Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студентки	*Овчаренко Світлани Олександрівни*
	(ПІБ)
академічної групи	*121М-21-1*
	(шифр)
Спеціальності	*121 Інженерія програмного забезпечення*
	(код і назва спеціальності)
освітньої програми	*«Інженерія програмного забезпечення»*
	(назва освітньої програми)
на тему:	*Розробка програмного забезпечення для підвищення ефективності постановки діагнозу з використанням методів штучного інтелекту*

_____*С.О. Овчаренко*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
розділів кваліфікаційної роботи				
спеціальний	*Проф. Алексєєв М.О.*			
Рецензент				
Нормоконтролер	*Проф. Лактіонов І.С.*			

Дніпро
2022

Міністерство освіти і науки України

Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

_____ Алексєєв М.О
(підпис) (прізвище, ініціали)

« » _____ 20 22 Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

студенту *121м-21-1* *Овчаренко Світлані Олександрівні*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Розробка програмного забезпечення для підвищення ефективності постановки діагнозу з використанням методів штучного інтелекту*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від __.__.2022 р. № ____

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – програмне забезпечення для аналізу симптомів та встановлення діагнозу на базі цього, його можливості та недоліки.

Предмет досліджень – DiagnosticYouTool - програмне забезпечення для постановки діагнозу за симптомами.

Мета НДР – покращити, спростити та пришвидчити процес постановки діагнозу, шляхом створення релевантного програмного забезпечення.

Вихідні дані для проведення роботи – теоретичні та експериментальні дослідження, основи нечіткої моделі даних при вирішенні задач ідентифікаційної експертизи.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень визначається тим, що вперше розроблено та оптимізовано DiagnosticYouTool за допомогою бази знань та навченої логіки мислення як лікар спрощує та пришвидшує процес постановки діагнозу

виключаючи непідходящі діагнози шляхом використання вірогідностної моделі.

Практична цінність результатів полягає у тому, що запропоновані в роботі моделі і методи дозволяють накопичувати та використовувати знання нейронної мережі та штучного інтелекту для вирішення задач постановки діагнозу.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Результати досліджень мають бути подані у вигляді, що дозволяє побачити та оцінити безпосереднє використання нейронної мережі та штучного інтелекту в постановці діагнозу. В результаті роботи повинен бути розроблений програмний комплекс для вирішення задачі аналізу симптомів та прогнозованої хвороби з процентною вірогідністю.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець**
Аналіз теми та постановка задачі	16.09.2022-30.09.2022
Побудова моделі нейронної мережі і створення бази знань	01.10.2022-31.10.2022
Створення DiagnosticYouTool	01.11.2022-18.12.2022

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект від реалізації результатів роботи очікується позитивним завдяки оптимізації затрат людини на не профільних лікарів та економії часу лікаря.

Соціальний ефект від реалізації результатів роботи очікується позитивним завдяки пришвидченню та оптимізації постановки діагнозу, що дозволяє зменшити людський фактор та помилку при визначенні чи свідчать данні симптоми про конкретну хворобу.

7 ДОДАТКОВІ ВИМОГИ

Завдання видав _____ *Алексєєв М.О.*
(підпис) (прізвище, ініціали)

Завдання прийняла до виконання _____ *Овчаренко С.О.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: _____ *р.*

Термін подання кваліфікаційної роботи до ЕК _____

РЕФЕРАТ

Пояснювальна записка: 78 стор., 42 рис., 2 додатки, 13 джерел.

Об'єкт дослідження: програмне забезпечення для аналізу симптомів та встановлення діагнозу на базі цього, його можливості та недоліки.

Предмет дослідження: DiagnosticYouTool - програмне забезпечення для постановки діагнозу за симптомами.

Мета роботи: покращити, спростити та пришвидчити процес постановки діагнозу, шляхом створення релевантного програмного забезпечення.

Методи дослідження. Для вирішення поставлених задач використані методи: аналізу даних, штучного інтелекту, нейронних мереж, теорії розпізнавання тексту з області обчислювального інтелекту, теорії нечітких множин, об'єктно-орієнтоване програмування.

Новизна отриманих результатів визначається тим, що вперше розроблено та оптимізовано DiagnosticYouTool за допомогою бази знань та навченої логіки мислення як лікар спрощує та пришвидшує процес постановки діагнозу виключаючи непідходящі діагнози шляхом використання вірогідностної моделі.

Практична цінність результатів полягає у тому, що запропоновані в роботі моделі і методи дозволяють накопичувати та використовувати знання нейронної мережі та штучного інтелекту для вирішення задач постановки діагнозу.

Область застосування: розроблена програмна система може бути використана як на сайті клініки при записі на прийом так і на прийомі безпосередньо у лікаря, а в не гострих випадках лікарями невідкладної допомоги на викликах.

Значення роботи та висновки: оптимізована система штучного інтелекту та технології нейронних мереж дозволяють проектувати програмні системи зі значним скороченням як матеріальних витрат, так і часових, що підтверджується розробленим програмним продуктом в даній роботі.

Прогнози щодо розвитку досліджень: покращити програмний продукт, додавши метод кластеризації класів діагнозів, з метою зменшення відсотка похибки при ідентифікації діагнозу.

У розділі «Економіка» проведені розрахунки трудомісткості розробки програмного забезпечення, витрат на створення ПЗ і тривалості його розробки, а також проведені маркетингові дослідження ринку збуту створеного програмного продукту.

Список ключових слів: база знань, діагноз, симптом, штучний інтелект, нейронна мережа, інформаційна система, програмний продукт.

ABSTRACT

Explanatory note: 78 pages, 42 figures, 2 applications, 13 sources.

Object of research: software for analyzing symptoms and establishing a diagnosis based on them, its capabilities and shortcomings.

Subject of research: DiagnosticYouTool - software for diagnosis by symptoms.

Purpose of Master's thesis: to improve, simplify and speed up the process of diagnosis by creating relevant software.

Research methods: The following methods are used to solve the problems: data analysis, artificial intelligence, neural networks, text recognition theory from the field of computational intelligence, fuzzy set theory, object-oriented programming.

Originality of research is associated with first time developing DiagnosticYouTool and its optimization with the help of a knowledge base and a trained logic of thinking as a doctor simplifies and speeds up the process of making a diagnosis by excluding inappropriate diagnoses through the use of a probabilistic model.

Practical value of the results consists of the models and methods proposed in the work allow you to accumulate and use knowledge of the neural network and artificial intelligence to solve the problems of diagnosis.

Scope of application: The developed software system can be used both on the website of the clinic when making an appointment and at the appointment directly with the doctor, and in non-urgent cases by emergency doctors on calls.

The value of the work and conclusions: an optimized system of artificial intelligence and neural network technologies allow designing software systems with a significant reduction of both material costs and time, which is confirmed by the developed software product in this work.

Research forecast and development: improve the software product by adding a method of clustering classes of diagnoses in order to reduce the percentage of error in diagnosis identification.

In the Economics section we calculated: the complexity of software

development, the costs of creating software and the duration of its development, as well as marketing research of the sales market of the created software product.

Keywords: knowledge base, diagnosis, symptom, artificial intelligence, neural network, information system, software product.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БЗ – база знань;

ШІ – штучний інтелект;

НМ – нейронна мережа;

ПЗ – програмний застосунок;

ОПР – особа, що приймає рішення;

СППР – система підтримки прийняття рішень;

UML – Unified Modeling Language.

ЗМІСТ

ВСТУП

В час інформаційних технологій все більше і більше завдань делегуються людьми комп’ютерам, що не тільки пришвидшує процес виконання, а й зменшує похибку, оскільки людська пам’ять обмежена та може бути зайнята власними думками, переживаннями, чи проблемами. Зі збільшенням кількості оброблюваної інформації, погіршенням екології, стресами, відсутністю гігієни сна, нездоровим способом життя, сидячою роботою останнім часом все більше людей страждають від якогось дискомфорту. При процессі пошуку інформаціїї людина може “нагуглити” собі найгірший діагноз та почати панікувати, або навпаки подумати, що хвороба пройде без лікування і тим самим втратити час і запустити її до хронічного стану. Тим часом ритм життя не дає можливості витрачати багато часу на відвідування лікарів, ліміт часу відведеного на одного пацієнта в приватних клініках становить пів години і щоб потрапити до профільного спеціаліста треба обов’язково відвідати терапевта. Для оптимізації та прискорення постановки діагнозу, а також зменшення часу проведеного на консультаціях лікарів, виникає нагальна потреба у створенні додатку, що буде мати хід думок лікаря, буде предсталяти собою нейронну мережу що навчатиметься на базі знань та використовуючи штучний інтелект та відогідністний аналіз буде ставити діагноз та радити якого лікаря слід відвідати та які питання краще йому поставити.

Виходячи з вищеописаної інформації, було прийнято рішення розробити додаток, що дозволяє аналізувати симптоми та ставити діагноз. Наукова новизна цього обумовлена саме використанням алгоритму що має хід думок лікаря-терапевта та вірогідностного підходу, бо як би ми не обмежували сет симптомів все одно деякі хвороби дуже схожі і відрізнити їх може лише лікар фахівець з досвідом на прийомі. А також використанням агрегованого списку симптомів та діагнозів, що дає змогу не забути про якесь не часто виникаюче захворювання.

Завдання даної кваліфікаційної роботи та об'єкт його діяльності безпосередньо пов'язані з освітньою програмою «Інженерія програмного забезпечення» та відповідає узагальненій тематиці кваліфікаційних робіт і переліку зазначених виробничих функцій, типових задач діяльності, умінню та компетенціям, якими повинні володіти магістри освітньої програми 121 «Інженерія програмного забезпечення» галузі знань 12 «Інформаційні технології». Виконання кваліфікаційної роботи надає можливості отримання автору кваліфікації «магістр з інженерії програмного забезпечення».

РОЗДІЛ 1
АНАЛІЗ ТЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Задача нейронної мережі

Перш за все треба зазначити що таке нейронна мережа: нейронна мережа – один із напрямків штучного інтелекту, мета якого змоделювати аналітичні механізми, що здійснюються людським мозком. Завдання, які вирішує типова нейромережа – класифікація, передбачення та розпізнавання. Нейросети здатні самостійно навчатися і розвиватися, будуючи свій досвід на помилках.

Нейросети це послідовність нейронів, з'єднаних між собою синапсами. Структура нейронної мережі прийшла у світ програмування прямо з біології. Завдяки такій структурі машина знаходить здатність аналізувати і навіть запам'ятовувати різну інформацію. Також нейронні мережі здатні як аналізувати вхідну інформацію, а й відтворювати її з власної пам'яті.

Іншими словами, нейромережа це машинна інтерпретація мозку людини, в якому знаходяться мільйони нейронів, що передають інформацію у вигляді електричних імпульсів.[2]

Клас задач, які можна вирішити за допомогою нейронної мережі, визначається тим, як мережа працює і як вона навчається. Таким чином, мережі можна застосовувати в ситуації, коли є визначеною відома інформація, і ви намагаєтесь з неї одержати якусь поки що не відому інформацію.

Застосування класичних статистичних методів для рішення задачі визначення стану хворого було описане ще в роботах Неймана. За допомогою медичної апаратури можна спостерігати за різними показниками стану здоров'я людини (наприклад, частотою пульсу, змістом різних речовин у крові, частотою подиху). Стадії виникнення деякої хвороби може відповідати визначена і досить складна (наприклад, нелінійна і взаємозалежна) комбінація змінювання змінних, що спостерігаються, котра може бути виявлена за допомогою нейромережевої моделі.[3]

1.2 Вірогідносний аналіз

Імовірнісний метод являє собою метод неконструктивного доведення, що, в першу чергу, використовується у комбінаториці для доведення існування наперед визначеного виду математичних об'єктів. Він працює через демонстрацію того, що, якщо випадково обрати об'єкти з деякого класу, ймовірність того, що результат є визначеного вигляду, більше нуля. Хоча доведення використовує ймовірність, кінцевий висновок визначається напевно, без будь-якої похибки. Якщо кожен об'єкт у наборі об'єктів не має певної властивості, тоді імовірність, що випадковий об'єкт, обраний з цього набору має цю властивість, становить нуль. І у зворотному напрямку, якщо імовірність, що випадковий об'єкт має цю властивість, більше нуля, тоді це доводить існування принаймні одного об'єкта у даному наборі, що має цю властивість. Навіть якщо ймовірність дуже мала; будь-яка позитивна ймовірність доводить це твердження.

Аналогічно, демонстрація того, що ймовірність (строго) менше 1, може бути використане для доведення існування об'єкта, що не задовольняє наперед визначеним вимогам. В медицині та фармакоінформатиці для цього може бути використана модель Маркова, з точки зору якої усі події це перехід з одного стану в інший, при цьому проводять припущення, що всі люди здорові (повне здоров'я - 1), а в момент виявлення захворювання вони переходять в інший стан. Частина хворих має більш ранні стадії і переходить у більш пізні стадії захворювання, а частина, навпаки, із більш пізніх стадій до більш ранніх за рахунок активного правильного лікування. Хворий завжди перебуває у певному стані.[4]

1.3 Застосування автоматизації та навчання для постановки діагнозу

Щодо медичної тематики експериментальні дані надаються в вигляді множини вихідних ознак або параметрів об'єкта і поставленого на їх основі діагнозу. Навчання нейронної мережі являє собою інтерактивний процес, в ході якого нейронна мережа знаходить приховані нелінійні залежності між вихідними параметрами і кінцевим діагнозом, а також оптимальну комбінацію вагових коефіцієнтів нейронів, що з'єднують сусідні шари, при якій похибка визначення класу образу прагне мінімуму [5].

Хороші результати показали моделі штучних нейронних мереж для діагностики психічних розладів, цукрового діабету, хвороби Паркінсона та Хантінгтона. Моделі багатоголосних персептронів застосовуються для прогнозування ризику виникнення остеопорозу. Логічний висновок та узагальнена регресія використані для діагностування гепатиту В [6].

Кожна система розпізнавання пристосована для розпізнавання тільки даного виду об'єктів або явищ (так, система, призначена для діагностики захворювань, не може діагностувати відмови апаратури, а система, призначена для читання букв алфавіту кириліцею , не може читати китайські ієрогліфи чи ноти) [7].

Необхідно розглянути змістовну та формальну постановку проблеми розпізнавання, що базуються на наступних положеннях.По-перше, розв'язання задач розпізнавання вимагає в загальному випадку побудови спеціальної системи розпізнавання. По-друге, рішення завдання розпізнавання необхідне (також загалом випадку) для того, щоб система управління, стоїть над системою розпізнавання, могла приймати правильні рішення. Наприклад, система медичної діагностики покликана встановлювати діагноз хворих на того, щоб лікар міг приймати обґрунтовані рішення щодо вибору стратегії лікування;

Виходячи зі сказаного, системи автоматизації повинні будуватися так, щоб забезпечувати системі управління можливість найефективніше

розпоряджатися своїми ресурсами, до пустим набором рішень, а саме побудова систем автоматизації та розпізнавання як і будь-яких технічних

систем, що не може бути здійснено без урахування відповідних обмежень.

Широке коло завдань, які вирішуються за допомогою нейромереж, не дозволяє поки створити універсальні потужні мережі, змушуючи розробляти

спеціалізовані нейронні мережі, що функціонують за різними алгоритмами. Основними перевагами нейронних мереж для вирішення складних завдань медичної діагностики є:

- Відсутність необхідності завдання у явній формі математичної моделі та перевірки справедливості серйозних припущень для використання статистичних методів;

- Інваріантність методу синтезу від розмірності простору, ознак і розмірів нейронних мереж та ін.

Однак використання нейронних мереж для задач медичної діагностики пов'язано також з низкою серйозних труднощів. До них слід віднести необхідність щодо великого Обсягу вибірки для налаштування мережі, орієнтованість математичного апарату на кількісні змінні.

Ідеальний метод діагностики повинен

мати стовідсоткові чутливість та специфічність:

– по-перше, не пропускати жодної дійсно хворої людини;

– по-друге, не переписувати діагноз здоровим людям.

Щоб запобігти пропуску випадку захворювання, можна забезпечити стовідсоткову чутливість методу. Але в такому випадку для системи це обертається, як правило, низькою специфічністю методу - у багатьох людей система діагностуватиме захворювання, яким насправді пацієнт не страждає.

Нейронним мережам на етапі навчання необхідний «учитель», який заздалегідь виконує класифікацію образів, що входять до навчальної вибірки і організовує її введення в комп'ютер. У конкурентних мережах реалізується «навчання без вчителя». Тобто в навчальній вибірці для образів невідомі

правильні (бажані) вихідні реакції. Для реалізації цього підходу необхідно вирішити дві основні проблеми:

– розробити методи розбиття образів на класи без вчителя – етап навчання;

- Виробити правила віднесення поточного вхідного образу до деякого класу – етап розпізнання.

Навчання нейронної мережі в більшості випадків є автоматизований процес, в якому тільки після його закінчення потрібна участь фахівця для оцінки результатів. Звичайно, часто може вимагатися коригування, створення додаткових мереж з іншими параметрами і т.д., проте завжди є можливість оцінити роботу системи на будь-якому етапі навчання, протестувавши контрольну вибірку.

Як зазначалося вище, навчання НМ виконується фактично за два етапи:

1. Навчання на навчальній множині поки не виконано одну з умов зупинки:

– помилка на навчальній множині стає менше заданої величини;

– помилка перестає змінюватися протягом певної кількості ітерацій;

– досягнуто верхньої межі числа ітерацій навчання.

2. Перевірка правильності;

1.4 Існуючі рішення постановки діагнозу

На даний момент існує сервіс Symptomate, що дає можливість пройти тест і визначити можливий діагноз базуючись на обтяжливих станах та звичках.

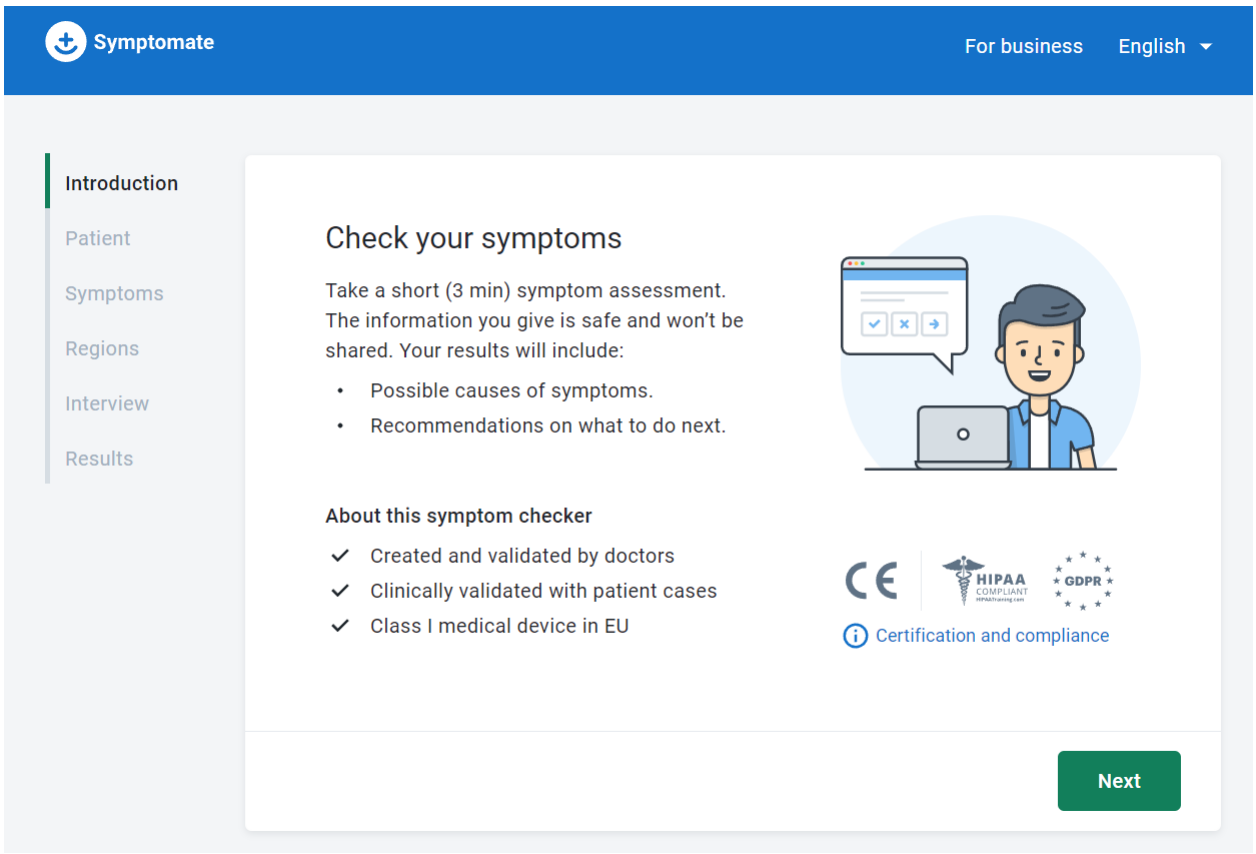


Рис. 1.4.1 Вікно Symptomate

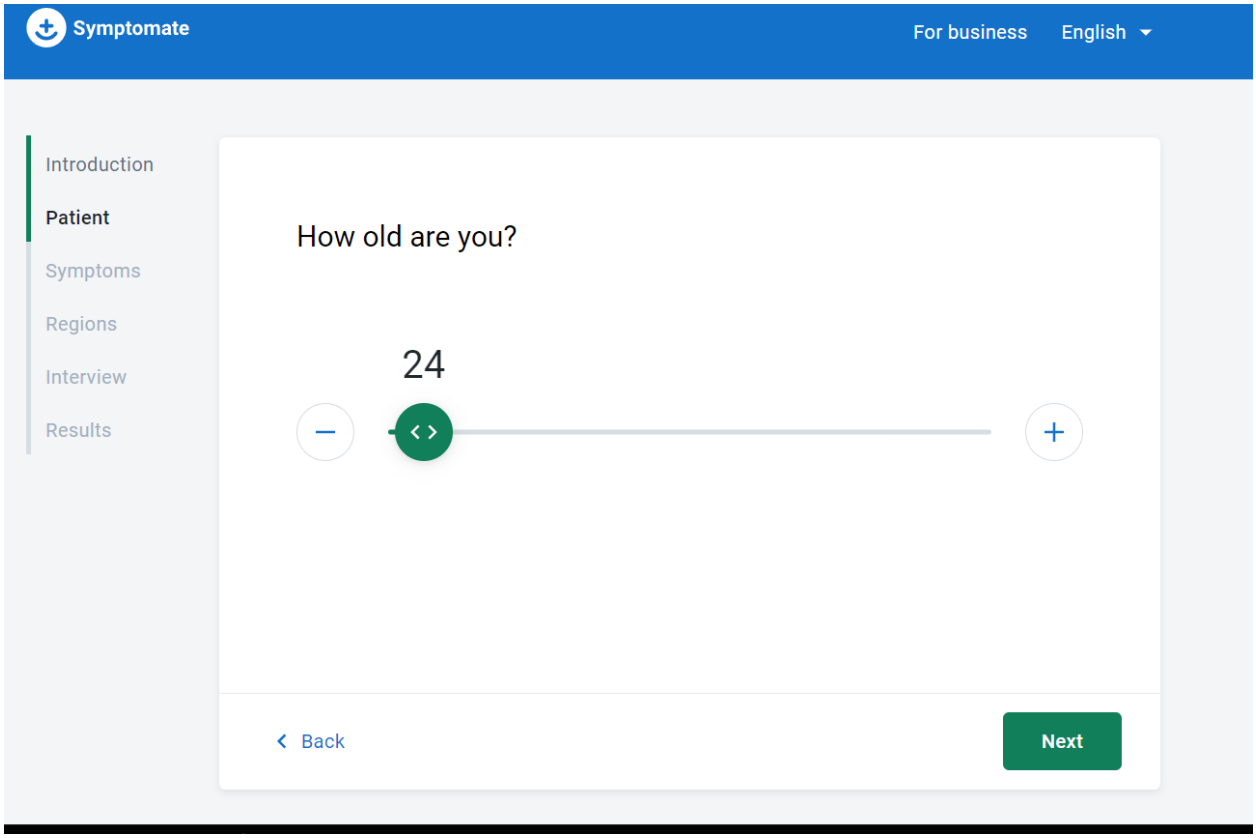


Рис. 1.4.2 Вікно вводу віку

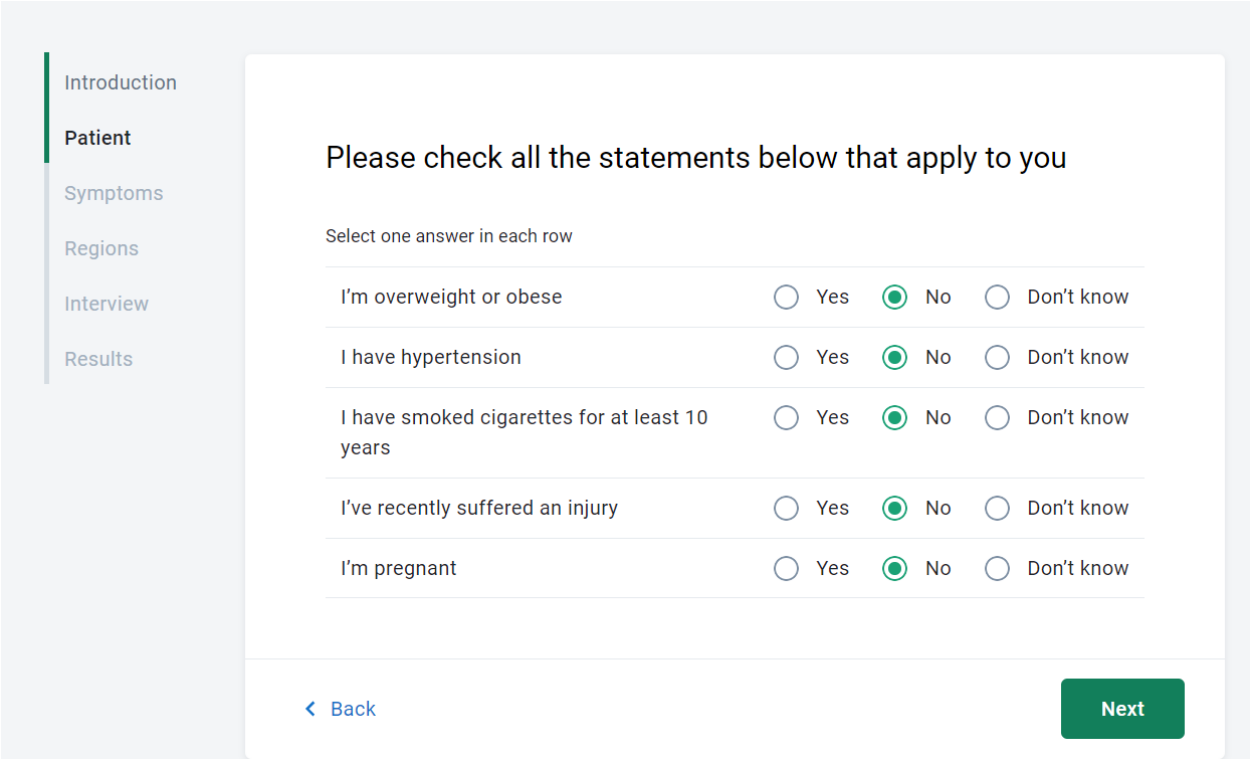


Рис. 1.4.3 Вікно обрання обтяжливих станів

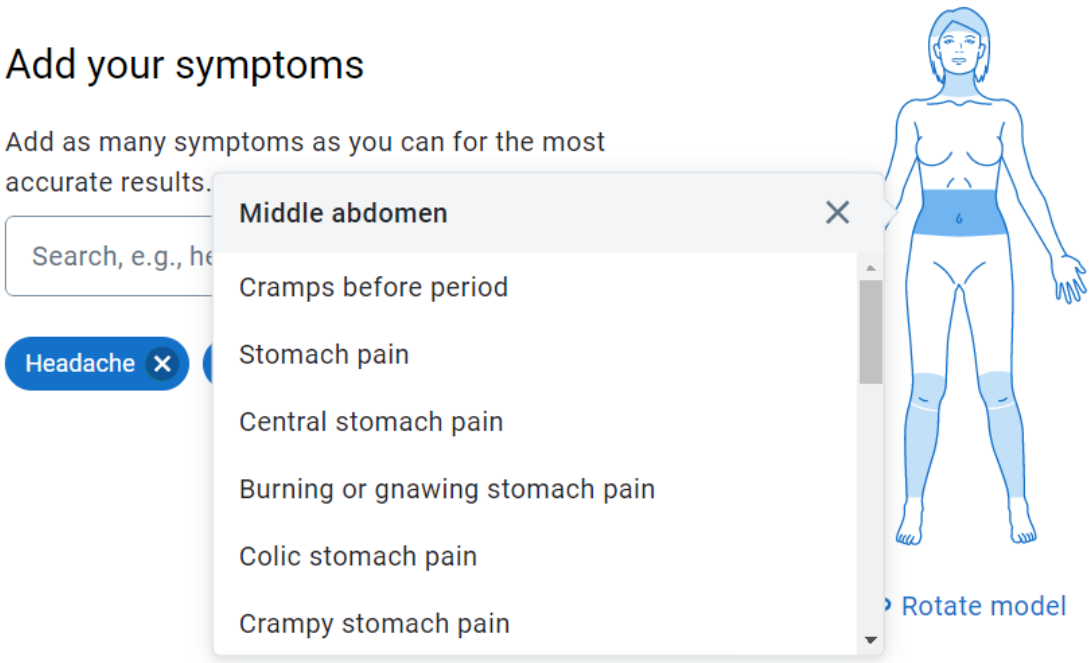


Рис. 1.4.4 Вікно обрання симптомів

Рис. 1.4.5 Вікно можливих хвороб

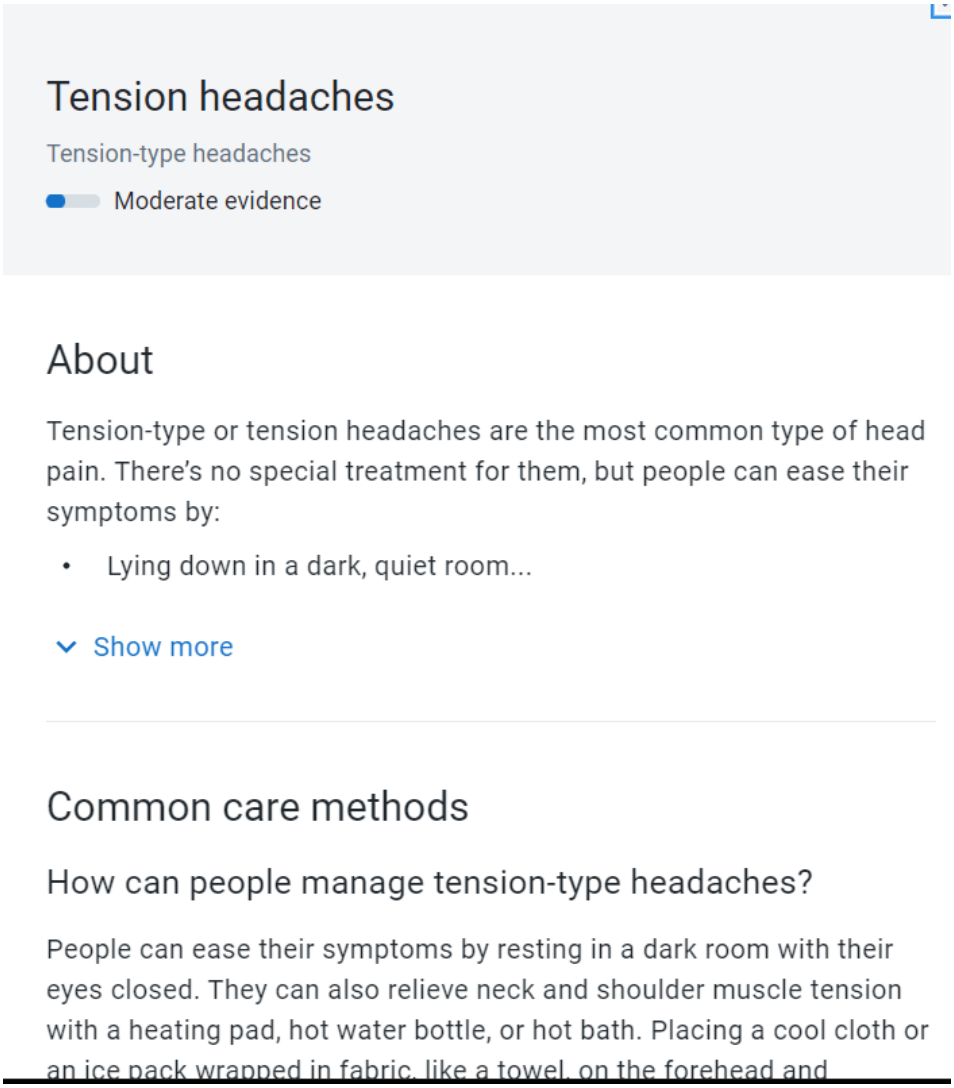


Рис. 1.4.6 Вікно обраного лікарем результату

1.5 Висновки

Ймовірносний аналіз та нейронні мережі добре підходять для встановлення діагнозу, в наш час на ринку представлений один англійськомовний додаток, що може поставити діагноз за симптомами проте його точність ще досить низька бо серед запропонованих варіантів дуже багато хвороб, деякі з яких не дуже стосуються цих симптомів.

РОЗДІЛ 2
ПОБУДОВА НЕЙРОННОЇ МЕРЕЖІ ДЛЯ ПОСТАНОВКИ ДІАГНОЗУ
2.1 Основні терміни нейронних мереж

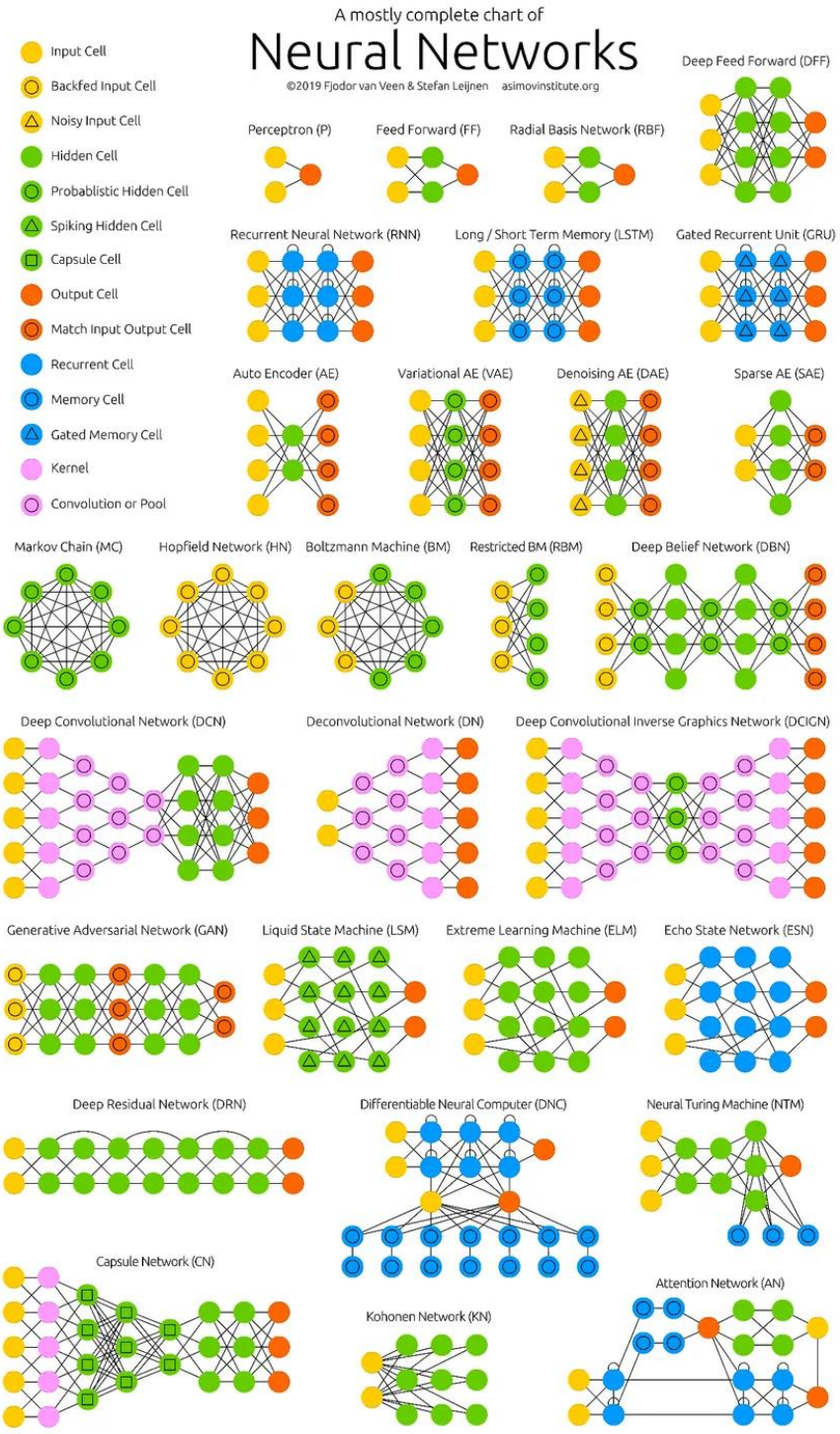
Навчання – це можливість нейронної мережі що викликає найбільше зацікавлення. Для заданої конкретної задачі для розв'язання та класу функцій навчання означає використання набору спостережень для знаходження функції, яка розв'язує цю задачу в певному оптимальному сенсі.

Це тягне за собою визначення такої функції витрат, для оптимального розв'язку при якому жоден розв'язок не має витрат, менших за витрати оптимального.

Функція витрат є важливим поняттям у навчанні, оскільки вона є мірою того, наскільки далеким є певний розв'язок від оптимального розв'язку задачі, яку потрібно розв'язати. Алгоритми навчання здійснюють пошук простором розв'язків, щоб знайти функцію, яка має найменші можливі витрати.

Для тих застосувань, де розв'язок залежить від даних, витрати обов'язково мусять бути функцією від спостережень, бо інакше модель не матиме зв'язку з даними. Їх часто визначають як статистику, для якої може бути зроблено лише наближення, тоді витрати зводяться до мінімуму над вибіркою з даних, а не над усім розподілом.

2.1.1 Основні типи нейронних мереж



[7]

Рис. 2.1.1 Типи нейронних мереж

1. Дрібні нейронні мережі (спільна фільтрація)

Нейронні мережі складаються з груп Перцептрону для імітації нервової структури людського мозку. Дрібні нейронні мережі мають єдиний прихований шар перцептрона. Одним із поширених прикладів дрібних нейронних мереж є спільна фільтрація. Прихований шар перцептрона буде навчений представляти схожість між сутностями з метою формування рекомендацій. Система рекомендацій в Netflix, Amazon, YouTube тощо використовує версію спільної фільтрації, щоб рекомендувати свої продукти відповідно до інтересів користувачів.

2. Багатошаровий перцептор (глибокі нейронні мережі)

Нейронні мережі з більш ніж одним прихованим шаром називаються глибокими нейронними мережами. Спойлер попередження! Усі наступні нейронні мережі є формою глибокої нейронної мережі, налаштованої / вдосконаленої для вирішення проблем, пов'язаних з доменом. Загалом, вони допомагають нам досягти універсальності. Враховуючи достатню кількість прихованих шарів нейрона, глибока нейронна мережа може наблизитись, тобто вирішити будь-яку складну реальну проблему.

Теорема універсального наближення є ядром глибоких нейронних мереж для навчання та підгонки будь-якої моделі. Кожна версія глибокої нейронної мережі розробляється повністю зв'язаним шаром максимізованого продукту множення матриць, який оптимізований алгоритмами зворотного розповсюдження. Ми продовжуватимемо вивчати вдосконалення, що призводять до різних форм глибоких нейронних мереж.

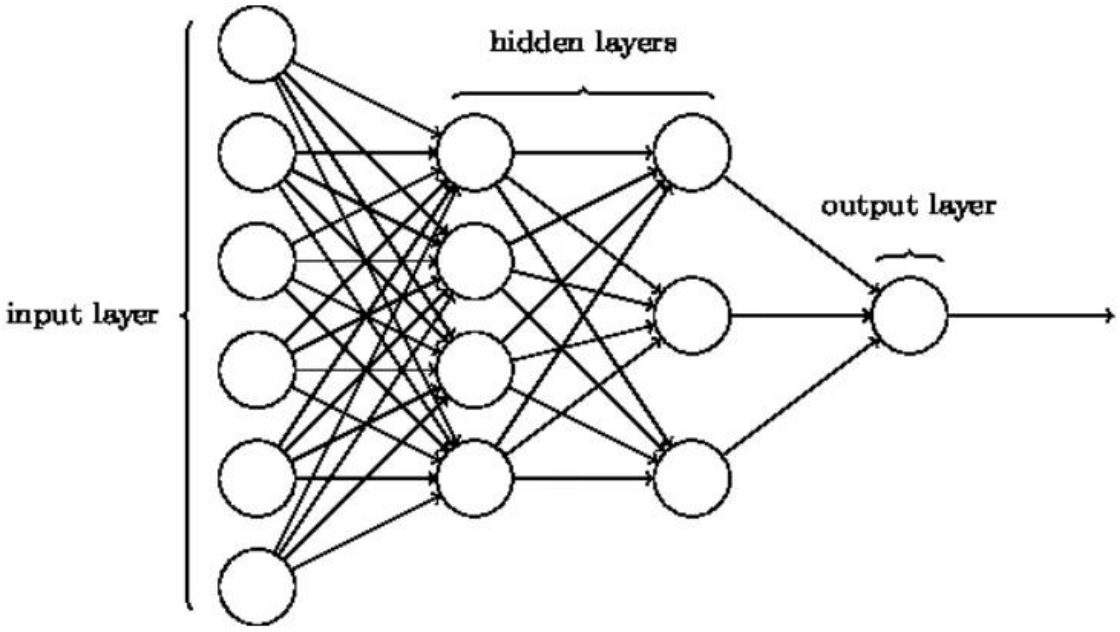


Рис. 2.1.2- Схема нейронної мережі

3. Конволюційна нейронна мережа (CNN)

CNN - це найзріліша форма глибоких нейронних мереж для отримання найбільш точних, тобто кращих, ніж у людини результатів комп'ютерного зору. CNN складаються з шарів згортків, створених шляхом сканування кожного пікселя зображень у наборі даних. Коли дані отримують приблизний рівень за шаром, CNN починає розпізнавати шаблони і тим самим розпізнавати об'єкти на зображеннях. Ці об'єкти широко використовуються в різних програмах для ідентифікації, класифікації тощо. Останні практики, такі як навчання трансферу в CNN, призвели до значних поліпшень неточності моделей. Google Translator і Google Lens - це найсучасніший приклад CNN.

Застосування CNN є експоненціальним, оскільки вони навіть використовуються для вирішення проблем, які в першу чергу не пов'язані з комп'ютерним зором. Тут можна знайти дуже просте, але інтуїтивне пояснення CNN.

4. Повторна нейронна мережа (RNN)

RNN - це найновіша форма глибоких нейронних мереж для вирішення проблем в NLP. Простіше кажучи, RNN подають вихід декількох прихованих

шарів назад до вхідного шару для агрегації та передачі наближення до наступної ітерації (епохи) вхідного набору даних. Це також допомагає моделі самостійно вчитися і швидше коригує прогнози. Такі моделі дуже корисні для розуміння семантики тексту в операціях NLP. Існують різні варіанти RNN, такі як Long Short Term Memory (LSTM), Reated Recurrent Unit (GRU) Gated тощо. На схемі нижче активація h1 та h2 подається відповідно на вхід x2 та x3.

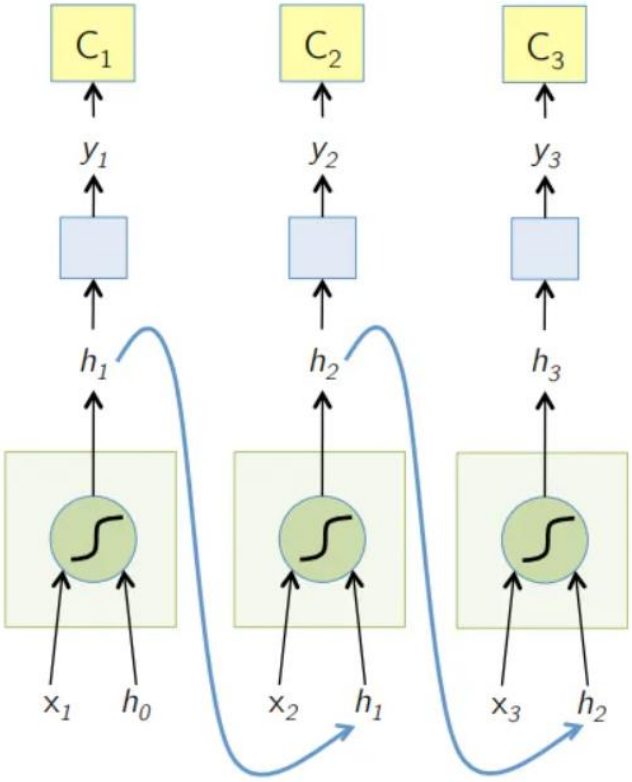


Рис. 2.1.3- Схема повторної нейронної мережі

5. Довга короткострокова пам'ять (LSTM)

LSTM розроблені спеціально для вирішення проблеми зниклих градієнтів з RNN. Зміна градієнтів трапляється з великими нейронними мережами, де градієнти функцій втрат, як правило, наближаються до нуля, роблячи призупинення нейронних мереж для навчання. LSTM вирішує цю проблему, запобігаючи функції активації в її періодичних компонентах і не змінюючи збережені значення. Ця невелика зміна значно покращила кінцеву модель, в

результаті чого технічні гіганти адаптували LSTM у своїх рішеннях. Переглянувши «найпростішу пояснення LSTM»

6. Мережі на основі уваги

Моделі уваги повільно переймають навіть нові RNN на практиці. Моделі уваги побудовані, орієнтуючись на частину підмножини інформації, яку вони надають, тим самим виключаючи переважну кількість фонової інформації, яка не потрібна для виконання завдання. Моделі уваги побудовані за допомогою комбінації м'якої та жорсткої уваги та облягання завдяки м'якій увазі, що поширюється ззаду Моделі з декількома увагами, складеними ієрархічно, називається Трансформер. Ці трансформатори ефективніше паралельно керувати стеками, так що вони дають найсучасніші результати із порівняно меншими даними та часом для навчання моделі. Розподіл уваги стає дуже потужним при використанні з CNN / RNN і може створювати опис тексту до зображення, як описано нижче.

7. Генеральна змагальна мережа (GAN)

Хоча моделі глибокого навчання дають найсучасніші результати, їх можуть обдурити набагато розумніші людські колеги, додавши шум до даних реального світу. GAN - це остання розробка в галузі глибокого навчання для вирішення подібних сценаріїв. GAN використовують непідконтрольне навчання, коли глибокі нейронні мережі навчаються з даними, згенерованими моделлю AI, а також з фактичним набором даних для підвищення точності та ефективності моделі. Ці змагальні дані в основному використовуються для того, щоб обдурити дискримінаційну модель з метою побудови оптимальної моделі. Отримана модель, як правило, є кращим наближенням, ніж може подолати такий шум. Інтерес до досліджень до GAN призвів до більш досконалих реалізацій, таких як Conditional GAN (CGAN), Laplacian Pyramid GAN (LAPGAN), Super Resolution GAN (SRGAN) тощо.

2.2 Опис штучного інтелекту в медицині

Найсучасніше застосування штучного інтелекту в медицині це - розробка і підбір медикаментів

Медичні препарати – це досить складні органічні сполуки, і пошук вірної формули потребує неабиякого часу. Адже препарати вимагають не лише ретельного підходу до розробки, а й безлічі тестів, перевірок, які теж не гарантують стовідсотково, що ліки подіють.

На допомогу розробникам може прийти штучний інтелект, який здатний швидко створювати правильні хімічні формули препаратів. Основа – задані параметри. Наприклад, зараз фірма Atomwise використовує ШІ для формування лікарських формул. Інший подібний проект розробила компанія Berg Health.

За схожим принципом проводиться і підбір медикаментів для того чи іншого пацієнта, аналізується вплив ліків на організм. Наприклад, система MedClueRx створена для того, щоб визначати, які препарати найбільш необхідні при нервових розладах, захворюваннях шлунково-кишкового тракту, епілепсії.

РОЗДІЛ 3

СТВОРЕННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПОСТАНОВКИ ДІАГНОЗУ

3.1 Структура та опис бази знань

База знань – це особливий база даних, розроблена для оперування знаннями (метаданними). База знань містить структуровану інформацію, що покриває деяку галузь знань, для використання кібернетичним пристроєм (або людиною) з конкретною метою. Сучасні бази знань працюють разом із системами пошуку інформації, мають класифікаційну структуру та формат подання знань.

Повноцінні бази знань містять у собі як фактичну інформацію, а й правила висновку, допускають автоматичні висновки про нововведених фактах як наслідок, осмислену обробку інформації. Область наук про штучний інтелект, що вивчає бази знань та методи роботи зі знаннями, називається інженерією знань.[8]

Ієрархічний метод уявлення основою знань набору понять та його відносин називається онтологією. Онтологію деякої галузі знань разом із відомостями про властивості конкретних об'єктів можна назвати базою знаний.[9]

Системи штучного інтелекту працюють з урахуванням закладених у яких баз знань. Це та модель, яка закладена програмістом або творцем у комп'ютер.

Для людини характерне як запам'ятовування деяких фактів, а й міркування про неї, і навіть аналізування, основі чого створювати логічні міркування.

У системах зі штучним інтелектом на даний момент реалізована модель міркувань (людської логіки). На основі бази знань та моделі міркувань система штучного інтелекту сама програмує свою роботу при вирішенні будь-якого завдання.

Існують два типи методів представлення знань:

1. Формальні моделі уявлення знань;

2. Неформальні (семантичні, реляційні) моделі уявлення знань.

Очевидно, всі методи представлення знань, які наведені вище, включаючи продукцію (це система правил, на яких заснована продукційна модель представлення знань), відносяться до неформальних моделей. На відміну від формальних моделей, основу яких лежить сувора математична теорія, неформальні моделі такої теорії не дотримуються. Кожна неформальна модель годиться тільки для конкретної предметної області і тому не має універсальності, яка властива формальним моделям. Логічний висновок – основна операція в СІІ – у формальних системах строгий і коректний, оскільки підпорядкований жорстким аксіоматичним правилам. Висновок у неформальних системах багато в чому визначається самим дослідником, який відповідає за його відповідність.[10]

Кожному з способів представлення знань відповідає власний метод опису знань.

1. Логічні моделі Основна ідея підходу при побудові логічних моделей уявлення знань - вся інформація, необхідна для вирішення прикладних завдань, розглядається як сукупність фактів та тверджень, які подаються як формули у певній логіці. Знання відображаються сукупністю таких формул, а отримання нових знань зводиться реалізації процедур логічного висновку. В основі логічних моделей уявлення знань лежить поняття формальної теорії, яке задається кортежем:

2. Мережеві моделі. В основі моделей цього лежить конструкція, названа раніше семантичною мережею. Мережеві моделі формально можна задати як H = . Тут I є множина інформаційних одиниць; C1, C2, ..., Cn – безліч типів зв'язків між інформаційними одиницями. Відображення P задає між інформаційними одиницями, що входять до I, зв'язку із заданого набору типів зв'язків.

3. Продукційні моделі. У моделях цього використовуються деякі елементи логічних і мережевих моделей. З логічних моделей запозичено ідею

правил виведення, які тут називаються продукціями, а з мережевих моделей - опис знань у вигляді семантичної мережі. В результаті застосування правил виведення до фрагментів мережного опису відбувається трансформація семантичної мережі за рахунок зміни її фрагментів, нарощування мережі та виключення з неї непотрібних фрагментів. Таким чином, у продукційних моделях процедурна інформація явно виділена та описується іншими засобами, ніж декларативна інформація. Замість логічного висновку, характерного для логічних моделей, у продукційних моделях з'являється висновок на знаннях.

4. Фреймові моделі. На відміну від інших типів у фреймових моделях фіксується жорстка структура інформаційних одиниць, що називається протофреймом.[8]

При створенні ідеального штучного інтелекту необхідно було врахувати та використати всі моделі уявлення знань. Величезна кількість вчених, які працюють над проблемою структуризації та застосування цих баз знань.

Діалог в додатку, побудований на методі «питання-відповідь». Машина видаватиме заздалегідь запрограмований текст, і визначатиме наскільки він співпадає з множиною пов’язаною з тим діагнозом.

3.2 Опис технологій

Для створення клієнтського Web-застосунку обрано бібліотеку React.

React - це бібліотека JavaScript, розроблена Facebook в 2013 році, яка відмінно підходить для створення сучасних односторінкових додатків будь-якого розміру і масштабу.

Переваги React:

– Легко вивчити, завдяки простому дизайну, використання JSX (HTML-подібний синтаксис) для шаблонів і дуже докладної документації.

– Розробники витрачають більше часу на написання сучасного JavaScript і менше турбуються про код, специфічному для фреймворка.

– Дуже швидкий, завдяки реалізації React Virtual DOM і різним оптимізаціям рендеринга.

– Відмінна підтримка рендеринга на стороні сервера, що робить його потужною платформою для контенто-орієнтованих додатків.

– Першокласна підтримка Progressive Web App (PWA) завдяки генератору додатків `create-react-app`.

– Прив'язка даних є односторонньою, що означає менше небажаних побічних ефектів. Redux, найпопулярніша платформа для управління станом додатків в React, її легко вчити і використовувати.

– React реалізує концепції функціонального програмування (FP), створюючи простий в тестуванні і багаторазово використовуваний код.

– Додатки можуть бути створені за допомогою TypeScript або Facebook's Flow, що мають вбудовану підтримку JSX.

– Перехід між версіями, як правило, дуже простий: Facebook надає «кодові модулі» для автоматизації більшої частини процесу.

– Навички, отримані в React, можуть бути застосовані до розробки на React Native.

Недоліки React:

– React не однозначний і залишає розробникам можливість вибирати кращий спосіб розвитку.

– Це може бути вирішено сильним лідерством проекту і хорошими процесами. Спільнота ділиться по способам написання CSS в React, які поділяються на традиційні таблиці стилів (CSS Modules) і CSS-in-JS (тобто Emotion і Styled Components).

– React відходить від компонентів на основі класів, що може стати перешкодою для розробників, яким більш комфортно працювати з об'єктно-орієнтованим програмуванням (ООП).

– Змішування шаблонів з логікою (JSX) може збити з пантелику деяких розробників при перших знайомствах з React [11].

3.3. Використані технічні засоби

Для користувача є важливим мати систему, спроможну запускати та працювати з сучасними браузерами, тож необхідно мати такі мінімальні параметри ПК :

- ЦП [CPU]: від core i3.
- Відеоадаптер [GPU]: 3D адаптер nVidia, Intel, AMD/ATI.
- Оперативна пам’ять [RAM]: 4 ГБ;
- Відеопам’ять [VRAM]: 128 МБ.
- Накопічувач [HDD]: 2 ГБ.

3.4 Опис автоматизованої системи постановки діагнозу

В розробленій системі є декілька вікон, які генеруються в залежності від відповіді на попередні питання, коли система знає, який можливий діагноз – генерація вікон припиняється та людина отримує агреговані данні своїх відповідей та можливий діагноз.

Спочатку необхідно ввести вік. Потім написати наявні симптоми через кому, якщо в передбаченому мною списку симптомів симптом буде в тому ж виді як ввів користувач, то система його розпарсить автоматично та покаже вікно з розпізнаними симптомами, які можна підтвердити та доповнити. Далі система буде ставити питання стосовно обраних симптомів і визначати вірогідність наявності якоїсь з хвороб, якщо ж жодна хвороба наявна в базі знань не відповідає симптомам, то система запропонує звернутись до терапевта, якщо під симптоми підпадають декілька хвороб то покаже всі з ймовірностю для кожної з них. Якщо лише одна, то лише одну з рекомендацією лікаря та рекомендацією питань, які було б добре поставити лікарю.

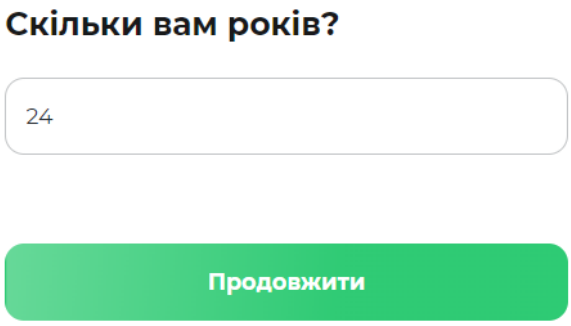


Рис. 3.4.1 Вікно вводу віку

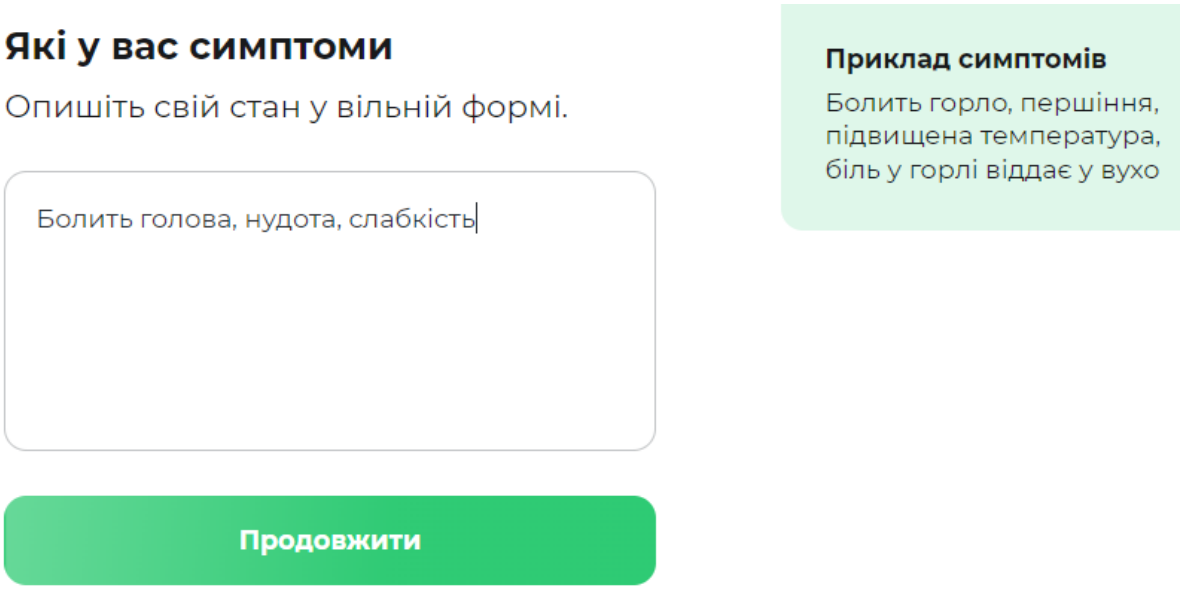


Рис. 3.4.2 Вікно вводу симптомів

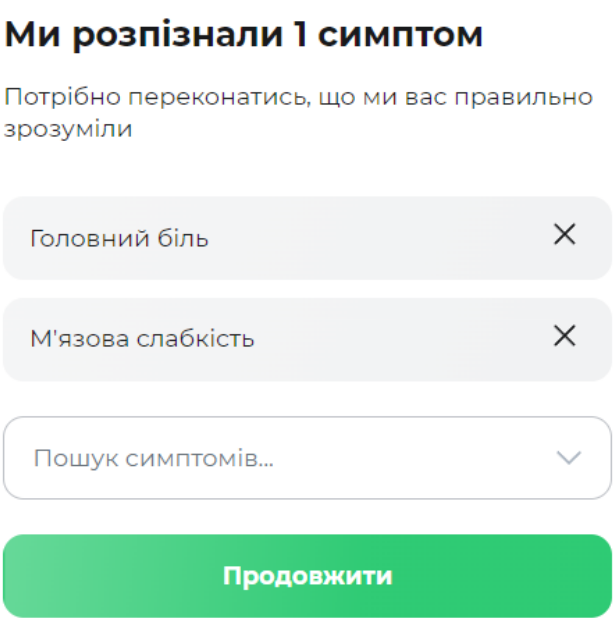


Рис. 3.4.3 Вікно уточнення симптомів

Рис. 3.4.4 Вікно уточнення симптомів

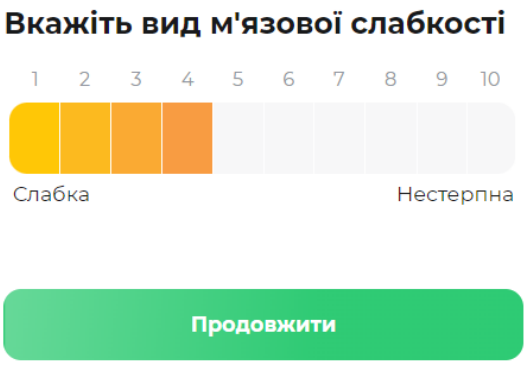


Рис. 3.4.5 Вікно уточнення сили прояву симптому

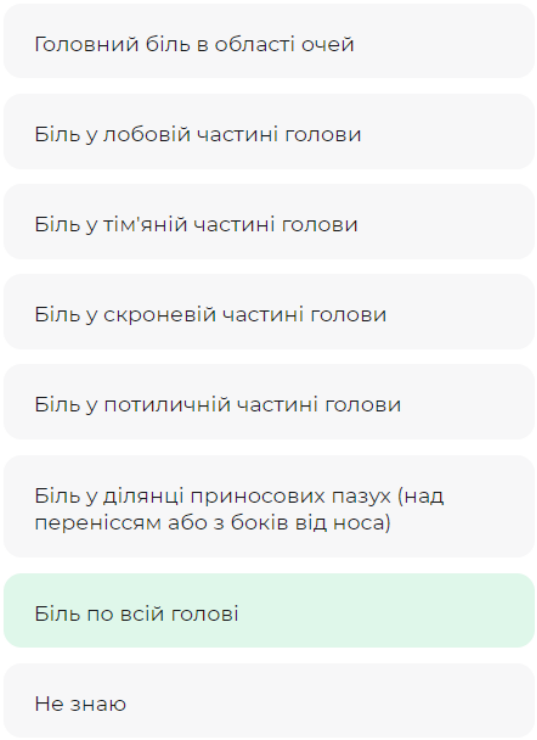


Рис. 3.4.6 Вікно уточнення типу головного болю

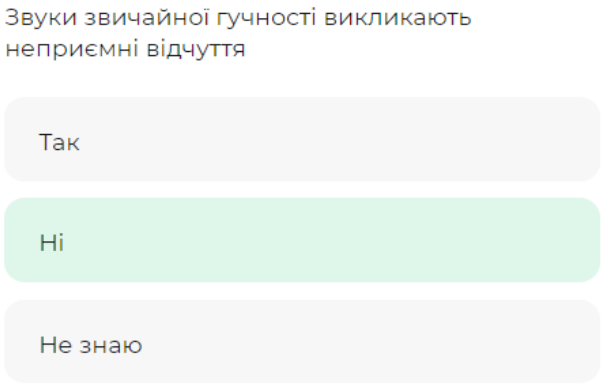


Рис. 3.4.7 Вікно уточнення

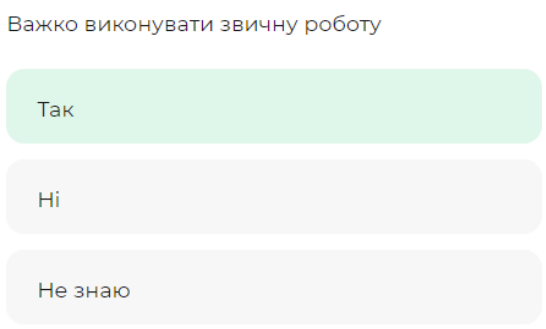


Рис. 3.4.8 Вікно уточнення

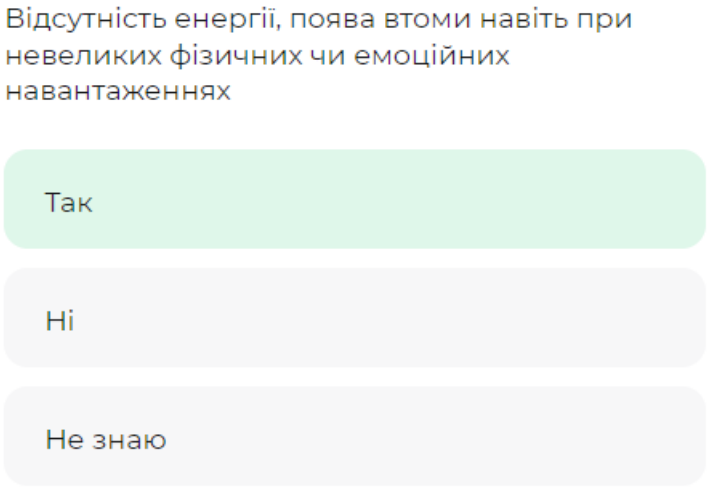


Рис. 3.4.9 Вікно уточнення

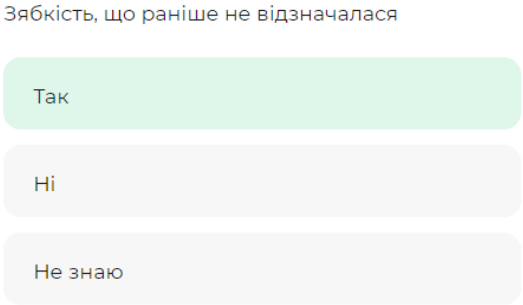


Рис. 3.4.10 Вікно уточнення

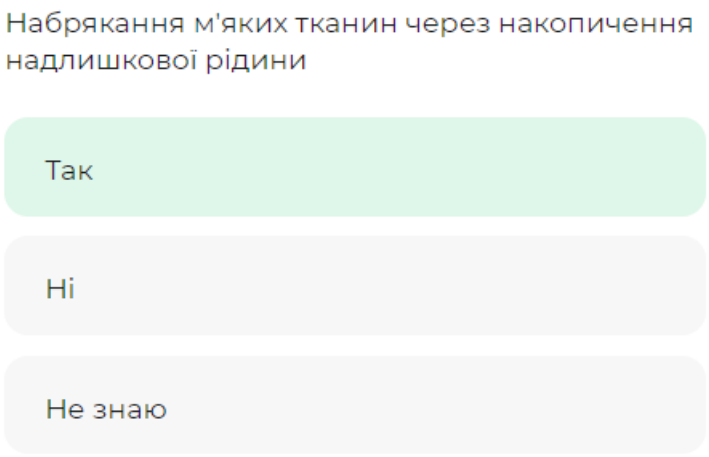


Рис. 3.4.11 Вікно уточнення

Рис. 3.4.12 Вікно уточнення

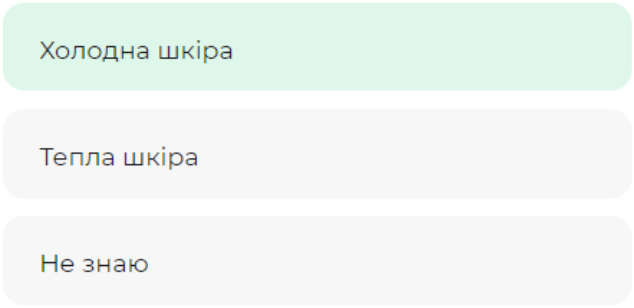


Рис. 3.4.13 Вікно уточнення

Список ваших відповідей

Інформація про пацієнта

- Вік: 24
- Стать: Жіноча стать

Підтверджені спостереження

- Головний біль
- М'язова слабкість
- Зниження працездатності
- Слабкість, швидка стомлюваність
- Підвищена чутливість до холоду
- Набряки
- Набряки на обличчі

Відкидані спостереження

- Зниження сили м'язів ніг, не пов'язане з попереднім фізичним навантаженням
- Слабкість м'язів обличчя
- Лицьовий біль
- Головний біль з'являється або посилюється при дії шуму
- Підвищена чутливість до звуків
- Нежить
- Зміни артеріального тиску
- Біль у грудній клітці
- Відчуття постійної втоми
- Випадання волосся на голові чи тілі
- Головний біль з'являється або посилюється при дії світла
- Мігрень у найближчих родичів
- Головний біль з'являється або посилюється після вживання алкоголю

Рис. 3.4.14 Вікно агрегованих відповідей

Синдром хронічної втоми - 97%
Постійне почуття втоми та перевтоми, занепад сил, що не минає навіть після тривалого відпочинку.

У 9 із 10 осіб із такими ж симптомами було діагностовано цей стан;
діагностику та лікування проводить лікар-невролог;
якщо підозрюється синдром хронічної втоми – не потрібні екстрені заходи, заплануйте візит до лікаря найближчим часом.
Я підозрюю, що маю синдром хронічної втоми, що робити?
1. Дотримуватися рекомендацій
Харчуватися різноманітно;
Додати до раціону більше фруктів та овочів;
Нормалізувати режим сну.
2. Записатися до лікаря-невролога
Які запитання поставити лікарю?
Що може спричинити синдром хронічної втоми?
Які методи лікування?
Які є немедикаментозні методи лікування?
Як змінити спосіб життя?

Рис. 3.4.15 Результат аналізу

Хвороби, які може визначити DiagnosticYouTool описано на рисунках 3.4.16-3.4.32. Наразі він обмежений, але в майбутньому може бути розширений всіма хворобами з клінічного протоколу та рекомендаціями щодо їх лікування від лікарів.

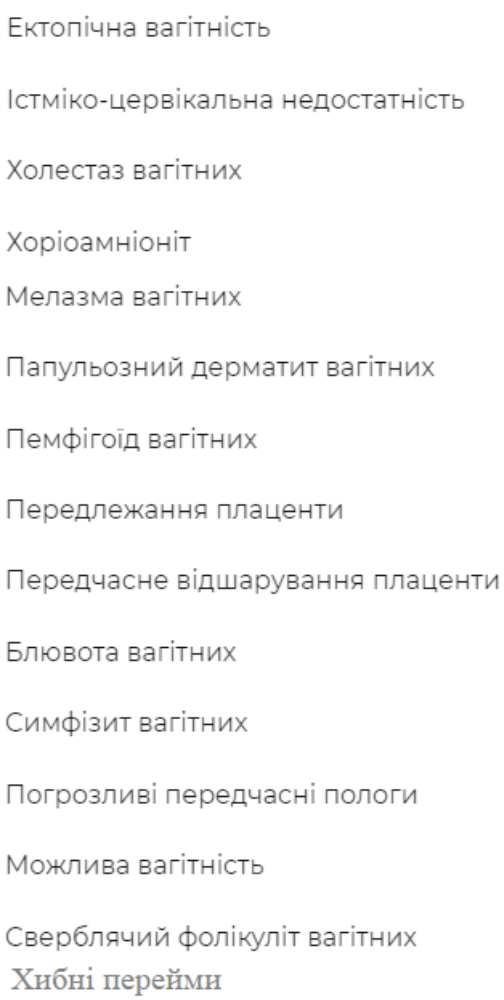


Рис. 3.4.16 Список хвороб пов’язаних з вагітністю

дисменорея

цервіцит

Ендометріоз

Гіперандрогенія у жінок

Мастит

Новоутворення молочної залози

Синдром полікістозних яєчників

вагініт

Запальні захворювання органів малого тазу

Рис. 3.4.17 Список жіночих хвороб

Баланіт

Баланопостит

Хвороба Пейроні

Епідідиміт

еректильна дисфункція

Фімоз

Гідроціль

Гіпогонадизм

Коротка вуздечка статевого члена

Новоутворення передміхурової залози

Новоутворення яєчка

Орхіт

Парафімоз

Перекрут яєчка

Перелом статевого члена

Пріапізм

Варикоцеле

Простатит

Сперматоцеле

Рис. 3.4.18 Список чоловічих хвороб

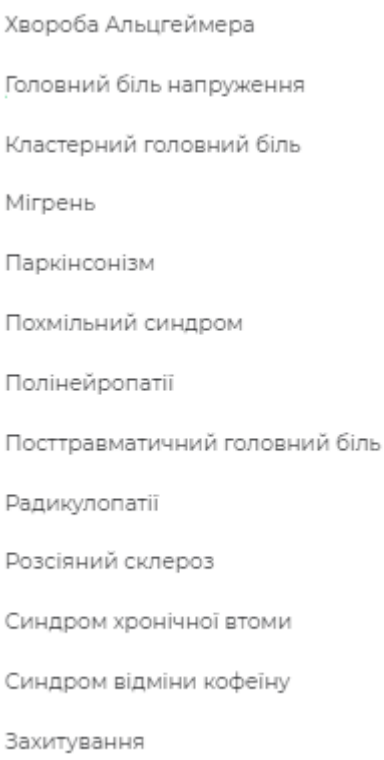


Рис. 3.4.19 Список хвороб нервової системи

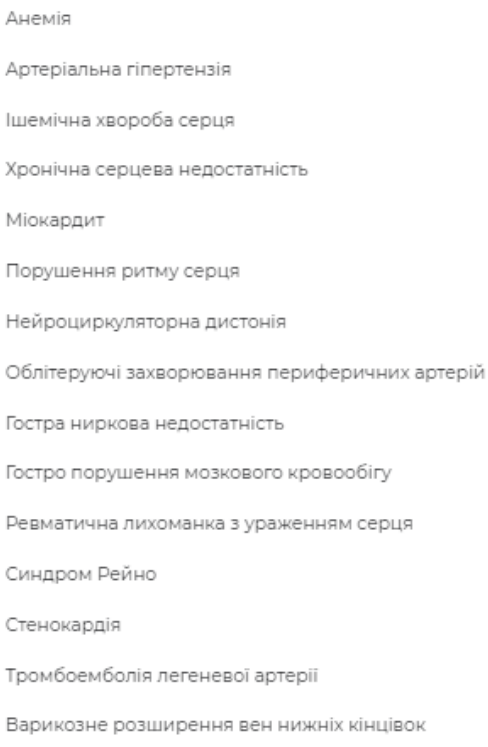


Рис. 3.4.20 Список серцево-судинних хвороб

Перелом великого пальця стопи

Перелом дистальних кінців променевої та ліктьової кістки

Перелом гомілки

Перелом кісточки

Перелом надколінка

Перелом у ділянці плечового суглоба

Перелом пальця стопи

Перелом шийки бедра

Переломи

Розтягнення зв'язок великого пальця стопи

Рис. 3.4.21 Список травм та забоїв

Хвороба Меньєра

Лабіринтит

ларингіт

Мастоїдит

Зовнішній отит

Отосклероз

Сірчана пробка

Синдром обструктивного нічного апное

Синусіт

Зниження слуху, пов'язане з впливом шуму

Середній отит

Тонзиліт

Вазомоторний риніт

Рис. 3.4.22 Список хвороб отолорингології

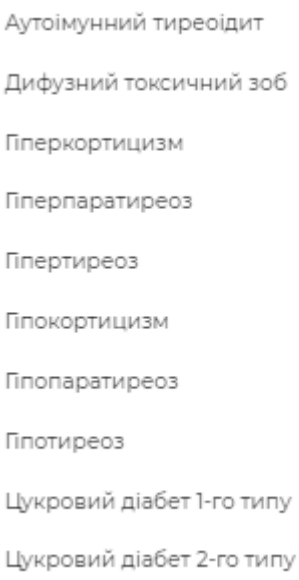


Рис. 3.4.23 Список ендокринологічних хвороб

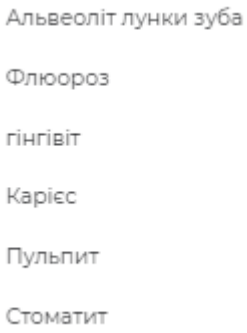


Рис. 3.4.24 Список стоматологічних хвороб

апендицит

Ціліакія

Функціональна диспепсія

Гастроезофагеальна рефлюксна хвороба (ГЕРХ)

Геморой

Хронічний панкреатит

Кишкова непрохідність

Лактазна недостатність

Новоутворення товстої кишки

Гострий панкреатит

Синдром роздратованого кишечника

Неспецифічні запальні захворювання кишок

Виразка шлунка або дванадцятипалої кишки

Рис. 3.4.25 Список хвороб шлунково-кишкового тракту

Коронавірусна інфекція

дифтерія

Грип

Кишкова інфекція

Кір

Краснуха

Менінгіт

Оперезуючий герпес

Скарлатина

Вітряна віспа

Рис. 3.4.26 Список інфекційних хвороб

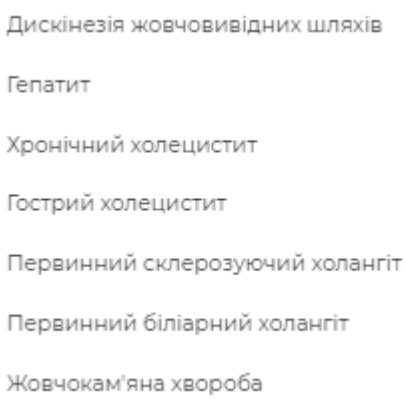


Рис. 3.4.27 Список хвороб печінки та жовчного міхура

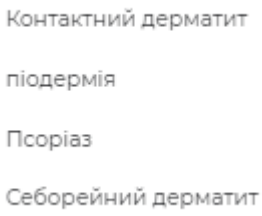


Рис. 3.4.28 Список хвороб шкіри

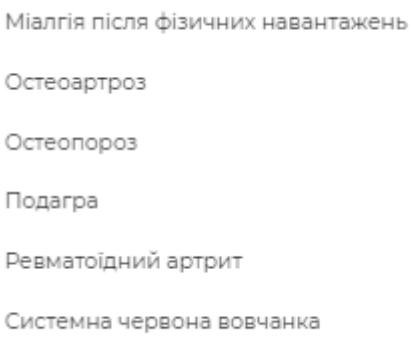


Рис. 3.4.29 Список хвороб опорного апарату

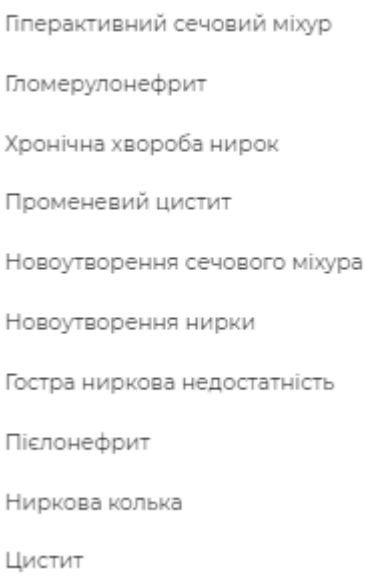


Рис. 3.4.30 Список хвороб нирок

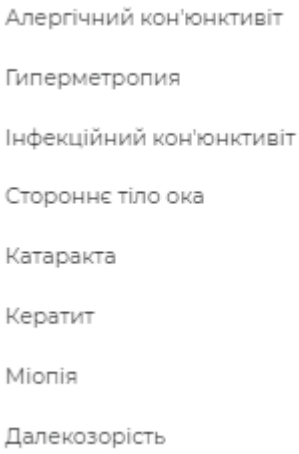


Рис. 3.4.31 Список хвороб очей

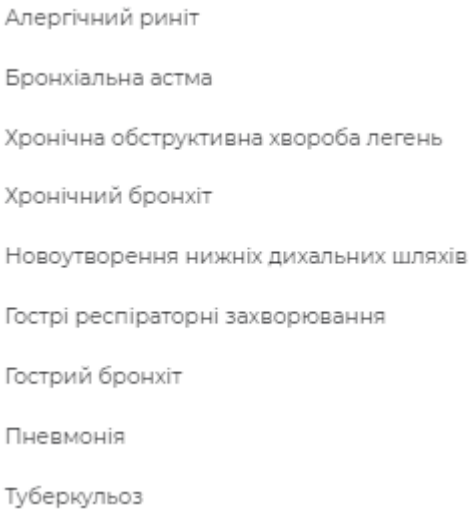


Рис. 3.4.32 Список хвороб органів дихання

ВИСНОВКИ

Беручи до уваги кризисну ситуацію у світі та небажання людей зайвий час перебувати в людних місцях, а також стреси, втому, погану екологію, люди починають відчувати якесь недомагання. Для того щоб оберегти їх від паніки під час простого гугління симптомів та іпохондрії, а також від відкладання лікування на потім було розроблено додаток DiagnosticYouTool, що допоможе швидко поставити вірогідний діагноз, підкаже до якого доктора слід звернутись, та які питання краще задати.

Практичне значення кваліфікаційної роботи полягає у розробці застосунку аналізу симптомів та постановки діагнозу, що надає можливість власноруч пройти тест та дізнатись до якого спеціаліста краще сходити або спростити та пришвидшити прийом лікаря відметаючи можливість забути якусь рідку хворобу та людський фактор з можливістю помилитись.

Під час виконання данної кваліфікаційної роботи були виконані наступні задачі:

– проаналізовано предметну галузь задачі, що розв’язується зі збором відомостей та постановкою задачі;

– проведено порівняння можливостей існуючих подібних застосунків;

– сформовано опис вимог до функціональних характеристик, інформаційної безпеки, до складу та параметрів технічних засобів, до інформаційної та програмної сумісності додатку;

– обрано технології, раціональну структуру та параметри програми;

– описано явні та неявні вимоги;

– написано програмний код веб-додатку;

– розроблено рекомендації щодо використання програми.

Базуючись на теорії щодо класифікації нейронних мереж було зроблено висновок, що нам найкраще підходять моделі на основі уваги, а також Марківське моделювання, тому програма реалізована на мові python з

використанням autoML та kernel. Клієнтська частина була розроблена з використанням ReactJS.

В подальшому застосунок може бути розширений додатковим функціоналом наприклад можна додати ролі користувачів і для лікарів додати протоколи лікування тих чи інших хвороб.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. МЕТОДИЧНІ РЕКОМЕНДАЦІЇ до виконання кваліфікаційних робіт здобувачами другого (магістерського) рівня вищої освіти спеціальностей 121 «Інженерія програмного забезпечення» / / Б.І. Мороз, О.В. Іванченко, О.В. Реута, О.С. Шевцова; М-во освіти і науки України, Нац. техн. ун-т “Дніпровська політехніка”. – Дніпро : НТУ «ДП», 2021. – 56 с.

2. URL: https://livingfo.com/shcho-take-nejronni-merezhi-ta-iak-vony-pratsiuiut/, дата звернення 18.10.2022

3. URL:https://dl.nure.ua/pluginfile.php/634/mod_resource/content/2/content/content2.html, дата звернення 18.10.2022

4. URL: https://tdmuv.com/kafedra/internal/upr_ekon/classes_stud/uk/pharm/tpkz/ptn/%D0%A4%D0%B0%D1%80%D0%BC%D0%B0%D0%BA%D0%BE%D0%B5%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D1%96%D0%BA%D0%B0/5/2%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D1%96%20%D0%B4%D0%B6%D0%B5%D1%80%D0%B5%D0%BB%D0%B0%20%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%97.htm дата звернення 12.12.2022

5. URL: https://uk.wikipedia.org/wiki/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B0_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0, дата звернення 18.10.2022

6. URL: https://uk.wikipedia.org/wiki/%D0%86%D0%BC%D0%BE%D0%B2%D1%96%D1%80%D0%BD%D1%96%D1%81%D0%BD%D0%B8%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4, дата звернення 18.10.2022

7. URL: https://metody_i_metodologiya_klinich_diagnostiki, дата звернення 18.10.2022

8. URL: https://uk.education-wiki.com/4761295-classification-of-neural-network, дата звернення 18.10.2022

9. URL:https://aiconference.com.ua/uk/news/primenenie-iskusstvennogo-intellekta-v-meditsine-effektivnaya-diagnostika-i-sozdanie-novih-lekarstv-92604#:~:text=%D0%97%D0%B0%D1%80%D0%B0%D0%B7%20%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97%20%D1%88%D1%82%D1%83%D1%87%D0%BD%D0%BE%D0%B3%D0%BE%20%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82%D1%83%20%D0%B2%D0%B8%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D0%BE%D0%B2%D1%83%D1%8E%D1%82%D1%8C%D1%81%D1%8F,%D0%B0%20%D1%82%D0%B0%D0%BA%D0%BE%D0%B6%20%D0%BF%D1%80%D0%B8%D0%B9%D0%BC%D0%B0%D1%82%D0%B8%20%D1%80%D1%96%D1%88%D0%B5%D0%BD%D0%BD%D1%8F%20%D1%81%D0%B0%D0%BC%D0%BE%D1%81%D1%82%D1%96%D0%B9%D0%BD%D0%BE., дата звернення 18.10.2022

10. URL: https://uadoc.zavantag.com/text/658/index-1.html?page=6, дата звернення 18.10.2022

11. URL: https://ru.wikipedia.org/wiki/React, дата звернення 18.10.2022

12. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : ДСТУ 3008-95. – Чинний від 1996–01–01. – К. : Держстандарт України, 1996. – 39 с.

13. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. — Київ : Держстандарт України, 1994. – 88 с.

https://github.com/SvetlanaOvcharenko/ DiagnosticYouTool

Головний файл веб-застосунку App.js:

```
import { React, useState, useEffect } from "react";
import Header from "./components/Header/Header.js";
import Main from "./components/Main/Main.js";
import Footer from "./components/Footer/Footer.js";
import About from "./components/About/About.js";
import Profile from "./pages/Profile.js";
import { BrowserRouter as Router, Switch, Route } from "react-router-dom";

import "./AppStyle.css";

export default function App() {
  const [error, setError] = useState(null);
  const [isLoaded, setIsLoaded] = useState(false);
  const [items, setItems] = useState([]);

  useEffect(() => {
    fetch("https://jsonplaceholder.typicode.com/users/1", {
      headers: { "Content-Type": "application/json" },
      credentials: "include",
    })
      .then((res) => res.json())
      .then(
        (result) => {
          setIsLoaded(true);
          setItems(result);
        },

        (error) => {
          setIsLoaded(true);
          setError(error);
        }
      );
  }, []);
  console.log(items);
  return (
```

```
    <Router>
      <>
        <Header />
        <Switch>
          <Route path="/profile" component={Profile} />
          <Route path="/about" component={About} />
          <Route path="/" exact component={Main} />
        </Switch>
        <Footer />
      </>
    </Router>
  );
}
```

Файл для обміну даними main.json:

```
{
  "products": [
    {
      "id": "1",
      "title": "GIGI",
      "description": "Грязевая маска для лица GIGI SOLAR ENERGY Mud mask for oil skin
 75 ml",
      "img": "https://images.ua.prom.st/2459027579_w640_h640_gryazevaya-maska-
dlya.jpg",
      "price": "812,00 грн."
    },
    {
      "id": "2",
      "title": "CO2",
      "description": "Маска-
активатор карбокситерапия Esthetic House CO2 Esthetic Formula Carbonic Mask",
      "img": "https://images.ua.prom.st/1833584142_w640_h640_maska-aktivator-
karboksiterapiya-esthetic.jpg",
      "price": "990,00 грн."
    },
    {
      "id": "3",
      "title": "MEDI-PEEL",
      "description": "Успокаивающая и увлажняющая маска с азуленом MEDI-
PEEL Azulene Water Calming Mask",
```

```
      "img": "https://images.ua.prom.st/2675951330_w640_h640_uspokaivayuschaya-i-
uvlazhnyayuschaya.jpg",
      "price": "539,99 грн."
    },
    {
      "id": "4",
      "title": "PURITO BHA",
      "description": "Увлажняющий ночной гель с кислотами с экстрактом чайного дерева
 PURITO BHA Dead Skin Moisture Gel",
      "img": "https://images.ua.prom.st/1989801352_w640_h640_uvlazhnyayuschij-
nochnoj-gel.jpg",
      "price": "180,00 грн."
    },


    {
      "id": "5",
      "title": "BLITHE",
      "description": "Сплэш-
маска успокаивающая BLITHE Patting Splash Mask Soothing & Healing Green Tea 150 мл Ко
реят",
      "img": "https://images.ua.prom.st/2307826372_w640_h640_splesh-maska-
uspokaivayuschaya-blithe.jpg",
      "price": "650,00 грн."
    }
  ]
}
```

Файл форми реєстрації login.js:

```
import { React, useState, SyntheticEvent } from "react";
import { Redirect } from "react-router-dom";
import {
  Nav,
  Form,
  Button,
  Navbar,
  Modal,
  Row,
  Col,
  Container,
  Image,
} from "react-bootstrap";
```

```
export default function Login(props) {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");

  const submit = (e) => {
    e.preventDefault();
    fetch("", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      credentials: "include",
      body: JSON.stringify({
        email,
        password,
      }),
    });
    //setRedirect(true);
    props.onHide();
  };

  return (
    <>
      <Modal centered show={props.show} onHide={props.onHide}>
        <Modal.Header closeButton>
          <Modal.Title className=" pl-4 ml-auto">Sign in</Modal.Title>
        </Modal.Header>
        <Modal.Body>
          <Form onSubmit={submit}>
            <Form.Group controlId="formLoginEmail">
              <Form.Control
                type="email"
                placeholder="Email address"
                onChange={(e) => setEmail(e.target.value)}
              />
            </Form.Group>
            <Form.Group controlId="formLoginPassword">
              <Form.Control
                type="password"
                placeholder="Repeat password"
                onChange={(e) => setPassword(e.target.value)}
```

```
              />
            </Form.Group>

            <Container className="d-flex justify-content-center">
              <Button variant="primary" size="lg" type="submit">
                Login
              </Button>
            </Container>
          </Form>
        </Modal.Body>
      </Modal>
    </>
  );
}
```

Головний файл сторінки профіля Profile.js:

```
import React from "react";
import App from "../App";
import CategorList from "../Categor/CategorList";
import CategorListAdd from "../Categor/CategorListAdd";
import Context from "../context";
import Product from "../Categor/Product";
import { Modal, Button } from "react-bootstrap";
import { Container } from "react-bootstrap";

export const Profile = () => {
  const [modalShow, setModalShow] = React.useState(false);
  const [categors, setCategors] = React.useState([
    {
      id: 1,
      title: "Компютери та ноутбуки",
      subcategory: "Ноутбуки",
      links: "http://htmlbook.ru/html/a",
    },
    {
      id: 2,
      title: "Мобільні і звязок",
      subcategory: "Телефони",
      links: "http://htmlbook.ru/html/li",
```

```
  },
  {
    id: 3,
    title: "Побутова техніка",
    subcategory: "Техніка для кухні",
    links: "ttt",
  },
]);

const [categorsAdd, setCategorsAdd] = React.useState([
  {
    id: 1,
    title: "Техніка",
    subcategory: "Ноутбуки",
    links: "http://htmlbook.ru/html/a",
  },
  {
    id: 2,
    title: "Техніка",
    subcategory: "Телефони",
    links: "http://htmlbook.ru/html/li",
  },
  { id: 3, title: "Кухня", subcategory: "Техніка для кухні", links: "ttt" },
  {
    id: 4,
    title: "Компютери та ноутбуки",
    subcategory: "Техніка для кухні",
    links: "ttt",
  },
]);

function MyVerticallyCenteredModal(props) {
  return (
    <Modal
      {...props}
      size="lg"
      aria-labelledby="contained-modal-title-vcenter"
      centered
    >
```

```
      <Modal.Header closeButton>
        <Modal.Title id="contained-modal-title-vcenter">
          Add categors
        </Modal.Title>
      </Modal.Header>
      <Modal.Body>
        <Context.Provider value={{ removeCategorAdd }}>
          {categors.length ? (
            <CategorListAdd categorsAdd={categorsAdd} />
          ) : (
            <p>Categories not added !!!!</p>
          )
          }

        </Context.Provider>
      </Modal.Body>
      <Modal.Footer>
        <Button onClick={props.onHide}>Close</Button>
      </Modal.Footer>
    </Modal>
  );
}
function removeCategor(id) {
  setCategors(categors.filter((categor) => categor.id !== id));
}
function removeCategorAdd(id, links) {
  setCategorsAdd(categorsAdd.filter((categor) => categor.id == id));
  if (categorsAdd.length == 1)  setCategors([...categors, ...categorsAdd]);
}

return (
  <Container fluid style={{paddingTop:'15px'}}>
  <div>
    <MyVerticallyCenteredModal
      show={modalShow}
      onHide={() => setModalShow(false)}
    />
    <div style={{ paddingBottom: "15px" }}>
      <p style={{ textAlign: "center" }}>
```

```
          <img
            style={{ width: "160px", height: "160px", borderRadius: "80px" }}
            src="https://images.unsplash.com/photo-1554555819-
f722cb0c01c5?ixid=MXwxMjA3fDB8MHxzZWFyY2h8ODN8fHBlcnNvbnxlbnwwfHwwfA%3D%3D&ixlib=rb-
1.2.1&auto=format&fit=crop&w=500&q=60"
          />
        </p>

        <h5 style={{ textAlign: "center", margin: "-0.5rem 1rem 0 0" }}>
          User name
        </h5>
      </div>

      <Context.Provider value={{ removeCategor }}>
        <div>
          <h4>
            Categories
            <button
              className="rn"
              variant="primary"
              onClick={() => setModalShow(true)}
            ></button>
          </h4>
          {categors.length ? (
            <CategorList categors={categors} />
          ) : (
            <p>Categories not added !!!!</p>
          )}
        </div>
      </Context.Provider>
      <div>
        <h4>Products</h4>
        <Product />
      </div>
    </div>
    </Container>
  );
};
export default Profile;
```

Файл головної сторінки товарів MainProduct.js:

```
import { React, useEffect, useState } from "react";
import { Card, Container, Row, Col } from "react-bootstrap";
import items from "../db/main.json";


export default function MainProduct() {
  return (
    <Container fluid>
      <Row className="">
        {items.products.map((item) => (
          <Card
            as={Col}
            md={4}
            lg={3}
            sm={6}
            key={item.id}
            className="p-0 text-center col-xl"
            style={{}}
          >
            <Card.Link href="#">
              <Card.Img variant="top" src={item.img} />
            </Card.Link>
            <Card.Body>
              <Card.Link href="#">
                <Card.Title style={BoldText}>{item.title}</Card.Title>
              </Card.Link>
              <Card.Text className="pt-0 pb-0" style={{ fontSize: "12px" }}>
                {item.description}
              </Card.Text>
            </Card.Body>
            <Card.Link href="#">
              <p style={BoldText}>{item.price}</p>
            </Card.Link>
          </Card>
        ))}
      </Row>
    </Container>
```

```
  );
}
const BoldText = { fontSize: "14px", fontWeight: "bold", color: "#000" };
```

Файл з описом стилів AppStyle.css:

```
button.d-sm-none.navbar-toggler.collapsed img {
  transform: rotate(-90deg);
  transition: 0.5s;
}
button.d-sm-none.navbar-toggler img {
  transform: rotate(90deg);
  transition: 0.5s;
}
.page-link {
  text-shadow: 1px 1px 2px white, 0 1px 2em white, 0 0 0.5em white;
  color: white;
  background-color: #4ca312;
  color: beige;
}
.page-link:hover {
  background-color: #b14713;
  color: white;
}
.card-body {
  padding: 0;
}
.pagination {
  display: flex;
  flex-wrap: wrap;
  max-width: max-content;
  padding: 5px 10px;
  margin: 10px auto;
  border-radius: 0;
  background-color: #eff0f6;
}
.page-item {
  margin: 5px;
}
.next .page-link,
.previous .page-link {
  text-shadow: 1px 1px 2px white, 0 1px 2em white, 0 0 0.5em white;
  color: #4fa517;
  background-color: #eff0f6;
  border: 0;
}
.next .page-link:hover,
.previous .page-link:hover {
  background-color: #b14713;
  color: white;
}
@media (max-width: 450px) {
  .modal-dialog {
    padding-right: 0rem;
  }
}
@media (min-width: 1200px) {
  .col-xl {
```

```
    flex: 0 0 12.5%;
    max-width: 12.5%;
  }
}
@media (max-width: 600px) {
  .page-item {
    margin: 0;
    padding: auto;
  }
}
.nav-item > a {
  color: rgb(83, 83, 83);
  text-decoration: none;
}
.nav-item > a:hover {
  color: rgb(49, 49, 49);
  text-decoration: none;
}
```

Головний файл додатку – App.js:

```
import React from 'react';
import { StatusBar } from 'react-native';
import { enableScreens } from 'react-native-screens';
import { NavigationContainer } from '@react-navigation/native';
import AppNavigator from './client/navigators/AppNavigator';

export default function App() {
	enableScreens(); // Оптимизация памяти в React Navigation
	window.navigator.userAgent = 'ReactNative';

	return (
		<NavigationContainer>
			<StatusBar barStyle='light-content' backgroundColor='#000'/>
			<AppNavigator/>
		</NavigationContainer>
	);
};
```

Файл компоненту кнопки – CustomButton.js:

```
import React from 'react';
import { TouchableOpacity, Text, StyleSheet } from 'react-native';

export default function CustomButton({ ...props }) {
    const { children, onPress, backgroundColor, textColor, fontWeight, borderWidth, borderColor, borderRadius } = props;
```

```
    const btnStyles = {
        ...styles.btn,
        backgroundColor: backgroundColor,
        borderWidth: borderWidth || 0,
        borderColor: borderColor || '',
        borderRadius: borderRadius || 0
    };
    const btnTextStyles = {
        ...styles.btnText,
        color: textColor,
        fontWeight: fontWeight || '300'
    };

    return (
        <TouchableOpacity
            style={btnStyles}
            onPress={onPress}
            activeOpacity={.7}
        >
            <Text style={btnTextStyles}>{children}</Text>
        </TouchableOpacity>
    );
}

const styles = StyleSheet.create({
    btn: {
        flex: 1,
        borderRadius: 7,
        justifyContent: 'center'
    },
    btnText: {
        textAlign: 'center',
        fontSize: 18,
        textTransform: 'uppercase',
    }
});
```

Файл компонента текстового поля – CustomInput.js:

```
import React, { useState } from 'react';
```

```
import { View, TextInput, TouchableOpacity, Image, StyleSheet } from 'react-native';

export default function CustomInput({ ...props }) {
    const [active, setActive] = useState(false);
    const { value, setValue, borderWidth, borderColor, borderRadius, activeBorderColor, placeholder,
        secureTextEntry = false, keyboardType = 'default', searching = false, onPressIcon = () => {} } = props;

    const inputStyles = {
        ...styles.input,
        borderWidth: borderWidth || 0,
        borderRadius: borderRadius || 0,
        borderColor: borderColor || '#000',
        paddingRight: searching ? 55 : styles.input.paddingHorizontal
    };
    const activeInputStyles = {
        borderColor: activeBorderColor || '#000',
        borderWidth: 3
    }

    const onFocusHandler = () => {
        setActive(true);
    }
    const onBlurHandler = () => {
        setActive(false);
    }

    return (
        <View style={styles.container}>
            <TextInput
                style={[inputStyles, active && activeInputStyles]}
                value={value}
                onChangeText={text => setValue(text)}
                placeholder={placeholder || ''}
                secureTextEntry={secureTextEntry}
                keyboardType={keyboardType}
                onFocus={onFocusHandler}
                onBlur={onBlurHandler}
            />
```

```
            { searching &&
            <TouchableOpacity
                style={styles.iconContainer}
                activeOpacity={.7}
                onPress={onPressIcon}
            >
                <Image
                    source={require('../img/search-icon.png')}
                    style={styles.icon}
                />
            </TouchableOpacity> }
        </View>
    );
}

const styles = StyleSheet.create({
    container: {
        position: 'relative',
        flex: 1
    },
    input: {
        flex: 1,
        paddingHorizontal: 12,
        fontSize: 15
    },
    iconContainer: {
        position: 'absolute',
        right: 10,
        top: 12,
    },
    icon: {
        width: 22,
        height: 22,
        tintColor: '#aaa'
    }
});
```

Файл компоненту аватару – Avatar.js:

```
import React from 'react';
```

```
import { View, Image, StyleSheet } from 'react-native';

export default function Avatar({ src, width, height }) {
    const containerStyles = {
        ...styles.container,
        width: width,
        height: height
    };

    return (
        <View style={containerStyles}>
            <Image
                style={styles.img}
                source={src ? {uri: src} : require('../img/user-thumb.png')}
            />
        </View>
    )
}

const styles = StyleSheet.create({
    container: {
        borderRadius: 50,
    },
    img: {
        width: '100%',
        height: '100%'
    }
});
```

Файл головної сторінки – HomeScreen.js:

```
import React, { useState, useEffect } from 'react';

import { View, ScrollView, StyleSheet } from 'react-native';

import { Header, Content } from '../components/Home';

import getSeparationDataRows from '../logic/getSeparationDataRows';


export default function HomeScreen({ navigation }) {

    const [products, setProducts] = useState([]);


    useEffect(() => {

        function fetchData() {

            const data = require('../db/main.json');


            setProducts(getSeparationDataRows(170, data['products']));

        }


        fetchData();

    }, []);
```

```
    return (
        <ScrollView>
            <View style={styles.container}>
                <Header/>
                <Content products={products} navigation={navigation}/>
            </View>
        </ScrollView>
    );
}

const styles = StyleSheet.create({
    container: {
        backgroundColor: '#fff',
        flex: 1,
        padding: 15
    }
});
```

Файл контенту головної сторінки – Content.js:

```
import React from 'react';
import { View, Text, StyleSheet } from 'react-native';
import ProductsList from '../Products/ProductsList';
import CustomButton from '../CustomButton';

export default function Content({ products, navigation }) {
    return (
        <View style={styles.container}>
            <View style={styles.headerContainer}>
                <Text style={styles.titleText}>Best price for today</Text>
            </View>

            <ProductsList products={products} navigation={navigation}/>

            <View style={styles.btnContainer}>
                <CustomButton
                    onPress={() => {console.log('Test')}}
                    borderWidth={2}
                    borderRadius={7}
                    fontWeight={'700'}
```

```
                    textColor={'#000'}
                >See more</CustomButton>
            </View>

        </View>
    );
}

const styles = StyleSheet.create({
    container: {
        flex: 1,
        marginTop: 30
    },
    headerContainer: {
        marginBottom: 15
    },
    titleText: {
        fontSize: 16,
        textTransform: 'uppercase',
        fontWeight: '700'
    },
    btnContainer: {
        marginTop: 20,
        height: 50
    }
});
```

Файл компонента товару – ProductItem.js:

```
import React from 'react';
import { View, Text, Image, TouchableOpacity, StyleSheet, Dimensions } from 'react-native';

const getCountItemsInLine = () => {
    const deviceWidth = Dimensions.get('window').width;
    const itemWidth = 170;

    return Math.floor(deviceWidth / itemWidth);
}

export default function ProductItem({ item, position, navigation }) {
    const itemsInLine = getCountItemsInLine(); // Count items in line
```

```
    const notLastInLine = position % itemsInLine !== 0 && itemsInLine !== 1;
         // item is not on the right side products
    const description = item['description'].length > 65
       ? item['description'].slice(0, 65) + '...' : item['description'];

    return (
        <View style={[styles.container, notLastInLine && { borderRightWidth: 2 }]}>
            <TouchableOpacity
                style={{flex: 1}}
                activeOpacity={.7}
                onPress={() => navigation.push('Product', { item })}
            >
                <View style={styles.imgContainer}>
                    <Image
                        style={styles.img}
                        source={{uri: item['img']}}
                    />
                </View>

                <View style={styles.infoContainer}>
                    <Text style={styles.titleText}>{item['title']}</Text>
                    <Text style={styles.descText}>{description}</Text>

                    <View style={styles.priceContainer}>
                        <Text style={styles.priceText}>{item['price']}
                          грн.</Text>
                    </View>
                </View>
            </TouchableOpacity>
        </View>

    );
}

const styles = StyleSheet.create({
    container: {
        width: `${100 / getCountItemsInLine()}%`,
        paddingBottom: 10,
        paddingHorizontal: 5,
```

```
        borderColor: '#ddd'
    },
    imgContainer: {
        width: '100%',
        height: +`${Math.floor(280 / getCountItemsInLine())}`,
        padding: 5
    },
    img: {
        width: '100%',
        height: '100%',
        resizeMode: 'contain'
    },
    infoContainer: {
        flex: 1,
        justifyContent: 'space-between',
        alignItems: 'center',
    },
    titleText: {
        textAlign: 'center',
        fontWeight: '700',
        fontSize: 18
    },
    descText: {
        color: '#777',
        textAlign: 'center',
        fontSize: 15,
    },
    priceContainer: {
        marginTop: 5
    },
    priceText: {
        textAlign: 'center',
        fontWeight: '700',
        fontSize: 18
    }
});
```

Файл контенту продукту – Content.js:

```
import React from 'react';
import { View, Text, Image, StyleSheet } from 'react-native';
```

```
import LineGraph from '../Graphs/LineGraph';

export default function Content({ item, navigation }) {
    return (
        <View style={styles.container}>

            <View style={styles.titleContainer}>
                <Text style={styles.titleText}>{item['title']}</Text>
            </View>

            <View style={styles.productImageContainer}>
                <Image
                    style={styles.productImage}
                    source={{uri: item['img']}}
                    resizeMode='contain'
                />
            </View>

            <View style={styles.graphContainer}>
                <LineGraph/>
            </View>

            <View style={styles.infoContainer}>
                <Text style={styles.priceText}>{item['price']} грн.</Text>
                <Text style={styles.linkText}>{item['link']}</Text>
            </View>
        </View>
    );
}

const styles = StyleSheet.create({
    container: {
        flex: 1
    },
    titleContainer: {
        marginTop: 10
    },
    titleText: {
        fontSize: 20,
```

```
        fontWeight: '700'
    },
    productImageContainer: {
        width: '100%',
        height: 250,
    },
    productImage: {
        width: '100%',
        height: '100%'
    },
    graphContainer: {
        height: 250,
        backgroundColor: 'green'
    },
    priceText: {
        fontSize: 18,
        fontWeight: '700'
    },
    linkText: {
        color: 'blue',
        fontSize: 18,
    }
});
```

```
{ "Б":
[
"Блідість шкіри",
"Біль в горлі",
"Біль у грудній клітці",
"Біль у молочній залозі",
"Біль у нижніх кінцівках",
"Біль у паху",
"Біль в шиї",
"Біль у спині",
"Біль у суглобах",
"Біль у вусі",
"Біль у верхніх кінцівках",
"Біль в животі",
```

```
"Біль за вухами",
"Блювота"
]
"В":
[
"Випадання волосся",
"Вологий кашель",
"Виділення з очей",
"Виділення з вуха",
"Виділення з піхви",
"Виділення крові з піхви",
"Відчуття кома в горлі",
"Виразки на шкірі",
"Відчуття стороннього тіла або піску в очах",
"Втрата нюху"
]
"Г":
[
"Галюцинації",
"Гній у калі",
"Головний біль"
]

"Д":
[
"Діарея",
"Двоєння в очах"
]
"Ж": "Жовтяний колір шкіри",
"З":
[
"Запор",
"Запаморочення",
"Зубний біль",
"Задишка",
```

"Зміни зовнішнього вигляду шкіри",

"Зниження артеріального тиску",

"Знебарвлений кал",

"Зниження маси тіла",

"Знижений статевий потяг",

"Зменшення м'язової маси",

"Збільшення маси тіла",

"Затруднене ковтання"

]

"К":

[

"Кашель",

"Крихкість нігтів",

"Кров на туалетному папері",

"Кров у сечі",

"Кров у калі",

"Кровоточивість ясен"

]

"М":

[

"Мокрота з кров'ю",

"Мутна сеча",

"М'язовий біль"

]

"Н":

[

"Набухання вен на шиї",

"Нетримання сечі",

"Неприємний запах із рота",

"Непритомний стан",

"Набряки",

"Набряки на обличчі",

"Набряки на нижніх кінцівках",

"Набряки на верхніх кінцівках",

"Нудота"

```
]
"О":
[
"Озноб",
"Очний біль",
"Охриплість",
"Оніміння обличчя",
"Оніміння нижніх кінцівок",
"Оніміння верхніх кінцівок",
"Опущення повік"
]
"П":
[
"Перепади настрою",
"Порушення слуху",
"Почервоніння очей",
"Печія",
"Пітливість",
"Пошкодження слизової оболонки рота",
"Підвищення артеріального тиску",
"Підвищена спрага",
"Погіршення пам'яті",
"Погіршення зору",
"Почуття припливів",
"Почуття серцебиття",
"Почуття спеки"
]
"Р":
[
"Розширення вен нижніх кінцівок",
"Розтяжки на шкірі"
]
"С":
[
"Свербіння очей",
```

"Схильність до синців",

"Скутість суглобів",

"Слабкий струмінь сечі",

"Слабкість, швидка стомлюваність",

"Слабкість м'язів обличчя",

"Слиз у калі",

"Судинні сітки на ногах",

"Стріляє у вусі",

"Судоми м'язів",

"Сухий кашель"

]

"Т":

[

"Темна сеча",

"Тремор",

"Тріщини на шкірі,

]

"Ч":"Чорний кал",

"Ш":

[

"Шум в вухах",

"Шкірний свербіж"

]

}

ДОДАТОК Б

ПЕРЕЛІК ДОКУМЕНТІВ НА ОПТИЧНОМУ НОСІЇ

Ім’я файла	**Опис**
Пояснювальні документи	
дипломна робота Овчаренко СО 121м-21-1.docx	Пояснювальна записка роботи. Документ Word.
дипломна робота Овчаренко СО 121м-21-1.pdf	Пояснювальна записка роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_Овчаренко.pptx	Презентація роботи