applications – Режим доступу: https://www.newgenapps.com/blog/how-practical-and-conventional-artificial-intelligence-are-different/

8. Джексон П. Введение в экспертные системы / П. Джексон. – М.: Вильямс, 2001. – 624 с.

9. Варшавский П. Р. Моделирование рассуждений на основе прецедентов в интеллектуальных системах поддержки принятия решений / П. Р. Варшавский, А. П. Еремеев // Искусственный интеллект и принятие решений. 2009. – № 2. – С. 45-57.

10. Murphy K. An introduction to graphical models / K. Murphy // Rap. tech. – 2001. – V. 96. – P. 1-19.

11. Sun L. Using Bayesian networks for bankruptcy prediction: Some methodological issues [Text] / Lili Sun, Prakash P. Shenoy // European Journal of Operational research. – 2007. – № 180. – P. 738–753.

12. Круглов В. В.Искусственные нейронные сети. Теория и практика / В. В. Круглов, В. В. Борисов. – 2-е изд., стереотип. – М.: Горячая линия-Телеком, 2002. – 382 с.

13. Субботін С. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: монографія / С. О. Субботін, А. О. Олійник, О. О. Олійник; під заг. ред. С. О. Субботіна. – Запоріжжя: ЗНТУ, 2009. –375 с.

14. Глазкова И. В. Оптимизация рационов питания с использованием компьютерных технологий / И. В. Глазкова, Ю. А. Ивашкин // Пищевая промышленность. – 2010. – №6. – С.61-63.

15. Mifflin M. D. A new predictive equation for resting energy expenditure in healthy individuals / M. D. Mifflin // The American journal of clinical nutrition. – 1990. – V. 51. – №. 2. – P. 241-247.

16. Кудряшова Т. И. Применение современных методов расчета калорийности питания для оценки рационов студенческой молодежи / Т. И. Кудряшова, Л. М. Минасян, А. Э. Акайзина // Вопросы питания. – 2018. – С. 415-417.

17. Суботін С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: [навч. посібн.] / С. О. Суботін. – Запоріжжя: ЗНТУ, 2008. – 341 с.

18. Прейзнер Є. Е. Продукційна модель експертної системи для вибору оптимального раціону харчування / Є. Е. Прейзнер, О. М. Яшина // Матеріали науково-практичної Інтернет-конференції молодих науковців і студентів «Інтелектуальний потенціал – 2020». – 2020. – №2. – С. 97-100.

19. Прейзнер Є. Е. Методи штучного інтелекту в сфері охорони здоров’я / Є. Е. Прейзнер, О. М. Яшина // Вимірювальна та обчислювальна техніка в технологічних процесах. – Хмельницький, 2020. – №1. – С. 84–87.

ДОДАТОК А

(обов’язковий)

ДІАГРАМА ЗАГАЛЬНОЇ СТРУКТУРИ ЕКСПЕРТНОЇ СИСТЕМИ

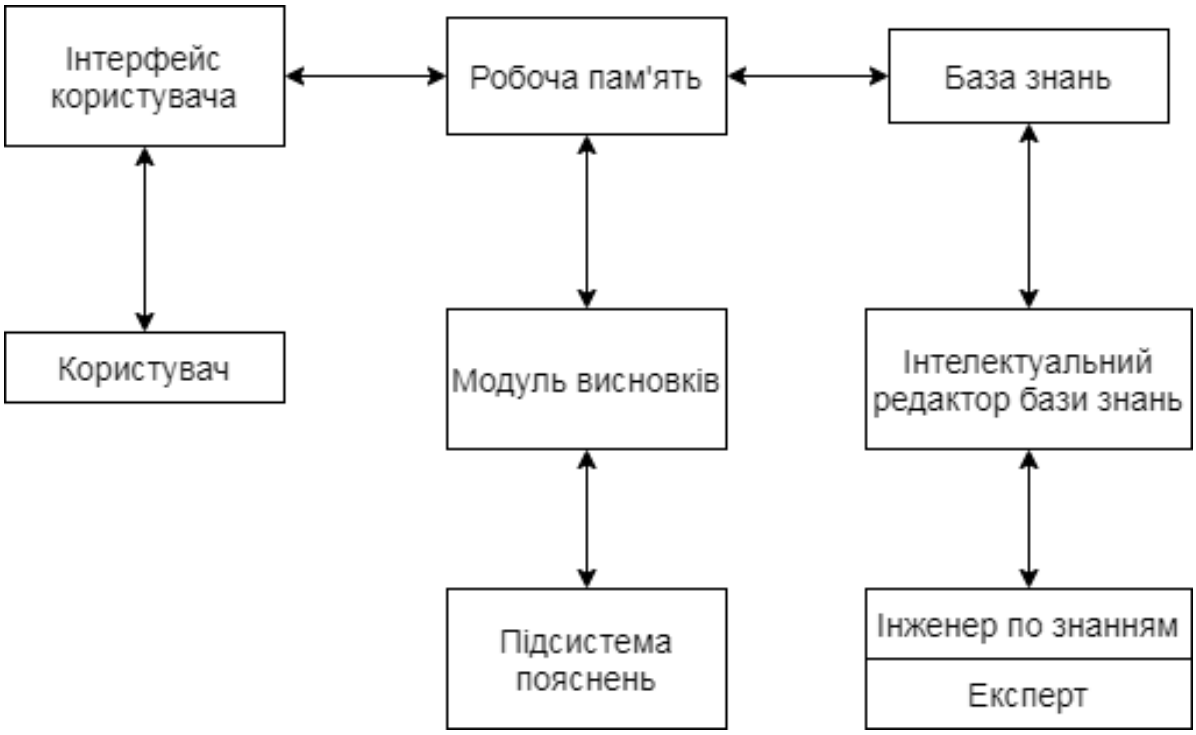


Рисунок А.1 – Загальна структура експертної системи

ДОДАТОК Б

(обов’язковий)

ДІАГРАМА ВДОСКОНАЛЕНОЇ СТРУКТУРИ ЕКСПЕРТНОЇ СИСТЕМИ

Рисунок Б.1 – Вдосконалена структура експертної системи

ДОДАТОК В

(обов’язковий)

ДІАГРАМА КЛАСІВ

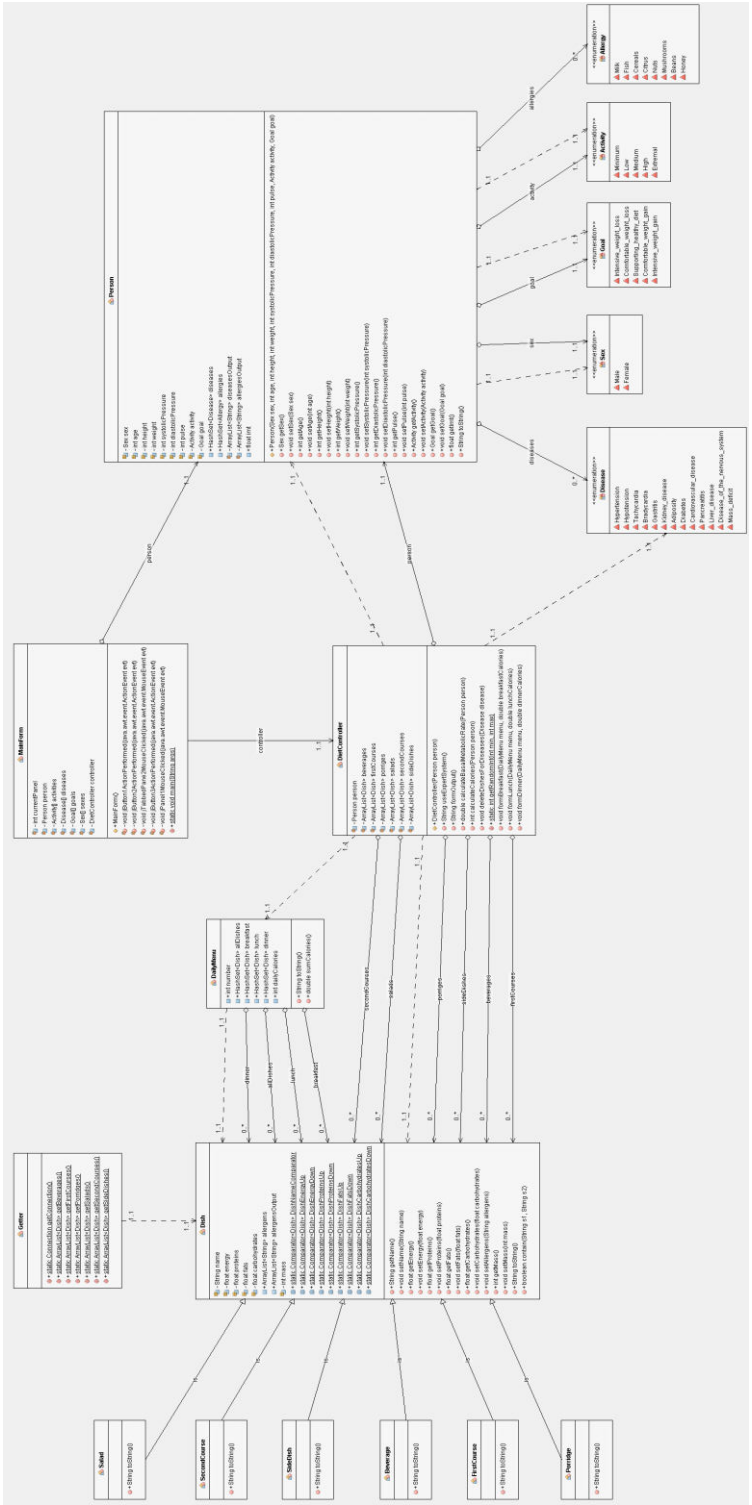


Рисунок В.1 – Повна діаграма класів

ДОДАТОК Г

(обов'язковий)

ПРОГРАМНИЙ КОД ОСНОВНИХ МОДУЛІВ

Г.1 Програмний код модуля DietController

```
package controllers;

import output.DailyMenu;
import java.util.ArrayList;
import person_models.*;
import db_models.*;
import java.sql.SQLException;
import java.util.HashSet;
import java.util.Iterator;
import java.util.Random;
import org.kie.api.KieServices;
import org.kie.api.runtime.KieContainer;
import org.kie.api.runtime.KieSession;

/**
 *
 * @author Евгений
 */
public class DietController {

    private Person person;
    private ArrayList<Dish> beverages;
    private ArrayList<Dish> firstCourses;
    private ArrayList<Dish> porriges;
    private ArrayList<Dish> salads;
    private ArrayList<Dish> secondCourses;
    private ArrayList<Dish> sideDishes;

    public DietController(Person person) throws SQLException {
        this.beverages = Getter.getBeverages();
        this.firstCourses = Getter.getFirstCourses();
        this.porriges = Getter.getPorridges();
        this.salads = Getter.getSalads();
        this.secondCourses = Getter.getSecondCourses();
        this.sideDishes = Getter.getSideDishes();
        this.person = person;
    }

    public String useExpertSystem() {
        try {
            KieServices ks = KieServices.Factory.get();
            KieContainer kContainer = ks.getKieClasspathContainer();
            KieSession kSession = kContainer.newKieSession("ksession-rules");
            kSession.insert(person);
            kSession.setGlobal("allBeverages", beverages);
            kSession.setGlobal("allFirstCourses", firstCourses);
            kSession.setGlobal("allPorriges", porriges);
            kSession.setGlobal("allSalads", salads);
            kSession.setGlobal("allSecondCourses", secondCourses);
```

```
            kSession.setGlobal("allSideDishes", sideDishes);
            kSession.fireAllRules();
            beverages = (ArrayList) kSession.getGlobal("allBeverages");
            firstCourses = (ArrayList) kSession.getGlobal("allFirstCourses");
            porriges = (ArrayList) kSession.getGlobal("allPorriges");
            salads = (ArrayList) kSession.getGlobal("allSalads");
            secondCourses = (ArrayList) kSession.getGlobal("allSecondCourses");
            sideDishes = (ArrayList) kSession.getGlobal("allSideDishes");
            kSession.dispose();
            return formOutput();
        } catch (Throwable t) {
            t.printStackTrace();
        }
        return "Експертна система не запустилась";
    }

    public String formOutput() {
        String output = "";
        int personCalories = (int) calculateCalories(person);
        output += "Добова норма калорій: " + personCalories + "\n";
        for (int i = 0; i < 7; i++) {
            DailyMenu day = new DailyMenu();
            day.number = i + 1;
            day.dailyCalories = personCalories;
            formBreakfast(day, (personCalories / 3));
            formLunch(day, (personCalories / 3));
            formDinner(day, (personCalories / 3));
            output = output + day.toString() + "\n";
        }
        return output;
    }

    public double calculateBasalMetabolicRate(Person person) {
        if (person.getSex() == Sex.Male) {
            if (null != person.getActivity()) {
                switch (person.getActivity()) {
                    case Minimum:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5)
                                * 1.2;
                    case Low:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5)
                                * 1.35;
                    case Medium:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5)
                                * 1.55;
                    case High:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5) * 1.75;
                    case Extremal:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5) * 1.95;
                    default:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) + 5) * 1.55;
                }
```

```
            } else {
                return ((9.99 * person.getWeight())
                        + (6.25 * person.getHeight())
                        - (4.92 * person.getAge()) - 161) * 1.55;
            }
        } else {
            if (null != person.getActivity()) {
                switch (person.getActivity()) {
                    case Minimum:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.2;
                    case Low:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.35;
                    case Medium:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.55;
                    case High:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.75;
                    case Extremal:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.95;
                    default:
                        return ((9.99 * person.getWeight())
                                + (6.25 * person.getHeight())
                                - (4.92 * person.getAge()) - 161) * 1.55;
                }
            } else {
                return ((9.99 * person.getWeight())
                        + (6.25 * person.getHeight())
                        - (4.92 * person.getAge()) - 161) * 1.55;
            }
        }
    }

    public int calculateCalories(Person person) {
        if (null == person.getGoal()) {
            return (int) calculateBasalMetabolicRate(person);
        } else {
            switch (person.getGoal()) {
                case Intensive_weight_loss:
                    return (int) ((int) calculateBasalMetabolicRate(person) *
0.8);
                case Comfortable_weight_loss:
                    return (int) ((int) calculateBasalMetabolicRate(person) *
0.9);
                case Supporting_healthy_diet:
                    return (int) calculateBasalMetabolicRate(person);
                case Comfortable_weight_gain:
                    return (int) ((int) calculateBasalMetabolicRate(person) *
1.1);
                case Intensive_weight_gain:
                    return (int) ((int) calculateBasalMetabolicRate(person) *
1.2);
                default:
                    return (int) calculateBasalMetabolicRate(person);
            }
        }
```

```
    }

    public static int getRandomInt(int min, int max) {
        Random randomGenerator = new Random();
        int index = randomGenerator.nextInt(max);
        return index;
    }

    public void formBreakfast(DailyMenu menu, double breakfastCalories) {
        int firstCourse_rnd = getRandomInt(0, firstCourses.size() - 1);
        int secondCourse_rnd = getRandomInt(0, secondCourses.size() - 1);
        int beverage_rnd = getRandomInt(0, beverages.size() - 1);
        int firstCourse_mass;
        int secondCourse_mass;
        int beverage_mass;
        while (menu.breakfast.size() < 3) {
            if (!menu.allDishes.contains(firstCourses.get(firstCourse_rnd))

&& !menu.allDishes.contains(secondCourses.get(secondCourse_rnd))
                    && !menu.allDishes.contains(beverages.get(beverage_rnd))) {
                firstCourse_mass = (int) ((breakfastCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.breakfast.add(dish1);
                secondCourse_mass = (int) ((breakfastCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.breakfast.add(dish2);
                beverage_mass = (int) ((breakfastCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.breakfast.add(dish3);
                menu.allDishes.add(dish1);
                menu.allDishes.add(dish2);
                menu.allDishes.add(dish3);
            } else {
                firstCourse_rnd = getRandomInt(0, firstCourses.size() - 1);
                firstCourse_mass = (int) ((breakfastCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.breakfast.add(dish1);
                menu.allDishes.add(dish1);
                secondCourse_rnd = getRandomInt(0, secondCourses.size() - 1);
                secondCourse_mass = (int) ((breakfastCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.breakfast.add(dish2);
                menu.allDishes.add(dish2);
                beverage_rnd = getRandomInt(0, beverages.size() - 1);
                beverage_mass = (int) ((breakfastCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.breakfast.add(dish3);
                menu.allDishes.add(dish3);
            }
        }
    }
```

```
    public void formLunch(DailyMenu menu, double lunchCalories) {
        int firstCourse_rnd = getRandomInt(0, firstCourses.size());
        int secondCourse_rnd = getRandomInt(0, secondCourses.size());
        int beverage_rnd = getRandomInt(0, beverages.size());
        int firstCourse_mass;
        int secondCourse_mass;
        int beverage_mass;
        while (menu.lunch.size() < 3) {
            if (!menu.allDishes.contains(firstCourses.get(firstCourse_rnd))

&& !menu.allDishes.contains(secondCourses.get(secondCourse_rnd))
                    && !menu.allDishes.contains(beverages.get(beverage_rnd))) {
                firstCourse_mass = (int) ((lunchCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.lunch.add(dish1);
                secondCourse_mass = (int) ((lunchCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.lunch.add(dish2);
                beverage_mass = (int) ((lunchCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.lunch.add(dish3);
                menu.allDishes.add(dish1);
                menu.allDishes.add(dish2);
                menu.allDishes.add(dish3);
            } else {
                firstCourse_rnd = getRandomInt(0, firstCourses.size());
                firstCourse_mass = (int) ((lunchCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.lunch.add(dish1);
                menu.allDishes.add(dish1);
                secondCourse_rnd = getRandomInt(0, secondCourses.size());
                secondCourse_mass = (int) ((lunchCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.lunch.add(dish2);
                menu.allDishes.add(dish2);
                beverage_rnd = getRandomInt(0, beverages.size());
                beverage_mass = (int) ((lunchCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.lunch.add(dish3);
                menu.allDishes.add(dish3);
            }
        }
    }

    public void formDinner(DailyMenu menu, double dinnerCalories) {
        int firstCourse_rnd = getRandomInt(0, firstCourses.size());
        int secondCourse_rnd = getRandomInt(0, secondCourses.size());
        int beverage_rnd = getRandomInt(0, beverages.size());
        int firstCourse_mass;
        int secondCourse_mass;
        int beverage_mass;
        while (menu.dinner.size() < 3) {
```

```
            if (!menu.allDishes.contains(firstCourses.get(firstCourse_rnd))

&& !menu.allDishes.contains(secondCourses.get(secondCourse_rnd))
                    && !menu.allDishes.contains(beverages.get(beverage_rnd))) {
                firstCourse_mass = (int) ((dinnerCalories * 0.35) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.dinner.add(dish1);
                secondCourse_mass = (int) ((dinnerCalories * 0.57) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.dinner.add(dish2);
                beverage_mass = (int) ((dinnerCalories * 0.08) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.dinner.add(dish3);
                menu.allDishes.add(dish1);
                menu.allDishes.add(dish2);
                menu.allDishes.add(dish3);
            } else {
                firstCourse_rnd = getRandomInt(0, firstCourses.size());
                firstCourse_mass = (int) ((dinnerCalories * 0.4) /
firstCourses.get(firstCourse_rnd).getEnergy() * 100);
                Dish dish1 = firstCourses.get(firstCourse_rnd);
                dish1.setMass(firstCourse_mass);
                menu.dinner.add(dish1);
                menu.allDishes.add(dish1);
                secondCourse_rnd = getRandomInt(0, secondCourses.size());
                secondCourse_mass = (int) ((dinnerCalories * 0.5) /
secondCourses.get(secondCourse_rnd).getEnergy() * 100);
                Dish dish2 = secondCourses.get(secondCourse_rnd);
                dish2.setMass(secondCourse_mass);
                menu.dinner.add(dish2);
                menu.allDishes.add(dish2);
                beverage_rnd = getRandomInt(0, beverages.size());
                beverage_mass = (int) ((dinnerCalories * 0.1) /
beverages.get(beverage_rnd).getEnergy() * 100);
                Dish dish3 = beverages.get(beverage_rnd);
                dish3.setMass(beverage_mass);
                menu.dinner.add(dish3);
                menu.allDishes.add(dish3);
            }
        }
    }
}
```

Г.2 Програмний код модуля Getter

```
package db_models;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Collections;
```

```
import javax.swing.JOptionPane;

/**
 *
 * @author Евгений
 */
public class Getter {

    public static Connection getConnection() {
        Connection con;
        try {
            con                                                                    =
DriverManager.getConnection("jdbc:mysql://localhost:3306/diet_db?useSSL=false",
                    "root", "root");
            return con;
        } catch (Exception e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Не вдалось підключитись до бази
даних");
            return null;
        }
    }

    public static ArrayList<Dish> getBeverages() throws SQLException {
        Connection connection = getConnection();
        String query = "select * from beverage";
        Statement st = null;
        ResultSet rs;
        ArrayList<Dish> list = new ArrayList<>();
        try {
            st = connection.createStatement();
            rs = st.executeQuery(query);
            Dish dish;
            while (rs.next()) {
                dish = new Beverage();
                dish.setName(rs.getString("Name"));
                dish.setEnergy((float) rs.getDouble("Energy"));
                dish.setProteins((float) rs.getDouble("Proteins"));
                dish.setFats((float) rs.getDouble("Fats"));
                dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
                if (rs.getString("Allergens") != null) {
                    dish.setAllergens(rs.getString("Allergens"));
                } else {
                    dish.setAllergens("Немає");
                }
                list.add(dish);
            }
            st.close();
            st = null;
            connection.close();
            connection = null;
        } finally {
            if (st != null) {
                try {
                    st.close();
                } catch (Exception sqlex) {
                }
                st = null;
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (Exception sqlex) {
                }
```

```
            connection = null;
        }
    }
    Collections.sort(list, Dish.DishNameComparator);
    return list;
}
public static ArrayList<Dish> getFirstCourses() throws SQLException {
    Connection connection = getConnection();
    String query = "select * from first_course";
    Statement st = null;
    ResultSet rs;
    ArrayList<Dish> list = new ArrayList<>();
    try {
        st = connection.createStatement();
        rs = st.executeQuery(query);
        Dish dish;
        while (rs.next()) {
            dish = new FirstCourse();
            dish.setName(rs.getString("Name"));
            dish.setEnergy((float) rs.getDouble("Energy"));
            dish.setProteins((float) rs.getDouble("Proteins"));
            dish.setFats((float) rs.getDouble("Fats"));
            dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
            if (rs.getString("Allergens") != null) {
                dish.setAllergens(rs.getString("Allergens"));
            } else {
                dish.setAllergens("Немає");
            }
            list.add(dish);
        }
        st.close();
        st = null;
        connection.close();
        connection = null;
    } finally {
        if (st != null) {
            try {
                st.close();
            } catch (Exception sqlex) {
            }
            st = null;
        }
        if (connection != null) {
            try {
                connection.close();
            } catch (Exception sqlex) {
            }
            connection = null;
        }
    }
    Collections.sort(list, Dish.DishNameComparator);
    return list;
}

public static ArrayList<Dish> getPorridges() throws SQLException {
    Connection connection = getConnection();
    String query = "select * from porridge";
    Statement st = null;
    ResultSet rs;
    ArrayList<Dish> list = new ArrayList<>();
    try {
        st = connection.createStatement();
        rs = st.executeQuery(query);
        Dish dish;
```

```
            while (rs.next()) {
                dish = new Porridge();
                dish.setName(rs.getString("Name"));
                dish.setEnergy((float) rs.getDouble("Energy"));
                dish.setProteins((float) rs.getDouble("Proteins"));
                dish.setFats((float) rs.getDouble("Fats"));
                dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
                if (rs.getString("Allergens") != null) {
                    dish.setAllergens(rs.getString("Allergens"));
                } else {
                    dish.setAllergens("Немає");
                }
                list.add(dish);
            }
            st.close();
            st = null;
            connection.close();
            connection = null;
        } finally {
            if (st != null) {
                try {
                    st.close();
                } catch (Exception sqlex) {
                }
                st = null;
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (Exception sqlex) {
                }
                connection = null;
            }
        }
        Collections.sort(list, Dish.DishNameComparator);
        return list;
    }

    public static ArrayList<Dish> getSalads() throws SQLException {
        Connection connection = getConnection();
        String query = "select * from salad";
        Statement st = null;
        ResultSet rs;
        ArrayList<Dish> list = new ArrayList<>();
        try {
            st = connection.createStatement();
            rs = st.executeQuery(query);
            Dish dish;
            while (rs.next()) {
                dish = new Salad();
                dish.setName(rs.getString("Name"));
                dish.setEnergy((float) rs.getDouble("Energy"));
                dish.setProteins((float) rs.getDouble("Proteins"));
                dish.setFats((float) rs.getDouble("Fats"));
                dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
                if (rs.getString("Allergens") != null) {
                    dish.setAllergens(rs.getString("Allergens"));
                } else {
                    dish.setAllergens("Немає");
                }
                list.add(dish);
            }
            st.close();
            st = null;
```

```
            connection.close();
            connection = null;
        } finally {
            if (st != null) {
                try {
                    st.close();
                } catch (Exception sqlex) {
                }
                st = null;
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (Exception sqlex) {
                }
                connection = null;
            }
        }
        Collections.sort(list, Dish.DishNameComparator);
        return list;
    }

    public static ArrayList<Dish> getSecondCourses() throws SQLException {
        Connection connection = getConnection();
        String query = "select * from second_course";
        Statement st = null;
        ResultSet rs;
        ArrayList<Dish> list = new ArrayList<>();
        try {
            st = connection.createStatement();
            rs = st.executeQuery(query);
            Dish dish;
            while (rs.next()) {
                dish = new SecondCourse();
                dish.setName(rs.getString("Name"));
                dish.setEnergy((float) rs.getDouble("Energy"));
                dish.setProteins((float) rs.getDouble("Proteins"));
                dish.setFats((float) rs.getDouble("Fats"));
                dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
                if (rs.getString("Allergens") != null) {
                    dish.setAllergens(rs.getString("Allergens"));
                } else {
                    dish.setAllergens("Немає");
                }
                list.add(dish);
            }
            st.close();
            st = null;
            connection.close();
            connection = null;
        } finally {
            if (st != null) {
                try {
                    st.close();
                } catch (Exception sqlex) {
                }
                st = null;
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (Exception sqlex) {
                }
                connection = null;
```

```
            }
        }
        Collections.sort(list, Dish.DishNameComparator);
        return list;
    }

    public static ArrayList<Dish> getSideDishes() throws SQLException {
        Connection connection = getConnection();
        String query = "select * from side_dish";
        Statement st = null;
        ResultSet rs;
        ArrayList<Dish> list = new ArrayList<>();
        try {
            st = connection.createStatement();
            rs = st.executeQuery(query);
            Dish dish;
            while (rs.next()) {
                dish = new SideDish();
                dish.setName(rs.getString("Name"));
                dish.setEnergy((float) rs.getDouble("Energy"));
                dish.setProteins((float) rs.getDouble("Proteins"));
                dish.setFats((float) rs.getDouble("Fats"));
                dish.setCarbohydrates((float) rs.getDouble("Carbohydrates"));
                if (rs.getString("Allergens") != null) {
                    dish.setAllergens(rs.getString("Allergens"));
                } else {
                    dish.setAllergens("Немає");
                }
                list.add(dish);
            }
            st.close();
            st = null;
            connection.close();
            connection = null;
        } finally {
            if (st != null) {
                try {
                    st.close();
                } catch (Exception sqlex) {
                }
                st = null;
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (Exception sqlex) {
                }
                connection = null;
            }
        }
        Collections.sort(list, Dish.DishNameComparator);
        return list;
    }
}
```

ДОДАТОК Д

(обов’язковий)

ІНСТРУКЦІЯ КОРИСТУВАЧА

Загальні відомості

Програма «Раціон харчування» створена, щоб допомогти користувачу покращити свій раціон, враховуючи індивідуальні особливості організму.

Установка і первинне налаштування.

Для запуску програми потрібно відкрити Diet.exe файл, надавши йому потрібні дозволи. Відкриття виконуваного файлу програми є стандартною процедурою для користувачів персональних комп’ютерів із операційною системою Windows та не потребує спеціальних знань чи вмінь.

Щоб використати програму користувачу необхідно знати свій зріст, вагу, а також показники артеріального тиску і пульсу в стані спокою. Крім того, він має визначитись з ціллю, яку йому необхідно досягти.

Основні поняття і визначення

В програмі використовуються такі медичні поняття, як систолічний та діастолічний тиск. Під систолічним тиском мається на увазі найбільше значення артеріального тиску при вимірі, а під діастолічним відповідно розуміється найменше значення.

Робота з програмою

Програма не має системи авторизації, тому одразу після запуску користувач побачить головне вікно програми з активною вкладкою введенням основних даних про стан здоров’я.

Далі користувачу необхідно ввести інформацію про своє здоров'я у відповідні поля на вкладці «Інформація про Вас» та натиснути кнопку «Далі».

Після натискання на кнопку, якщо введено всі необхідні дані і вони є коректними, відбудеться перехід на вкладку «Особливості організму».

На цій вкладці користувачу треба вказати чи має він перелічені алергії або захворювання і знову натиснути кнопку «Далі». Без виконання цього кроку користувач не зможе отримати результат виконання програми.

Якщо все зроблено вірно, через декілька секунд після натискання на кнопку перед користувачем у текстовій формі з'явиться сформований тижневий раціон, розроблений відповідно до особливостей його організму.

Повідомлення про помилки

При частковому заповненні полів на вкладці «Інформація про Вас» і натисканні на кнопку «Далі» користувач побачить повідомлення про помилку з текстом «Вкажіть всі дані». Відповідно при заповненні полів на цій вкладці некоректними даними користувач побачить повідомлення про помилку з текстом «Перевірте правильність введених даних».

Крім того, користувача також буде проінформовано, якщо в процесі формування меню виникне помилка або не вдасться підключитися до бази даних страв. Так, в першому випадку на вкладці «Результати», замість сформованого раціону, буде виведено «Експертна система не запустилась». Відповідно, при невдалому підключенні до бази даних користувач побачить повідомлення про помилку з текстом «Не вдалось підключитись до бази даних».

ДОДАТОК Е

(обов'язковий)

КОПІЇ НАУКОВИХ ПУБЛІКАЦІЙ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Хмельницький національний університет

Військовий інститут Київського національного університету ім.Тараса Шевченка

ПВНЗ "Університет економіки і підприємництва"

Вінницький національний технічний університет

Західноукраїнський національний університет

Інтелектуальний потенціал - 2020

збірник наукових праць молодих науковців і студентів

сформовано за матеріалами
Всеукраїнської науково-практичної конференції
молодих науковців і студентів
«Інтелектуальний потенціал – 2020»

9-10 листопада 2020 р.

Частина 1

Хмельницький
2020

ББК 74.480.278
С.88

«Інтелектуальний потенціал – 2020» - збірник наукових праць молодих науковців і студентів / Колектив авторів – Хмельницький: ПВНЗ УЕП, 2020. – Частина 1. – 104 с.

Відповідальний редактор: *Желавська Н.В.*

Відповідальний за випуск: *Чешун В.М.*

Редакційна колегія*:*
Желавський О.Б.
Кльоц Ю.П.
Чешун В.М.
Тімофєєва Л.В.

ЗМІСТ

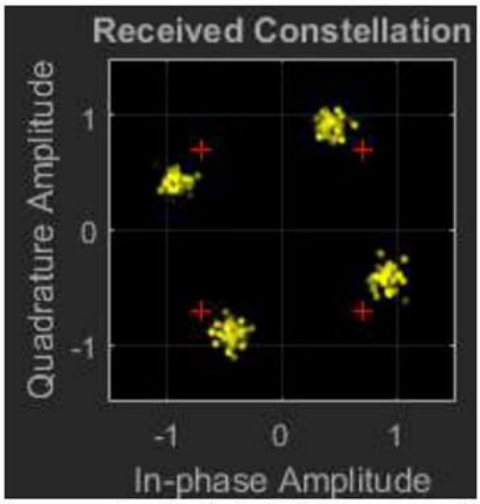


Рисунок 2 – Сузір'я QPSK модуляції, обумовлене зміщенням фази

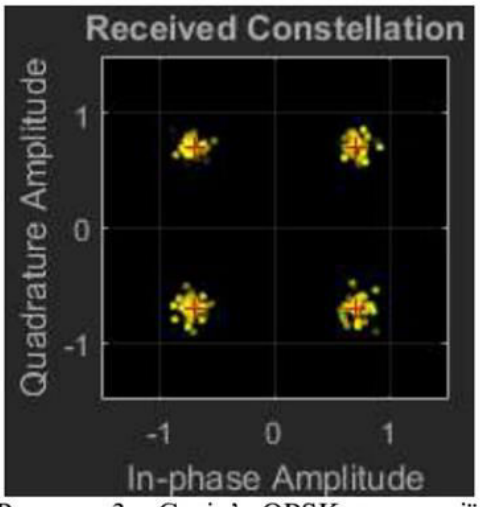


Рисунок 3 – Сузір'я QPSK модуляції, обумовлене зміщенням частоти

Енергетична ефективність не когерентної цифрової системи зв'язку з модуляцією QPSK при виникненні фазового зміщення 30^0 у середовищі розповсюдження зменшується на 10 дБ. Додавання кола символьної синхронізації компенсує цей поворот сигнального сузір'я. Збільшення фазового зміщення у середовищі розповсюдження до 45^0 для когерентної системи зв'язку приводить до зменшення енергетичної ефективності на 2 дБ.

Перелік посилань

1. Скляр Б. Цифровая связь. Теоретические основы и практическое применение/ Б. Скляр. Изд. 2-е, испр.: Пер. с англ. – М. : Издательский дом «Вильямс», 2003. – 1104 с.

2. Прокис Д. Цифровая связь / Д. Прокис. Пер. с англ. под ред. Д.Д. Кловского. – М.: Радио и связь, 2000. – 800 с.

3. Rice M. Digital communications: a discrete-time approach. – New Jersey: "Pearson Education, Inc.", 2009. – 778 p.

Продукційна модель експертної системи для вибору оптимального раціону харчування.

Яшина О.М., Прейзнер Є.Е.
Хмельницький національний університет

В умовах сьогодення питання збалансованого харчування стоїть в центрі уваги сучасної медицини та включене до списку найважливіших проблем, що розглядаються Всесвітньою організацією охорони здоров'я.

Швидке збільшення чисельності населення нашої планети призвело до відповідного зростання виробництв харчових ресурсів і продуктів харчування. Різноманіття продуктів, в свою чергу, ставить питання культури харчування і розумного споживання їжі для збереження здоров'я. У раціоні сучасної людини все частіше зустрічається висококалорійна їжа: кондитерські вироби, борошняні вироби, жирне м'ясо, алкоголь. Важливо відзначити, що незбалансоване, висококалорійне харчування і переїдання є основними причинами розвитку надмірної ваги і ожиріння, що в свою чергу призводить до серцево-судинних захворювань, діабету та інших важких хвороб. З іншого боку голодування та недоїдання з метою схуднення можуть призвести до появи не менш серйозних захворювань.

Метою даної роботи є створення програмної системи для формування рекомендацій по оптимізації раціону харчування для користувача на основі аналізу інформації про його стан здоров’я та цілей, які він хоче досягти.

Аналіз літературних джерел дозволив встановити, що в сфері штучного інтелекту для вирішення поставлених задач найбільше підходить методологія експертних систем.

До основних переваг експертних систем відносять:

–збереження знань системи протягом довгого часу;

–легкість передачі або відтворення;

–можливість одержання й об’єднання експертних знань з багатьох джерел;

–формалізація і перевірка знань;

–можливість застосування у якості інтелектуальної бази даних;

–об'єктивний результат за будь-яких обставин;

–можливість пояснення рішень.

Таким чином, в розроблюваному програмному комплексі експертна система буде використовуватись для раціонального вибору продуктів та страв на основі властивостей продуктів і раціону в цілому за критеріями харчової та біологічної цінності в залежності від стану здоров’я людини (зріст, вага тіла, стать, наявність алергій та хвороб), її щоденної активності та цілей, які вона хоче досягти.

Для вирішення наведених задач буде використовуватись продукційна модель експертної системи. Під даною моделлю мається на увазі система, заснована на продукційних правилах. Кожне з таких правил містить дві частини: умова та дія.

Загальна стратегія вирішення проблем в продукційній моделі полягає в розбитті завдань на фрагменти, які можна легше довести. При цьому, існують 2 підходи до вирішення завдань:

–прямий логічним висновок;

–зворотній логічний висновок.

Системи з прямим логічним висновком знаходяться під управлінням фактів. Вони починають свою роботу з відомих початкових фактів і продовжують, використовуючи правила для створення висновків або виконання певних дій. Подібна логіка вирішення проблем ідеально підходить для нашого випадку, тому системи зі зворотнім логічним висновком розглядати не доцільно.

Таким чином при побудові програмного забезпечення буде вперше удосконалено процес вибору оптимального раціону харчування за допомогою засобів штучного інтелекту.

Крім того, після використання експертної системи потрібно ще підрахувати розміри порцій добового раціону, враховуючи не повторюваність цих страв, їх поживну цінність та кількість мінералів. Для досягнення цього буде використовуватись алгоритм формування індивідуального меню с урахуванням індивідуальних особливостей людини та математична модель, яку можна описати наступним чином:

$$\mu_k = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \tag{2.1}$$

де μ_k- змінна включення до складу раціону при наступних обмеженнях:

1) по не повторюваності страв в добовому раціоні

$$\mu_k = 1\,;\ k \in\ k_a, k_b, k_c \tag{2.2}$$

2) на граничну калорійність одного прийому їжі

$$\sum_k \mu_k Q_k \le Q_{доб}\eta_l \tag{2.3}$$

де Q_k - нормативна калорійність k-тої страви, ккал; $Q_{доб}$ - гранична калорійність добового раціону, ккал; η_l - частка добової калорійності, яка припадає на сніданок, обід і вечерю, таким чином, що

$$\sum_l \eta_l = 1 \tag{2.4}$$

3) на граничний обсяг сніданку, обіду та вечері

$$\sum_k \mu_k V_k \le V_{доб}\omega_l \tag{2.5}$$

де $V_{доб}$ - граничний добовий обсяг раціону, г; V_k - об'єм k-тої страви, г; ω_l - частка добового обсягу, що припадає на сніданок, обід і вечерю, таким чином, що

$$\sum_{l} \omega_l = 1 \tag{2.6}$$

4) по верхньому і нижньому кордонах змісту і-го хімічного елемента

$$B_i^{min} \le \mu_k \sum_{J} \alpha_{ij} Z_{jk} \le B_i^{max} \tag{2.7}$$

де B_i^{min} та B_i^{max} - відповідно нижні і верхні обмеження на вміст в добовому раціоні білків, жирів та вуглеводів, мг; α_{ij} - зміст і-го елемента хімічного складу в одиниці j-го продукту; Z_{jk} - рекомендований вміст j-го продукту в k-ій страві, г;

5) по масі k-ої страви, г

$$\sum_{J} Z_{jk} = Z_k \tag{2.8}$$

Таким чином на основі викладеного вище можна сформувати схему руху і перетворення даних при роботі системи (Рисунок 1).

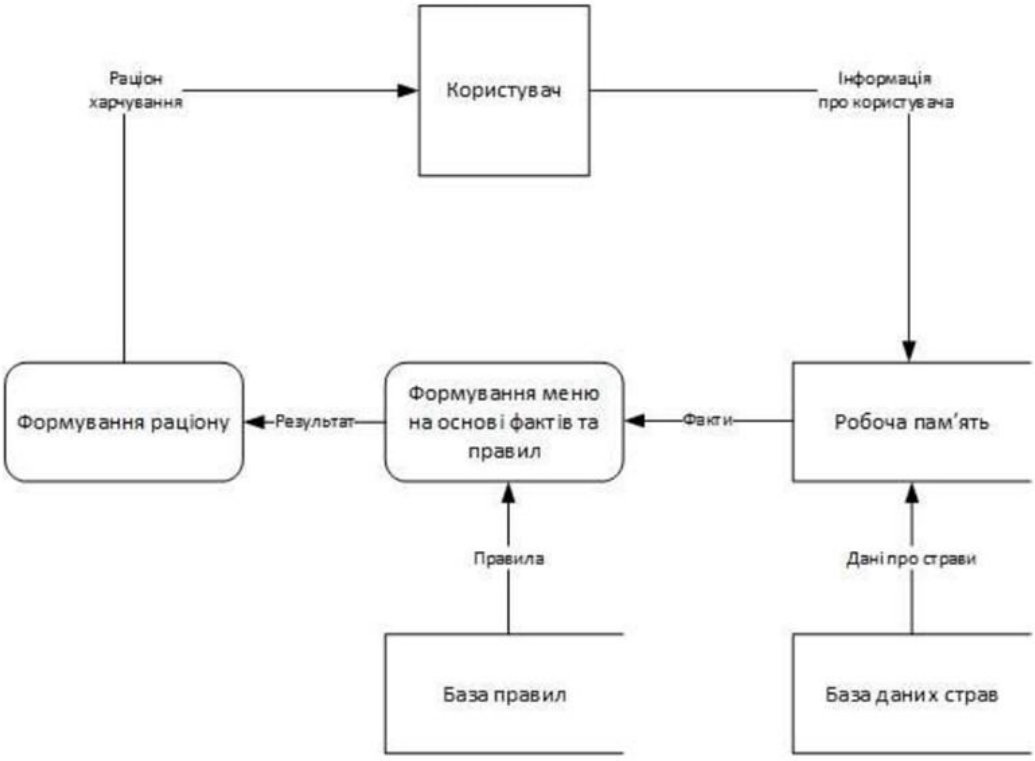


Рисунок 1 – Модель роботи програмної системи

Робота програмної системи складається з таких кроків:

1. Введення користувачем інформації про себе.
2. Завантаження даних про здоров'я користувача до робочої пам'яті.
3. Завантаження даних про страви до робочої пам'яті.
4. Завантаження фактів до експертної системи.
5. Завантаження правил до експертної системи.
6. Формування меню, яке найкраще підходить користувачу, на основі фактів та правил.
7. Формування раціону на основі результатів роботи експертної системи.
8. Виведення сформованого раціону у вікно інтерфейсу користувача.

Отже, в даній роботі вдосконалено продукційну модель для вибору оптимального раціону харчування. В процесі розробки, на основі ряду досліджень та публікацій було проаналізовано процес формування і оптимізації раціону харчування людини, пояснено що являє собою раціональне та нераціональне харчування і чим вони відрізняються. Використання розробленої продукційної моделі здатне зробити процес вибору раціону харчування більш ефективним. Крім того, створену модель можна покращити, додавши нові правила в базу знань.

Література

1. Сидоркина И.Г. Системы искусственного интеллекта: учебное пособие [текст] / И.Г Сидоркина// М.: КНОРУС. – 2011. – 248 с.

2. Глазкова И.В., Ивашкин Ю.А. Оптимизация рационов питания с использованием компьютерных технологий// Пищевая промышленность. – 2010. – №6. – С.61-63.

3. Герасименко Н.Ф. Здоровое питание и его роль в обеспечении качества жизни / Герасименко Н.Ф., Позняковский В.М., Челнакова Н.Г. // Технологии пищевой и перерабатывающей промышленности АПК - продукты здорового питания. – 2016. – № 4 (12). – С. 52-57.

4. Хох В.Д. Дослідження методів побудови експертних систем / В. Д. Хох, Є. В. Мелешко, М. С. Якименко // Системи управління, навігації та зв'язку. – 2016. – № 4 (10). – С. 48-52.

5. Сєдих О. Л. Дослідження методології побудови та принципів функціонування експертних систем / О. Л. Сєдих, В. О. Овчарук // Международное переодическое научное издание. Научные труды SWorld. – Иваново : Научный мир, 2016. – № 1 (42). – С. 13–19.

ISSN 2219-9365
DOI: 10.31891/2219-9365

Міжнародний науковий журнал

ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ

2020, № 1

International scientific-technical journal

MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES

2020, Issue 1

Хмельницький 2020
Khmelnytskyi 2020

МІЖНАРОДНИЙ НАУКОВО-ТЕХНІЧНИЙ ЖУРНАЛ
ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ

Затверджений як фахове видання (перереєстрація), група «Б»
Наказ МОН 28.12.2019 №1643

Засновано в травні 1997 р. ***Виходить 2 рази на рік***

Хмельницький, 2020, № 1 (65)

Засновник і видавець: **Хмельницький національний університет (до 2005 р. — Технологічний університет Поділля, м. Хмельницький)**

Наукова бібліотека України ім. В.І. Вернадського http://nbuv.gov.ua/j-tit/vott

Журнал включено до наукометричних баз:

РИНЦ	http://elibrary.ru/title_about.asp?id=37653
Index Copernicus	http://jml2012.indexcopernicus.com/,p24781565,3.html h-індекс 49,97
Google Schoolar	http://scholar.google.com.ua/citations?user=nwN_nusAAAAJ&hl=uk - індекс 9
Національна бібліотека України ім. В.І. Вернадського	http://nbuv.gov.ua/j-tit/vott

Рекомендовано до друку рішенням Вченої ради Хмельницького національного університету, протокол № 10 від 28.05.2020

Адреса редакції: **Україна, 29016, м. Хмельницький, вул. Інститутська, 11, Хмельницький національний університет редакція журналу "Вимірювальна та обчислювальна техніка в технологічних процесах"**

☎ **067-347-74-57**

e-mail: **mscientificjournal@gmail.com**

web: **http://journals.khnu.km.ua/index.php/MeasComp**

Зареєстровано Міністерством України у справах преси та інформації.
Свідоцтво про державну реєстрацію друкованого засобу масової інформації
Серія КВ № 23279-13119ПР від 24 травня 2018 року (перереєстрація)

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

ЗМІСТ

МЕТРОЛОГІЯ, СТАНДАРТИЗАЦІЯ, СЕРТИФІКАЦІЯ ТА ВИМІРЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ

ІНФОКОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ, АВТОМАТИЗАЦІЯ ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСА

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

УДК 004.9
DOI: 10.31891/2219-9365-2020-65-1-13

ПРЕЙЗНЕР Є. Е., ЯШИНА О. М.
Хмельницький національний університет

МЕТОДИ ШТУЧНОГО ІНТЕЛЕКТУ В СФЕРІ ОХОРОНИ ЗДОРОВ'Я

В даній статті описано та досліджено сучасний стан використання штучного інтелекту (ШІ) в сфері охорони здоров'я, його методів та засобів. Під час огляду обсягів використання та методів застосування ШІ в медицині в сучасному світі, також розглядалися питання про поточний стан розвитку сфери ШІ в цілому, її актуальність, перспективність та можливості на даному етапі розвитку.

Не зважаючи на велику кількість інвестицій в розробку ШІ для використання в медицині, на даний момент ефективність роботи всіх існуючих рішень недостатньо висока.

Ключові слова: штучний інтелект(ШІ). охорона здоров'я, ринок штучного інтелекту

PREIZNER E., YASHYNA O.
Khmelnytskyi National University

METHODS OF ARTIFICIAL INTELLIGENCE IN THE FIELD OF HEALTHCARE

The vital activity and adequate functioning of society directly depends on the preservation of the health of the population of any state. The most important area in this direction is health care, because health in general is a key aspect of the existence of any living organism, including humans. This means that the importance of health is difficult to overestimate. For this reason, countries around the world are making significant efforts to improve the quality of health care, using the latest advances in science and technology.

Such achievements undoubtedly include developments in the field of information technology (IT), as this industry in the modern world is one of the most promising in terms of the number of achieved results of human activity. Thus, today strong investment investments are made in the development of software for health care, because the development of high quality and effective health care is impossible without the use of modern means of medical information processing, implementation of intellectual management methodology and advanced communication technology.

This article describes and explores the current state of artificial intelligence (AI) in health care. During the review of the scope and methods of application of AI in medicine in the modern world, also considered the current state of development of AI in general, its relevance, prospects and opportunities at this stage of development.

Today all the artificial intelligence systems that exist are called weak AI, because it can't learn the way people do or think the way people do. It can do only one thing, such as searching the Internet for queries, recognizing objects in images, diagnosing a specific disease. Besides at the moment the efficiency of all existing solutions in the field of healthcare is not high enough. However the AI field is still growing, for example over the past four years, annual investment in artificial intelligence increases by more than $4.25 billion (total investment in 2014) and at the same time the field of Healthcare led AI investment with $4 billion investment in 2019.

Key words: artificial intelligence (AI). healthcare, artificial intelligence market.

Вступ. Життєдіяльність та адекватне функціонування суспільства напряму залежить від збереження здоров'я населення будь-якої держави. Найважливішою сферою в цьому напрямку є охорона здоров'я, адже здоров'я загалом є основним аспектом існування будь-якого живого організму, у тому числі людини. А це означає, що важливість здоров'я досить важко переоцінити. З цієї причини країни по всьому світу докладають чималих зусиль, щоб поліпшити якість охорони здоров'я, використовуючи для цього новітні досягнення науки і технологій.

До таких досягнень безперечно відносяться розробки у сфері інформаційних технологій (IT), оскільки дана галузь у сучасному світі є однією із найперспективніших за кількістю досягнутих результатів людської діяльності. Так, вже сьогодні в розробку програмних засобів для галузі охорони здоров'я здійснюються потужні інвестиційні вклади, адже розвиток високоякісної й ефективної охорони здоров'я неможливий без застосування сучасних засобів опрацювання медичної інформації, упровадження методології інтелектуального керування та високорозвиненої техніки зв'язку [1].

Методологічною основою для написання даної статті стали праці різних науковців як вітчизняних, так і зарубіжних. Зокрема Карпов О.Н., Бондаренко М.Ф. займаються мовними системами штучного інтелекту; Деркач Т.М. досліджує використання штучного інтелекту у сфері праці; доктринальний аналіз поняття штучного інтелекту здійснили Васильєв А.А., Шпоппер Д.; можливості та методи використання нейронних мереж досліджували Ларіонова А.В., Арутюнян В.Г.; Гусєв А.В. визначив перспективи використання нейронних мереж в системі охорони здоров'я.

Мета статті полягає у дослідженні та висвітленні сучасного стану використання штучного інтелекту в сфері охорони здоров'я.

Виклад основного матеріалу. В сучасних умовах термін «штучний інтелект» є досить розповсюдженим та широковживаним. Це пов'язано перш за все із сучасними тенденціями розвитку

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

інформаційних технологій (ІТ), адже сьогодні найважливішою сферою досліджень ІТ є саме штучний інтелект. Подідні напрями розвитку можна легко пояснити, якщо поглянути на переваги розробок в даній сфері і порівняти їх з іншими досягненнями. Так, на відміну від інших програмних засобів ШІ дає можливість ефективно вирішувати велике коло задач у різних сферах діяльності людини, зокрема економіці, науці, виробництві, охороні здоров'я, побуті, освіті, банківській сфері, торгівлі тощо. До таких задач відносять завдання класифікації, кластеризації, моделювання, прийняття рішень, задачі регресії для прогнозування, оцінки будь-якої цифрової інформації та ін.

Фактично в даний час на основі методів штучного інтелекту створюються і розвиваються різні програмні системи, головною особливістю яких є здатність вирішувати інтелектуальні завдання так, як це робила б людина, яка розмірковує над їх вирішенням [2].

Системи ШІ умовно можна поділити на два класи – сильний ШІ і слабкий ШІ. Сильний, або універсальний, штучний інтелект визначається як ШІ, який можна порівняти з людським інтелектом, тобто ШІ, який може вчитися так, як це роблять люди, і при цьому не поступається за рівнем розвитку більшості людей, а в багатьох сенсах навіть перевершує їх [3].

Так, всі інші системи, включаючи системи штучного інтелекту, які оточують нас зараз, називаються слабким ШІ, оскільки вони можуть робити тільки одну справу, наприклад здійснювати пошук по запитам в Інтернеті, розпізнавати об'єкти на зображеннях, ставити діагноз по конкретному захворюванню тощо. Використання такого ШІ робить життя комфортнішим, а роботу - більш продуктивною. Такі системи з кожним днем все більше вдосконалюються, але вже зараз системи з ШІ багато завдань роблять краще, ніж люди. Однак які б заняття не вміли робити подібні системи, окремо це всього лише потужні інструменти, що швидко розвиваються, рік від року просуваючи технології ШІ вперед і наближаючи створення сильного ШІ [3].

У зв'язку з еволюцією поняття ШІ необхідно також згадати про так званий ефект ШІ (AI Effect). Цей ефект полягає в тому, що кожен раз, коли ШІ справді досягає немислимого раніше результату спостерігачі знецінюють значимість цієї навички і перестають вважати її задачею ШІ.

Проте даний ефект ніяк не впливає на колосальний ріст світового ринку штучного інтелекту. Це чітко видно по обсягам інвестицій в компанії, що займаються розробкою ШІ (рис. 1). Так в кінці січня 2020 р. компанія CB Insights провела щорічний аналіз глобальних тенденцій інвестування в штучний інтелект і повідомила, що в 2019 році стартапи, які спеціалізуються на технологіях ШІ залучили рекордні інвестиції – $26,6 млрд, уклавши більше 2200 угод по всьому світу. Для порівняння, в 2018 році було укладено близько 1900 угод на загальну суму $22,1 млрд, а в 2017-му – близько 1700 угод на $16,8 млрд [4]. Тобто з 2016-го р. щорічно обсяг інвестицій збільшується більше, ніж на $4,25 млрд (загальний обсяг інвестицій в 2014 році).

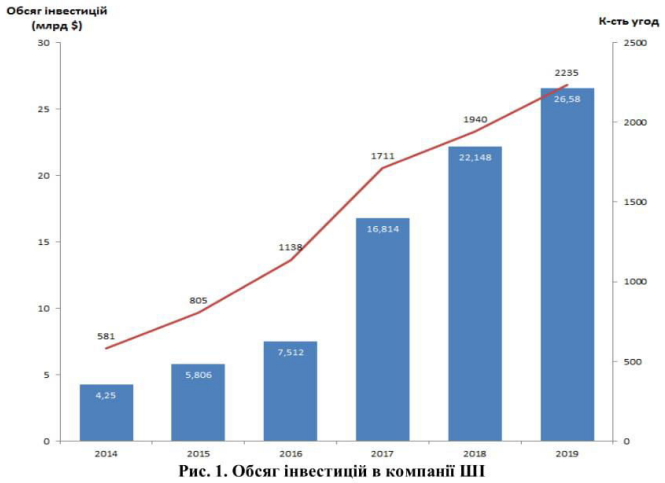


Рис. 1. Обсяг інвестицій в компанії ШІ

Якщо ж поглянути на дану ситуацію з боку інвестицій в розробку штучного інтелекту для певних галузей, то за підсумками 2019 року серед сфер інвестицій лідируючу позицію займала охорона здоров'я – до компаній, які займаються розробкою ШІ в даній галузі інвестори вклали $4 млрд. За нею слідують такі сфери, як фінанси ($2,2 млрд), роздрібна торгівля ($1,5 млрд), продажі та кібербезпека. Активність угод зі злиттів і поглинань також була найвищою в сфері охорони здоров'я, продажів і роздрібної торгівлі [5]. Крім того за прогнозами аналітиків IDC поширення коронавіруса COVID-19 призведе до вибухового зростання витрат на штучний інтелект в світі.

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

Подібний інтерес до розвитку ШІ в медицині цілком виправданий, адже дані технології допоможуть вирішувати широкий спектр різноманітних задач в різних галузях медицини, таких як:

– розпізнавання медичних зображень (знімків МРТ, висновків УЗД, кардіограм, результатів комп'ютерної томографії);

– розробка оптимального раціону харчування із врахуванням персональних особливостей організму;

– розробка лікарських препаратів (мікроскопічний аналіз, вивчення ефективності препаратів, дослідження вірусів і пошук ефективних вакцин);

– розробка зручних протезів з урахуванням анатомічних особливостей людини;

– віддалена допомога пацієнту;

– лікування ракових захворювань (аналіз клінічної картини стану пацієнта і надання ефективної схеми лікування).

Вже сьогодні найбільші IT-компанії світу, такі як IBM, Microsoft, Google, Intel та інші мають власні розробки на основі штучного інтелекту, які допомагають вирішувати подібні задачі. Так компанія IBM використовує свій суперкомп'ютер оснащений системою штучного інтелекту під назвою IBM Watson для визначення оптимальної, доказової, заснованої на даних стратегії лікування раку. Дана програма використання ресурсів суперкомп'ютера має назву IBM Watson for Oncology і з 2013 року використовується для допомоги в прийнятті управлінських рішень при лікуванні хворих на рак легень. Перед запуском цієї програми в Watson для навчання були завантажені сотні тисяч медичних документів, в тому числі 25 тисяч історій хвороб, понад 300 медичних журналів і понад 200 підручників, загальним обсягом близько 15 млн сторінок тексту, але незважаючи на це ефективність роботи даної програми знаходиться під сумнівом [6].

Програма IBM Watson for Oncology працює так: медпрацівник вводить в програму історію хвороби пацієнта, вона порівнює її зі своєю величезною бібліотекою наукових статей і клінічних рекомендацій і після натискання на кнопку «Запитати Ватсона» видає свої клінічні рекомендації на основі кращих сучасних клінічних практик, найсучасніших досліджень і інших досягнень доказової медицини [7]. Крім IBM Watson for Oncology компанія IBM має ще один проект, який використовує ШІ в сфері охорони здоров'я. Він має назву IBM Medical Sieve і зараз знаходиться на стадії розробки. Головною метою даного проекту є допомога лікарям у дослідженні медичних зображень (результати МРТ, рентген-знімків, кардіограм).

Ще одна програма для допомоги в лікуванні раку належить компанії Microsoft і має назву Hanover. Суть її роботи полягає в тому, щоб опрацювати всі документи про існуючі на даний момент лікарські засоби проти раку (сьогодні в світі розробляються сотні нових лікарських засобів проти раку, а нові дослідження публікуються щохвилини) та передбачити, які препарати та які комбінації є найбільш ефективними. Інакше кажучи, дана програма повинна допомогти лікарям лікувати пацієнтів за допомогою персоналізованих комбінацій, спрямованих на конкретні складові їх захворювання, з врахуванням останніх розроблених препаратів, адже жодному лікарю не уявляється можливим ознайомитись зі всіма роботами в даній галузі [8].

Холдинг Alphabet теж має проект, який застосовує технології ШІ в медицині, цей проект має назву DeepMind Health (раніше належала компанії Google і називалась Google DeepMind Health) та використовується в лондонських лікарнях для діагностики очних хвороб і для довідкової допомоги при лікуванні деяких онкологічних захворювань [7].

Висновки. На основі опрацювання спеціальної літератури в ході здійсненого дослідження можна зробити наступні висновки. На даному етапі розвитку штучний інтелект хоч і не може вчитися як людина, проте він все ж являється потужним інструментом для виконання одних задач та перспективним для інших Саме тому з кожним роком сфера штучного інтелекту та кількість компаній в ній значно зростає (рис. 1). Крім того сьогодні найбільш пріоритетною галуззю використання ШІ є сфера охорона здоров'я.

Найбільш оптимальні та вдосконалені рішення в даній сфері можуть створити тільки ті компанії, що мають доступ до надвеликих об'ємів даних. Це говорить про те, що відсутність великої кількості даних для навчання є однією з головних перешкод в розробці ШІ, що в свою чергу дещо гальмує розвиток відповідної сфери. Також варто зазначити, що існуючі на даний момент рішення в області охорони здоров'я тільки розвиваються і поки що не можуть значно покращити існуючі методи лікування або створити нові. Проте, вони здатні допомагати людині слідкувати за станом свого здоров'я, надавати рекомендації для покращення здоров'я і прискорювати розробку ліків. Крім того вони можуть оптимізувати деякі аспекти роботи лікаря та зробити високоякісну медицину більш доступною.

Література

1. Haux R. Health care in the information society: what should be the role of medical information? // Methods Inf. Med. – 2002. – № 41(1). – P. 31-35.

2. Гусев А. В. Перспективы нейронных сетей и глубокого машинного обучения в создании решений для здравоохранения // Врач и информационные технологии. – 2017. – № 3. – С. 92-105.

3. Пройдаков Э. М. Современное состояние искусственного интеллекта // Науковедческие исследования. 2018. № 2018. С. 129–153.

Міжнародний науково-технічний журнал
«ВИМІРЮВАЛЬНА ТА ОБЧИСЛЮВАЛЬНА ТЕХНІКА В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ»
ISSN 2219-9365

4. CB Insights: AI startup funding hit new high of $26.6 billion in 2019 [Електронний ресурс]. – Режим доступу: https://venturebeat.com/2020/01/22/cb-insights-ai-startup-funding-hit-new-high-of-26-6-billion-in-2019/
5. Investors poured $4B into healthcare AI startups in 2019 [Електронний ресурс]. – Режим доступу: https://www.fiercehealthcare.com/tech/investors-poured-4b-into-healthcare-ai-startups-2019
6. Искусственный интеллект в медицине [Електронний ресурс]. – Режим доступу: https://22century.ru/popular-science-publications/artificial-intelligence-in-medicine
7. Как доктор Ватсон не смог победить рак [Електронний ресурс]. – Режим доступу: https://medportal.ru/mednovosti/kak-doktor-vatson-ne-smog-pobedit-rak/
8. Artificial intelligence project slices cancer data [Електронний ресурс]. – Режим доступу: https://www.heraldtribune.com/news/20161004/artificial-intelligence-project-slices-cancer-data

References

1. Haux R. Health care in the information society: what should be the role of medical information? // Methods Inf. Med. – 2002. – № 41(1). – P. 31-35.
2. Gusev A. V. Perspektivy neyronnykh setey i glubokogo mashinnogo obucheniya v sozdanii resheniy dlya zdravookhraneniya // Vrach i informatsionnyye tekhnologii. – 2017. – № 3. – S. 92-105.
3. Proydakov E. M. Sovremennoye sostoyaniye iskusstvennogo intellekta // Naukovedcheskiye issledovaniya. 2018. № 2018. S. 129–153.
4. CB Insights: AI startup funding hit new high of $26.6 billion in 2019 [Elektronniy resurs]. – Rezhim dostupu: https://venturebeat.com/2020/01/22/cb-insights-ai-startup-funding-hit-new-high-of-26-6-billion-in-2019/
5. Investors poured $4B into healthcare AI startups in 2019 [Elektronniy resurs]. – Rezhim dostupu: https://www./fiercehealthcare.com/tech/investors-poured-4b-into-healthcare-ai-startups-2019
6. Iskusstvennyy intellekt v meditsine [Elektronniy resurs]. – Rezhim dostupu: https://22century.ru/popular-science-publications/artificial-intelligence-in-medicine
7. Kak doktor Vatson ne smog pobedit rak [Elektronniy resurs]. – Rezhim dostupu: https://medportal.ru/mednovosti/kak-doktor-vatson-ne-smog-pobedit-rak/
8. Artificial intelligence project slices cancer data [Elektronniy resurs]. – Rezhim dostupu: https://www.heraldtribune.com/news/20161004/artificial-intelligence-project-slices-cancer-data

Надійшла / Paper received: 11.04.2020 Надрукована / Paper Printed : 04.06.2020

ДОДАТОК Ж

(обов’язковий)

ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ

Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій

Автор роботи:
ст. гр. ІПЗм-19-1 Прейзнер Є. Е.
Керівник роботи:
к. т. н., доцент Яшина О. М.

Актуальність

- харчування завжди відіграє важливу роль в житті людини.
- в умовах пандемії велика кількість людей веде малорухливий спосіб життя
- не всі можуть собі дозволити послуги фахівця в даній області.

Аналіз стану проблеми та інших рішень

Переважна більшість сучасних засобів в сфері харчування мають велику кількість недоліків, основним з яких є невисока якість результатів. Крім того на даний момент жодна з програм для оптимізації раціону харчування не використовує найновіші програмні технології, а саме штучний інтелект, що можна легко помітити якщо поглянути на невелику точність результатів в деяких додатках.

Об’єкт, предмет та мета дослідження

Об’єкт дослідження - інтелектуальні інформаційні системи та засоби вибору оптимального раціону харчування.

Предмет дослідження - моделі та методи вдосконалення вибору оптимального раціону харчування із використанням смарт-технологій.

Мета дослідження - удосконалення процесу вибору оптимального раціону харчування із використанням смарт-технологій.

Завдання дослідження

- провести аналіз предметної області;
- дослідити методи, інструменти та пакети прикладних програм, які використовуються для вибору оптимального раціону харчування;
- покращити процес вибору оптимального раціону харчування за допомогою засобів штучного інтелекту;
- вдосконалити метод вибору оптимального раціону харчування із врахуванням індивідуальних особливостей;
- розробити програмну систему з реалізацією запропонованого методу;
- провести практичну апробацію отриманих результатів.

Наукова новизна

- вдосконалено метод вибору оптимального раціону харчування із врахуванням індивідуальних особливостей;
- вдосконалено процес вибору оптимального раціону харчування за допомогою засобів штучного інтелекту.

Практичне значення

Запропонований метод повинен покращити результати розробки персональних раціонів та спростити отримання якісних безкоштовних індивідуальних планів правильного харчування для широкого кола людей, що в свою чергу має допомогти вирішити загальну проблему збалансованого харчування.

Модель роботи системи

Математична модель формування індивідуального меню

$$\mu_k = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix}$$

де μ_k - змінна включення до складу раціону при наступних обмеженнях:

1) по не повторюваності страв в добовому раціоні

$$\mu_k = 1\,;\ k \in\ k_a, k_b, k_c$$

2) на граничну калорійність одного прийому їжі

$$\sum_k \mu_k Q_k \leq Q_{\text{доб}} \eta_l$$

Математична модель формування індивідуального меню

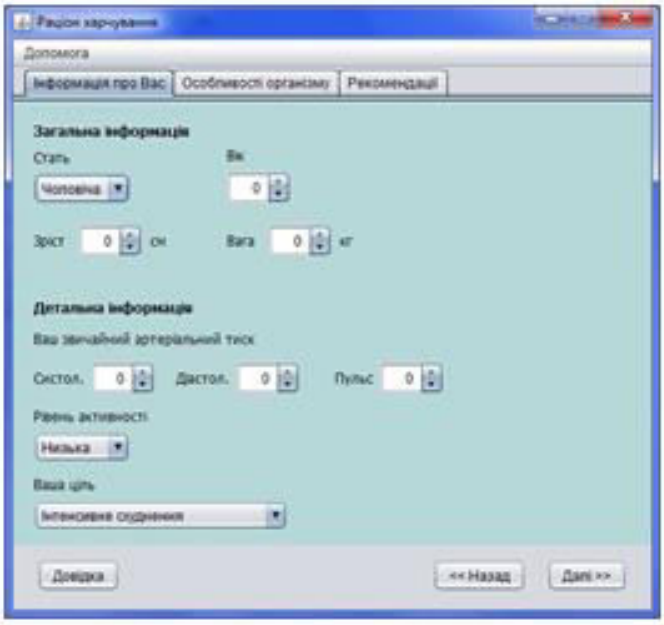
3) по верхньому і нижньому кордонах вмісту *i*-го хімічного елемента в кожному прийомі їжі

$$B_i^{min} \leq \mu_k \sum_k \alpha_{ik} \leq B_i^{max}$$

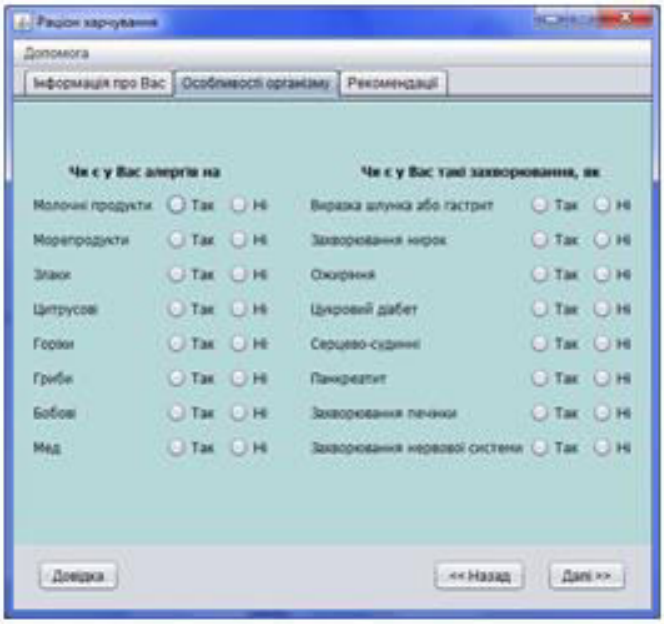
Обсяг страви в прийомі їжі розраховується за формулою

$$V_{nk} = \left(\frac{Q_{\text{доб}} \eta_l \beta_{mk}}{Q_k} \right) 100$$

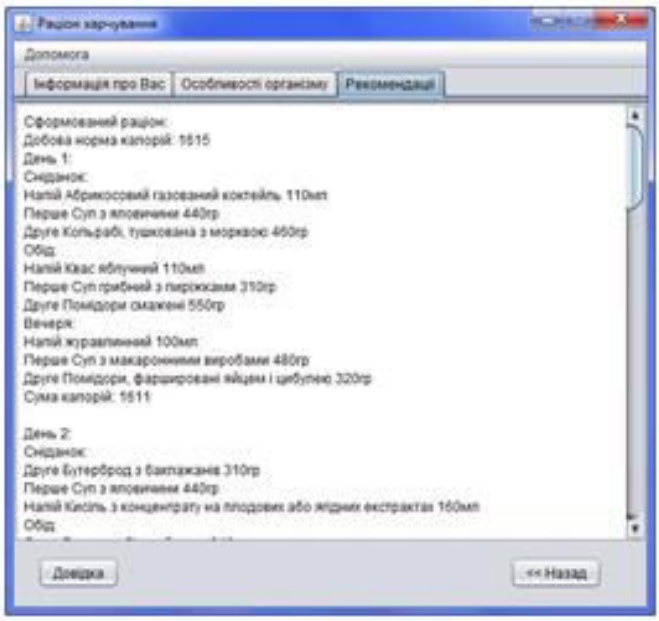
Головне вікно програми



Детальна інформація про здоров’я



Результат роботи програми



Публікації

- Прейзнер Є.Е. Продукційна модель експертної системи для вибору оптимального раціону харчування / Є.Е. Прейзнер, О.М. Яшина // Матеріали науково-практичної інтернет-конференції молодих науковців і студентів «Інтелектуальний потенціал - 2020» – С. 97–100.
- Прейзнер Є.Е. Методи штучного інтелекту в сфері охорони здоров'я / Є.Е. Прейзнер, О.М. Яшина // Міжнародний науково-технічний журнал "ВОТТП". – Хмельницький, 2020. – №1. – С. 84–87.

Висновки

- проведено аналіз предметної області;
- проаналізовано та описано існуючі методи, інструменти та рішення в предметній області;
- вдосконалено процес вибору оптимального раціону харчування за допомогою засобів штучного інтелекту;
- вдосконалено метод вибору оптимального раціону харчування із врахуванням індивідуальних особливостей;
- розроблено програмну систему, яка реалізує запропонований метод;
- опубліковано результати дослідження на науковій конференції.

Дякую за увагу

Fri Dec 04 15:58:46 EET 2020 Харин Володимир Руславович Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 2.0%

Сховища перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 8%**

	Документ		Сумарний збіг по базі Даних	
	Символи	Лексеми	Символи	Лексеми
ID: 82520 Назва: Технологія розробки програмної системи вибору оптимального раціону Технологія розробки програмної системи вибору оптимального раціону Додано в БД: 2020-12-04 Автора: Є. Е. Прейзнер Керівники: О. М. Яшина Консультанти: Опоненти:	107835	1532	4195 (4%)	55 (4%)

Джерело плагіату

ID	Опис	Найбільш плагіату в документі	
		Символи	Лексеми

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНУ РОБОТУ

Дипломник студент групи ІПЗм-19-1 Прейзнер Є.Е.

Тема Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій

Спеціальність 121 – Інженерія програмного забезпечення

Обсяг дипломної роботи:

Кількість листів креслень 0 ; кількість сторінок записки 82

1.Короткий зміст ДР та прийнятих рішень Представлена робота присвячена актуальній темі в області охорони здоров'я, і складається з наступних розділів: вступ, дослідження предметної області, вибір методів та моделей для вирішення проблеми, проектування програмного забезпечення для вирішення проблеми, реалізація програмного забезпечення для вирішення проблеми, висновки, додатки.

2. Висновок про відповідність ДР поставленому завданню Магістерська кваліфікаційна робота виконана у відповідності з завданням із дотриманням всіх вимог.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: В першому розділі студент провів детальний аналіз предметної області, дослідив процеси харчування людини і оптимізації раціону, представив та описав особливості існуючих рішень в розглянутій сфері, на основі яких довів актуальність роботи і визначив вимоги для створюваної системи. В другому розділі було досліджено всі існуючі підходи до реалізації штучного інтелекту, обґрунтовано чому саме обраний підхід найкраще підходить для вирішення проблеми, а також запропоновано новий метод оптимізації, який полягає у використанні експертної системи разом із наведеною математичною моделлю. Відповідно в третьому і четвертому розділах автором проведено проектування системи, здійснено опис програмної реалізації і протестовано розроблений продукт.

4. Позитивні сторони роботи До позитивних сторін роботи слід віднести вибір актуального направлення дослідження, всебічний та детальний аналіз усіх існуючих стратегій вирішення проблеми, а також глибоке опрацювання всіх аспектів реалізації і використання запропонованого рішення.

5. Негативні сторони роботи В ході рецензування виявлено недоліки по оформленню представленого матеріалу, які були усунені.

6. Оцінка графічного оформлення та пояснювальної записки роботи Представлені матеріали роботи чітко та логічно структуровані, що відображає послідовність виконання поставлених завдань. Проте викладення матеріалу мало декілька стилістичних та орфографічних помилок, які згодом були виправлені. Таким чином виконання пояснювальної записки та графічного оформлення заслуговує оцінки «добре».

7. Відгук про роботу в цілому Зміст представленої роботи в повній мірі розкриває обрану тему. Проведені дослідження в достатній мірі аргументовані, виконані на високому теоретичному та практичному рівні. Результатом проведення досліджень стали відповідні висновки і конкретні пропозиції щодо вдосконалення процесу вибору оптимального раціону харчування за допомогою використання засобів штучного інтелекту.

8. Інші зауваження

9. Оцінка дипломної роботи Робота заслуговує оцінки «добре», а її автор – присвоєння кваліфікації «магістра» з інженерії програмного забезпечення.

РЕЦЕНЗЕНТ (прізвище, ім'я, по-батькові, посада, місце роботи) Говорущенко Тетяна Олександрівна, доктор технічних наук, професор, зав. кафедри комп'ютерної інженерії та системного програмування ХНУ

"02" грудня 2020 р. _____ (підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ ПО КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: **Технологія розробки програмної системи вибору оптимального раціону харчування із використанням смарт-технологій**

Автор: **Прейзнер Євгеній Едуардович**

Спеціальність: **121 «Інженерія програмного забезпечення»**

Освітня програма: «Інженерія програмного забезпечення»

Науковий керівник: **к.т.н., доцент Яшина Оксана Миколаївна**

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	–
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	–
3	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	–
4	Інше:	–

Підтвердження: Виявлені запозичення не є плагіатом, так як розміщені в розділах, що не описують безпосередньо авторське дослідження, складають 6.53 % та мають посилання на приведений список літературних джерел, стандартні конструкції коду.

7.12.2020р.
Дата

Підпис керівника

Підпис завідувача кафедри

Гарант ОП