Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Адаптивний алгоритм пошуку асоціативних правил.
(тема)

Виконав:
студент 2 курсу, групи СШІм-18-2Шкіль А.Р
(прізвище, ініціали)

Спеціальність 122 – Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми Освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва спеціалізації)

Керівник Узлов Д.Ю
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 – Комп'ютерні науки
(код і повна назва)

Тип програми Освітньо-наукова
(освітньо-проіфесійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Шкілю Артурові Руслановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Адаптивний алгоритм пошуку асоціативних правил
_____
_____
затверджена наказом університету від 30 03 20 20 р. № 480 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 05 2020 р.
3. Вихідні дані до роботи документація платформ VOIP навчальні матеріали по методам асоціативних правил
4. Перелік питань, що потрібно опрацювати в роботі: Метод навчання асоціативним правилам, технологія голосових дзвінків, проектування клієнт-серверних додатків, розробка мобільних додатків.
_____
_____
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)Рисунок 1 – Схема ініціації дзвінка через сервер маршрутизації; Рисунок 2 – Структура RTP пакету; Рисунок 3 – Схема алгоритму Apriori; Рисунок 4 – Архітектура додатку; Рисунок 5 – Схема даних

інтегрованої БД; Рисунок 6 – Схема даних центральної бази даних; Рисунок 7 – Архітектура модулю додатку; Рисунок 8 – Діаграма послідовності процесу авторизації; Рисунок 9 – Діаграма послідовності процесу реєстрації;

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		.	
Основна частина	ас. Узлов Д. Ю.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на дипломну роботу	30.03.2020	виконано
2	Аналіз предметної області і постановка задачі	01.04.2020	виконано
3	Дослідження методів асоціативного навчання	02.04.2020-10.04.2020	виконано
4	Проектування додатку	10.04.2020-15.04.2020	виконано
5	Розробка додатку	15.04.2020-24.04.2020	виконано
6	Тесування додатку	24.04.2020-25.04.2020	виконано
7	Аналіз даних, використання асоціативних правил	26.04.2020	виконано
8	Оформлення графічних матеріалів	28.04.2020	виконано
9	Оформлення пояснювальної записки	01.05.2020	виконано
10	Попередній захист	19.05.2020	виконано
11	Захист перед ЕК	20.05.2020	

Дата видачі завдання 30 03 2020 р.

Студент _____
(підпис)

Керівник роботи _____ ас. Узлов Д. Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи містить 70 с., 20 рисунків, 5 таблиць, 15 джерел, 4 додатки.

IOS, VOIP, АСОЦІАТИВНЕ НАВЧАННЯ, БАЗА ДАНИХ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, ІНТЕРНЕТ ТЕЛЕФОНІЯ, МОВНИЙ ОБМІН

Об’ктом дослідження є методи асоціативного навчання.

Метою даної атестаційної роботи є розробка мобільного застосунку на тему «вивчення іноземних мов», а також аналіз бази даних додатку одним з методів асоціативного навчання для отримання асоціативних правил, з послідуючою інтеграцією отриманих правил в додаток з метою його оптимізації.

Методи дослідження: аналіз літератури, документації, та Internet–джерел.

Було розроблено мобільний додаток, що складається з клієнтської та серверної частин. Додаток дозволяє користувачам автоматично знаходити партнера для практикування іноземної мови на основі обраної теми, а також забезпечує голосові дзвінки.

РЕФЕРАТ

Пояснительная записка к дипломной работе содержит 70 с., 20 рисунков, 5 таблиц, 15 источников, 4 приложения.

IOS, VOIP, АССОЦИАТИВНОЕ ОБУЧЕНИЕ, БАЗА ДАННЫХ, ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ, ИНТЕРНЕТ ТЕЛЕФОНИЯ, ЯЗЫКОВОЙ ОБМЕН

Объектом исследования являются методы ассоциативного обучения.

Целью данной дипломной работы является разработка мобильного приложения на тему: «практика иностранных языков», а также анализ базы данных приложения одним из методов ассоциативного обучения для получения ассоциативных правил, с последующей интеграцией полученный правил в приложение с цель его оптимизации.

Методы исследования: анализ литературы, документации, и Internet–источников.

Было разработано мобильное приложение, которое состоит из клиентской и серверной частей. Приложение позволяет пользователям автоматически находить партнера для практики иностранного языка на основе выбранной темы, а также обеспечивает голосовые звонки.

ABSTRACT

Explanatory note contains 70 pages, 20 figures, 5 tables, 15 sources, 4 applications.

ASSOCIATION RULE LEARNING, DATA, DATABASE, VOIP, IOS, MINING, LANGUAGE EXCHANGE, VOICE CALLING

The object of the research is the association rule learning methods.

The goal of the following work is to develop the «foreign languages practicing» mobile application, to analyze app's database, using an associating rule learning method for borrowing the association rules with can be then integrated into the app for some performance improvements.

Research methods: literature analysis, documentation analysis and analysis of Internet sources.

An application that consists of client and server parts had been developed. The app allows users to automatically find partners for language practicing and also allows them to use built–in voice calling functionality.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕННЬ, СИМВОЛІВ, ОДИНИЦЬ, СКРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

ОС – операційна система;

API – Application programming interface – прикладний програмний інтерфейс;

IDE – Integrated development environment –інтегроване середовище розробки;

iOS –мобільна операційна система, розроблена компанією Apple;

GAR – Genetic association rules – генетичні асоціативні правила;

RTCP – RTP control protocol – протокол контролю передачі даних в реальному часі;

RTP – Real time transport protocol – протокол передачі даних в реальному часі;

SIP – Session initiation protocol – протокол встановлення сесії;

SDK – Software development kit – набір засобів розробки.

TCP/IP – Transport control protocol/Internet protocol – протокол керування передачею/міжмережевий протокол;

UDP – User datagram protocol – протокол датаграм користувача;

VOIP – Voice over Internet protocol – технологія передачі голосу через інтернет.

ВСТУП

Методи штучного інтелекту щороку набувають все ширшого використання в багатьох галузях науки й техніки, медицині, освіті, фінансовій сфері. Все більше проблем, пов’язаних з аналізом даних, створенням моделей на основі даних вирішуються за допомогою методів штучного інтелекту.

З самого початку методи штучного інтелекту потребували певного набору даних, специфічних для конкретної предметної області. Одними з перших таких розробок були експертні системи, що часто потребували ручного введення даних експертом в заданій області. Очевидно, що такий підхід мав значні обмеження, адже врешті–решт, на виході такої системи можна отримати лише те, що вже було задано на вхід.

Подальший розвиток інформаційних технологій, ріст мережі інтернет сприяли стрімкому розвитку різноманітних додатків, сервісів, загальною характерною рисою яких є збір великої кількості інформації в процесі своєї роботи. Аналіз таких масивів даних, виявлення в них прихованих закономірностей поставили нові задачі, для вирішення яких були необхідні нові підходи до методів штучного інтелекту.

Одним з розділів штучного інтелекту, що забезпечив нові можливості при аналізі великох кількісті даних є добування даних (англ. Data Mining), також відомий як знаходження знань в базах даних. За допомогою методів добування даних можна автоматично знаходити приховані закономірності в різноманітних базах даних. Такі закономірності можуть бути корисними для подальшої оптимізації тих процесів, в результаті яких було накопичені аналізовані ці дані. Добування даних – це комплексна область, що поєднує методи статистичного аналізу, машинного навчання, штучних нейронних мереж.

Наразі існує багато предметних областей, яким пратаманна присутність великих наборів даних, до яких можна примінити методи добування даних. Це може бути база даних транзакцій гіпермаркету, чи база даних всіх зіграних коли–небудь матчів в шахи, або база даних, що містить інформацію про користувачів певного програмного сервісу. Все більше прикладного програмного забезпечення використовує алгоритми штучного інтелекту як для покращення користувацького досвіду (англ. User Experience), так і для оптимізації комерційних процесів.

Наприклад, для онлайн–сервісу, що накопичує інформацію про своїх користувачів, методи добування данних можуть слугувати для виявлення звичок чи смаків користувачів, та подальшої оптимізації сервісу й бізнес–процесів на основі отриманих результатів.

Існує декілька підходів до добування даних: виявлення асоціативних правил, класифікація, кластеризація та інші. Усі вони мають свої певні області застосування, в залежності від природи аналізованих даних, та від бажаного кінцевого результату.

Метод виявлення асоціативних правил (англ. Association rule learning) – це метод для виявлення певних залежностей, при аналізі великих наборів даних. Знайдені за допомогою цього метода правила можуть бути корисними для кращого розуміння процесів, та їх оптимізації. Даний метод працює, якщо дані можна подати у вигляді транзакцій, де кожна транзакція має певний набір атрибутів. В такому разі, виведені правила покажуть, що якщо для певних транзакцій характерний набір атрибутів X, то для них також, скоріш за все, буде характерним набір атрибутів Y.

Прикладом отриманого за даним методом правила може бути такий вираз: «40% користувачів, що цікавляться технологіями, також цікавляться науковою фантастикою, та серед усіх даних, 18% включають ці обидва атрибути». Отримавши таке правило, стає очевидним, що його можна

використати наприклад для кращого позиціювання контенту всередині додатку, що збільшить конверсію, та в решті–решт призведе до збільшення прибутку, якщо мова йде про комерційний продукт.

Наразі, коли рівень очікуваннь користувачів від використання програмних продуктів високий як ніколи, наявність інтелектуальних компонентів для кращого відображення контенту, аналізу звичок користувача, та інших оптимізацій вже, фактично, стали стандартом високоякісних сервісів та додатків. Важко собі уявити сервіс відео–хостингу YouTube без функції видачі запровонованих відео, на основі раніше переглянути. Пошуковий сервіс Google генерує основну частину прибутку за рахунок видачі релевантних рекламних оголошеннь, підібраних з використанням методів штучного інтелекту. В цьому контексі, вміння реалізовувати методи добування даних, та інтегрувати їх відіграють важливу роль в створенні сучасних, конкурентоздатних продуктів.

1 МЕТА РОБОТИ

Метою роботи є розробка мобільного додатку та відповідної інформаційної системи, що має складатись з клієнтької та серверної частин. Для цього спочатку провести аналіз засобів розробки, обрати найбільш оптимальні застосунки, які дозволять розробити стабільний та зручний додаток.

Розроблений мобільний додаток має стати платформою для мовного обміну онлайн.

Провести аналіз різноманітних методів асоціативного навчання, визначити переваги та недоліки кожного методу. Обрати один з методів на основі проведеного аналізу та зформованих вимог.

Використовуючи обраний метод асоціативного навчання, проаналізувати базу даних, утворену користувачами, для виділення асоціативних правил, що показують певні залежності між характеристиками користувачів, або показують певні закономірності в поведінці користувачів додатку.

Отримані асоціативні правила інтегрвувати в розроблений додаток, для покращення досвіду взаємодії користувача, шляхом оптимізації формування внтрішнього контенту.

2 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

2.1 Сучасний стан методу мовного обміну

Вивчення іноземної мови складається з декількох окремих напрямків, до кожної з яких потрібен особливий підхід - це написання, говоріння та сприймання на слух.

Одним із найскладніших завдань у вивченні будь-якої іноземної мови, у відриві від її носіїв, є говоріння та сприймання, адже єдиний спосіб досягти прогресу - регулярна практика.

Мовний обмін - це процес спільної мовної практики партнерів, які розмовляють різними мовами [1]. Як правило, партнери навчають один одного рідної мови. При такому підході співрозмовники мають шанс значно вдосконалити свої навички говоріння, розширити свій словниковий запас, здобути впевненість у володінні іноземною мовою.

Проблема такого підходу полягає в тому, що іноді важко знайти постійного партнера, який би практикував мову. Для її вирішення існують тематичні форуми, на яких користувачам пропонується заповнити особисту інформацію, знайти потенційних партнерів, а потім домовитись про спосіб практики. З розвитком Інтернету стало можливим не лише знаходити партнерів в Інтернеті, але і безпосередньо практикувати - використовуючи програмне забезпечення для Інтернет-дзвінків.

З появою технології VOIP - голосових даних у режимі реального часу через Інтернет стало можливим створення нового класу додатків для спілкування та обміну даними. Поволі такі програми стали повноцінною заміною класичної стільникової телефонії - технологія VOIP фактично безкоштовна і забезпечує додаткові функції при її використанні, наприклад відеодзвінки. Технологія VOIP повністю реалізована за допомогою

стандартних протоколів Інтернету, а тому може бути інтегрована в будь-яке програмне забезпечення, яке має стандартні інтерфейси з протоколами TCP / IP. Це дозволяє створювати додатки, що орієнтуються на певну тематичну область, одночасно дозволяючи дзвінки між користувачами.

Використовуючи тематичний додаток, користувачі можуть знайти партнерів та практикувати мови в будь-який час та де завгодно. Однак є ще одна проблема з таким підходом - користувачам часто важко знайти спільні теми та вести випадкову бесіду з незнайомцями в Інтернеті. Іноді доводиться витрачати багато часу на пошуки партнера, узгодження теми, часу тощо.

Концепція, запропонована в розробленому додатку в цій роботі, - це автоматичний підбір партнера на основі обраної теми розмови. Таким чином, користувачі не повинні домовлятися між собою про якісь формальності перед початком розмови. Одне з головних завдань, яке потрібно вирішити для забезпечення роботи такого підходу, - це інтелектуальний підбір тем, щоб користувачеві пропонувалося саме ті теми, які його цікавлять.

2.2 Огляд технології VOIP

VOIP (голосовий протокол через Інтернет) - це технологія, яка дозволяє здійснювати аудіо та відеодзвінки за допомогою Інтернету замість звичайної мережі загального телефонного зв’язку. Ця технологія, як і традиційна телефонна мережа, складається з постачальника послуг та кінцевих користувачів.

Технологія VOIP працює за допомогою набору Інтернет-протоколів на транспортних та прикладних рівнях, які, в свою чергу, є доповненнями до стандартного протоколу Internet. Тому цю технологію ще називають IP телефонією.

Основний принцип технології Інтернет-дзвінків - кодувати отриманих

за допомогою пристрою голосового введення данних, розбивати їх на окремі пакети та надсилати такі пакети за допомогою відповідних протоколів. Одержувач, у свою чергу, повинен використовувати відповідний алгоритм кодування / декодування для декодування отриманих пакетів та відтворення їх через пристрій виводу.

Існує кілька стандартів і стеків відповідних протоколів для впровадження технології VOIP - деякі розроблені для інтеграції зі звичайними мережами загального телефонного зв’язку, деякі забезпечують роботу лише через IP-мережі.

2.3 Протоколи VOIP

Технологія VOIP включає декілька протоколів, кожен з яких виконує конкретне завдання щодо забезпечення дзвінків через Інтернет.

Щоб розпочати дзвінок, сторони повинні домовитись про вибір алгоритмів кодування та декодування, мати спільний протокол для ініціювання та закінчення виклику тощо.

Для прямої передачі голосових даних використовується протокол, який визначає формат пакетів голосових даних, а також метаінформацію, необхідну для правильного відтворення їх на стороні приймача.

Також потрібно мати протокол контролю зв'язку, контролю якості.

2.3.1 Протокол ініціювання сесії

SIP (протокол ініціювання сесії) - протокол, стандартизований в RFC 3261 [2]. Цей протокол несе відповідальність за пошук адрес учасників дзвінка, визначення та узгодження технічних параметрів, а також процедуру початку та закінчення дзвінка. SIP використовує стандартний формат для

кодування загальнодоступних адрес обох сторін - ці адреси використовуються як телефонний номер, знаючи, який дзвінок зробити. Загалом SIP включає шість методів, стандартизованих в RFC 3261 [2] (табл. 2.1).

Таблиця 2.1 – Методи протоколу SIP

Метод	Опис
1	2
INVITE	Використовується для ініціації сесії
ACK	Використовується для підтврдження встановлення сесії
BYE	Використовується для запиту закінчити поточну сесію
OPTIONS	Використовується для запиту технічних характеристик
CANCEL	Використовується для відхилення поточного запиту
REFISTER	Використовується для інформування серверу регістрації про поточну адресу

Для ініціювання сеансу сторона, що викликає, генерує запит INVITE та надсилає його через один з транспортних протоколів. Цей запит містить інформацію про характеристики пристрою, з якого робиться запит, алгоритм

кодування та інші.

Приймаюча сторона відповідає на дзвінок за допомогою SIP-коду, визначеного в стандарті. Крім того, відповідь може також містити технічну інформацію.

Протокол SIP використовує процедуру трифазного ініціювання, тому після отримання відповіді від приймаючої сторони сторона, що викликає, надсилає повідомлення ACK, сповіщаючи про початок сеансу.

Кожна сторона має можливість закінчити поточний сеанс, надіславши повідомлення BYE. Отримавши це повідомлення, інша сторона повинна відповісти методом ACK для стандартного кінця сесії.

Перш ніж розпочати процедуру налаштування виклику, сторона, що викликає, може використовувати метод OPTIONS для ідентифікації характеристик пристрою іншої сторони. Це може бути корисно для запобігання ситуації, коли приймаючий пристрій взагалі не може підтримувати Інтернет-дзвінки.

Метод REGISTER використовується для повідомлення адреси центральному серверу, який виконує маршрутизацію в мережі SIP. Це необхідно для того, щоб ви могли здійснювати дзвінки на пристрій, який не знаходиться постійно в одній мережі і тому має різні мережеві адреси. Щоб мати можливість приймати дзвінки в таких умовах, необхідно регулярно повідомляти про свою мережеву адресу серверу маршрутизації, який переадресовує виклик (рис. 2.1).

Рисунок 2.1 – Схема ініціації дзвінка через сервер маршрутизації

Окрім наведених методів, SIP має інші розширення, що включають очікування дзвінків, шифрування даних, аутентифікацію. Також даним протоколом передбачена можливість виконування дзвінків між SIP-пристроями та звичайними телефонами, в разі присутності відповідних шлюзів між публічною телефонною мережею та мережею інтернет.

2.3.2 Протокол передачі даних в реальному часі

RTP - протокол передачі даних в режимі реального часу є основним функціональним компонентом, що забезпечує роботу технології VOIP. Насправді лише цей протокол може повністю забезпечити базовий Інтернет-дзвінок за умови, що обидві сторони заздалегідь знають мережеві адреси один одного, якщо не потрібно контролювати якість, і домовитися про початок і кінець сеансу. Звичайно, у реальних програмних програмах не обійтися без таких функцій, тому протокол RTP співпрацює з іншими підтримуючими протоколами технології VOIP.

RTP - протокол прикладного рівня, який зазвичай працює через протокол UDP транспортного рівня - протокол датаграм користувача - додаючи до нього свої заголовки та дані (рис. 2.2).

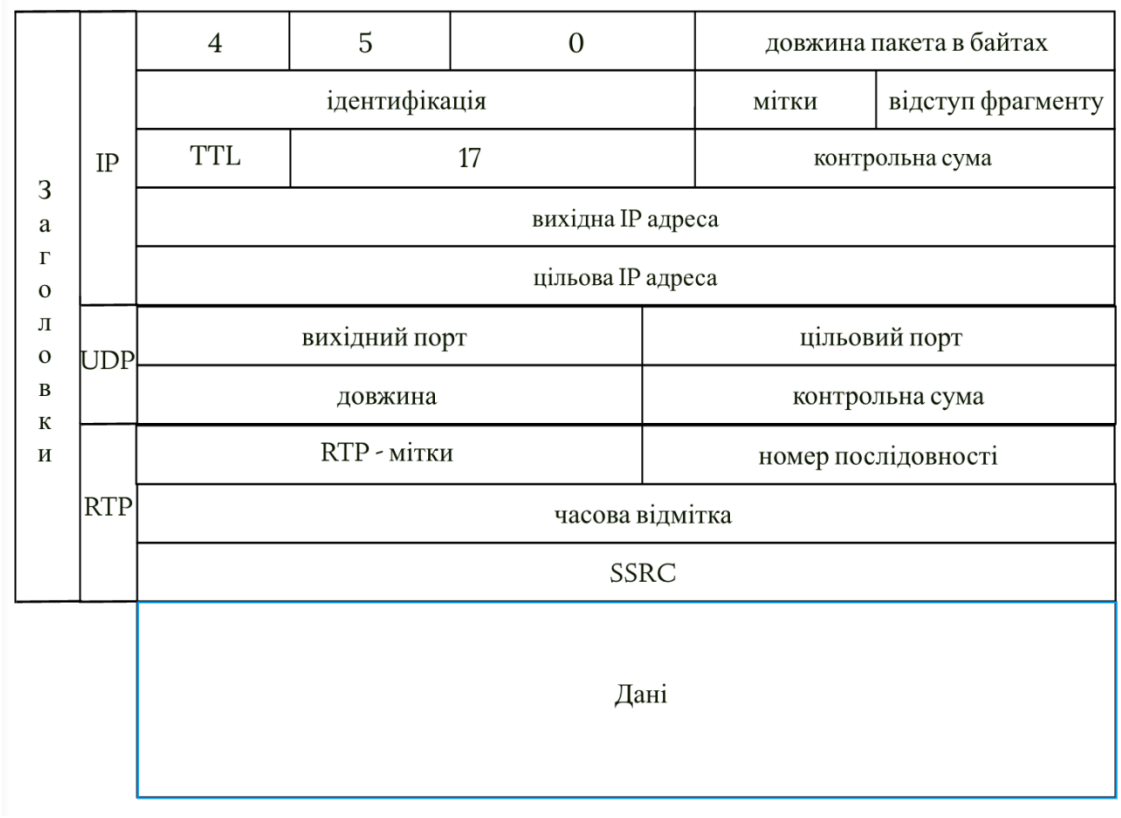

Рисунок 2.2 – Структура RTP пакету

RTP не надає жодних гарантій доставки пакетів [3]. Однак є кілька важливих функцій, які RTP виконує - крім прямої доставки пакетів з голосовими даними - це виявлення втрачених пакетів, моніторинг затримок доставки, виявлення вхідних пакетів.

Першим етапом запуску сеансу RTP є кодування вхідних голосових сигналів за допомогою вибраного алгоритму кодування. Якість кодованого звуку також залежить від цього алгоритму, який визначається кількістю біт на одиницю вибірки. Під час ініціювання сеансу вибраний кодек

надсилається хосту у відповідних заголовках.

Наступний крок передачі даних - пакетизація - поділ закодованих зразків на окремі датаграми для передачі. Вибираючи розмір пакетів, необхідно враховувати компроміс між затримкою формування пакету - кількість часу до того, як пакет буде готовий відправити, та ефективністю транспортування - відправка кожного пакету передбачає додаткові накладні витрати при додаванні заголовків протоколу на різних рівнів. Зазвичай час упаковки вибирається досить короткий - від 20 до 30 мілісекунд, якщо мова йде про передачу аудіо даних.

Пакетні дані надсилаються через протокол RTP. Оскільки дані передаються в режимі реального часу, необхідно забезпечити мінімальну затримку. Це означає, що немає часу виявити пропущені пакети та переслати їх, тому транспортний рівень використовує протокол UDP - він не містить ніяких процедур контролю цілісності, і по суті є доповненням до протоколу IP для забезпечення мультиплексування. Однак RTP містить відповідні поля у своєму заголовку для викриття вхідних пакетів.

Коли RTP-пакет надходить до пункту призначення, дані, які він містить, відокремлюються від заголовків і буферуються, готуючи достатню кількість зразків для відтворення. Вибір оптимального розміру буфера має вирішальне значення для забезпечення якості звуку. Занадто малий розмір буфера може спричинити прогалини у відтворюваному аудіо, а занадто великий може спричинити помітну затримку.

Останній крок - розшифрувати отримані дані та відтворити їх у вигляді аудіо чи відео.

2.3.3 Протокол контролю передачі в реальному часі RTCP

RTCP - це протокол, який працює разом з протоколом RTP і дозволяє

учасникам сеансів RTP обмінюватися звітами про якість зв'язку та іншою статистикою.

2.4 Аналіз існуючих імплементацій технології VOIP

Технологія Інтернет-телефонії - це складна система, що складається з багатьох компонентів і підсистем. Очевидно, його реалізація з нуля може зайняти багато років роботи цілої команди фахівців. Тому, розробляючи прикладну тематичну програму на основі цієї технології, важливо використовувати одне з готових рішень.

На щастя, наразі існує декілька таких рішень: від наборів бібліотек з відкритим кодом низького рівня до інтегрованих платформ і SDK.

2.4.1 Бібліотека PJSIP

PJSIP - це безкоштовна бібліотека з відкритим кодом, написана мовою програмування C [4]. Ця бібліотека реалізує основні протоколи, необхідні для роботи технології VOIP - протокол встановлення сеансу, протокол опису сеансу, протокол передачі даних у режимі реального часу.

Бібліотека PJSIP призначена для інтеграції з використанням різних рівнів інтерфейсу користувача. Ще однією вагомою перевагою цієї бібліотеки є її здатність працювати на великій кількості операційних систем - включаючи всі сучасні мобільні операційні системи.

На найнижчому рівні PJSIP складається з окремих бібліотек: PJMEDIA, PJNATH, PJLIB-UTIL, PJLIB. Використання цього рівня забезпечує найбільшу універсальність та можливість максимально ефективно використовувати ресурси певної цільової системи. Однак інтеграція PJSIP у вигляді окремих бібліотек низького рівня має суттєвий недолік за рівнем

складності та часу.

Наступний рівень в архітектурі PJSIP - це єдиний інтерфейс, який об'єднує всі бібліотеки низького рівня в одну. Додатковий рівень абстракції завжди коштує певної ефективності, але економить багато часу та витрати на інтеграцію.

Крім того, PJSIP також має повністю об'єктно-орієнтований інтерфейс PJSUA2, реалізований мовою програмування C ++.

Загалом, можна сказати, що ця бібліотека може бути хорошим вибором для реалізації VOIP-інфраструктури з нуля, оскільки вона забезпечує готову реалізацію більшості стандартизованих протоколів, має відкритий код та досить детальну документацію.

PJSIP використовує як подвійну ліцензію GPL, так і фірмову ліцензію, що може бути узгоджено з авторами проекту. Слід пам’ятати, що GPL вимагає, щоб усі похідні роботи публікували вихідний код, тому він не підходить для комерційних проектів.

2.4.2 Платформа Sinch

Sinch - це хмарна платформа для розробки мобільних додатків на основі технологій зв'язку в Інтернеті, включаючи аудіо та відеодзвінки. Ця платформа забезпечує повністю інтегрований програмний інтерфейс та набір бібліотек для швидкої реалізації функцій VOIP.

Основна перевага платформи Sinch - простота інтеграції. Sinch надає набори розробки для операційних систем iOS, Android, які інкапсулюють роботу з компонентами низького рівня та інтеграцію з SinchAPI - веб-інтерфейсом платформи.

2.5 Огляд існуючих аналогів

В даний час існує кілька аналогів застосунку, запропонованого в цій роботі. Однак кожен з них має приблизно однакову концепцію обміну мовою. Користувачі реєструються, надають певну особисту інформацію, а потім повинні шукати відповідного партнера, домовлятися про всі деталі, такі як час, домовитись про тему розмови тощо. Все це займає багато часу, тоді як більшість користувачів просто хочуть практикувати.

2.5.1 Мобільний застосунок для практики мов «Tandem»

Тандем - це платформа для обміну мовами. Існують мобільні додатки для платформ iOS та Android [5]. Користувача просять створити обліковий запис, вказати особисту інформацію, рідну та цільову мови. На основі цих даних складається перелік відповідних потенційних партнерів (додаток А). Вибравши партнера, ви можете зв’язатися з ними за допомогою вбудованого месенджера, і врешті розпочати свою практику за допомогою інтегрованого аудіо та відеочату.

Платформа безкоштовна для спілкування між користувачами. Крім того, є можливість пройти практику з професійними викладачами, які встановлюють тарифи на їхні послуги.

2.5.2 Мобільний застосунок для практики мов «HelloTalk»

HelloTalk - ще один додаток для операційних систем iOS та Android, який забезпечує платформу для обміну мовою. У порівнянні з тандемом HelloTalk більше орієнтований на текстове спілкування, хоча також має вбудований аудіо та відеочат. Вагомою перевагою цієї послуги є велика база

даних користувачів - понад десять мільйонів [6]. В інших аспектах платформа повторює стандартні концепції цієї галузі, а саме: формування списку партнерів, які відповідають параметрам (додаток А), особиста координація деталей практики між користувачами.

2.6 Концепція запропонованого додатку

Хоча вищезазначені аналоги служать платформою для обміну мовою, вони не забезпечують жодних механізмів, щоб забезпечити негайний початок практики. Якщо подивитись на аналоги більш глобально, то вони по суті є звичайними соціальними мережами, спеціалізованими для конкретної цільової аудиторії.

Концепція, запропонована у розробленому додатку, принципово відрізняється - користувачі не повинні домовлятися про будь-які деталі перед початком практики. Сама система вибере партнерів на основі обраної теми розмови. Таким чином, користувачі повинні лише вибрати мову, яку вони хочуть практикувати, і тему, яку вони хотіли б обговорити.

Це виключає необов'язкове проміжне посилання у формі попереднього спілкування користувача, яке, як показала практика при дослідженні аналогів, зменшує отриману кількість діалогів.

Однак такий підхід викликає таке важливе питання: які теми розмови запропонувати користувачеві? Потрібно, щоб відображені теми збігалися з потенційними інтересами користувача.

Для вирішення цієї проблеми ідеальний метод асоціативного навчання - за допомогою аналізу баз даних формуються асоціативні правила, які можна інтегрувати в додаток. Очевидно, що для аналізу бази даних вона вже повинна існувати. Це означає, що метод асоціативного навчання можна використовувати лише після того, як програма вже зібрала певний масив

репрезентативних даних - або з тестової групи користувачів, або з реального.

Метод асоціативного навчання - це добре вивчена сфера обміну даними. Існує кілька алгоритмів, які реалізують цей метод і здатні працювати над різними типами даних - бінарними, категоричними, числовими.

У наступних розділах буде надано детальний аналіз методу та доступних реалізацій, і виберемо той, який найкраще відповідає цілям та найкращим чином відповідає природі даних, що генеруються користувачами програми.

2.7 Загальний огляд методу навчання асоціативних правил

Вперше проблема визначення асоціативних правил була пов’язана з проблемою прийняття рішень, з якою стикаються майже всі великі торгові центри - оптимальне розміщення товарів. Прогрес у цьому питанні став можливим завдяки впровадженню штрих-кодів та налагодженню цифрового обліку товарів. Це дало можливість збирати всі транзакції, здійснені в базі даних, яка містила всю можливу інформацію про кожну покупку, здійснену клієнтами. Стало зрозуміло, що аналіз зібраних даних дасть можливість краще зрозуміти звички клієнтів, які можна використовувати для вдосконалення маркетингу, тобто фактично оптимізувати процес.

Саме при вирішенні цієї проблеми був розроблений алгоритм асоціативного навчання, який зараз використовується у багатьох проблемах, де необхідно знайти зв’язки між транзакціями в базі даних.

Сферами застосування алгоритму викладання асоціативних правил є аналіз ринку, маркетинг, виявлення різних видів шахрайства, медицина.

Цей алгоритм працює на наборі транзакцій, де кожна транзакція містить набір властивостей. Простий приклад знайденого правила полягає в тому, що 80% клієнтів, які купують комп'ютер, також купують клавіатуру та

мишку.

2.7.1 Формальне визначення методу навчання асоціативних правил

Нехай $I = \{i_1, i_2, \dots i_m\}$ – множина літералів, назвімо їх властивостями. Нехай D –множина транзакцій, в якій кожна транзакція T – це множина властивостей, при чому $T \subseteq I$. Якщо транзакція T має набір деяких властивостей X з множини I, то асоціативне правило можна описати як $X \Longrightarrow Y$, при чому $X \subset I$, $Y \subset I$, і$X \cap Y = \emptyset$[7].

Кожне правило характеризується двома властивостями c% (confidence) та s% (support).

Confidence (c%) –впевненість, відношення кількості транзакцій, що задовольняють Y, серед тих, що задовольняють X. Впевненістьхарактеризуєміру впевненості у тому, що знаючи X, можна визначити Y.

Support (s%) – підтримка, це пропорція транзакцій, що задовольняють X і Y, до загальної кількості транзакцій в множині D.

Виділивши множину всіх транзакцій D, задача навчання асоціативних правил зводиться до того, щоб знайти всі можливі правила виду $X \Longrightarrow Y$, де значення підтримки (s%) та впевненості(c%)мають значення вищі, ніж мінімальні значення, задані користувачем, які позначаються як min(s%) та min(c%).

Варто зазначити, що набір транзакцій D не має чіткого формату представлення, це може бути будь–який набір даних, наприклад таблиця бази даних, або файл.

2.8 Огляд існуючих алгоритмів

Було запропоновано декілька варіантів здійснення викладання асоціативних правил. Кожен з них має свої переваги та недоліки, які будуть проаналізовані в наступних розділах.

2.8.1 Алгоритм Apriori

Цей алгоритм знаходить усі набори властивостей, які задовольняють заданому значенню min (s%), тобто мають досить високу частоту серед усіх транзакцій. Після виявлення таких множин серед них виділяються асоціативні правила [8].

У рамках цього алгоритму кожна транзакція визначається як набір властивостей. Apriori працює, знаходячи всі часті властивості, потім розширює набір таких властивостей до тих пір, поки ці властивості досить часто з’являються в початкових даних.

Першим кроком алгоритму є пошук наборів властивостей, які задовольняють заданому значенню min (s%)

Розглянемо роботу алгоритму на прикладі бази даних транзакцій, наведеної в таблиці 2.2.

Таблиця 2.2 – База даних транзакцій

Номер транзакції	A	B	C	D	E
1	1	1	0	0	0
2	0	0	1	0	1
3	1	1	0	0	0

Якщо встановлене значення min (s%) дорівнює 30%, вибираються лише ті властивості, які трапляються в операціях принаймні в 2 рази (табл. 2.3).

Після обраних властивостей алгоритм Апріорі формує всі можливі пари властивостей, враховуючи, що порядок не важливий, тобто AB такий же, як і BA.

Таблиця 2.3 – Властивості, що відповідають заданному значенню s%

Властивість	Частота	значення s% (%)
A	3	75
B	3	75
C	2	40
E	2	40

Після формування наборів властивостей алгоритм залишає лише ті множини, які задовольняють заданому значенню min (s%).

З решти наборів у майбутньому формуються нові набори, які вже будуть складатися з трьох властивостей.

Описана процедура триватиме до тих пір, поки новостворені набори властивостей задовольнять min (s%).

Таким чином, можна зобразити роботу Апріорі за допомогою схеми (рис. 2.3)

На малюнку 2.3 показані основні етапи алгоритму Apriori для визначення кінцевого набору властивостей, які мають досить високу частоту в заданій базі даних.

Сформувавши набір властивостей з досить високою частотою, ви

можете вивести асоціативні правила, вибравши ті, які мають досить високе значення довіри. Тобто, якщо ABCD – набір, що задовольняє значенню min(s%), то можна визначити, чи підтримується правило AB $\Longrightarrow$ CD за формулою 2.1.

$$c\% = \frac{s\%(ABCD)}{s\%(AB)}, \tag{2.1}$$

де c% – міра впевненості (confidence);

s% – міра підтримки (support).

Якщо значення c% більше, ніж значення min (c%), правило підтримується. Описана процедура може бути представлена як псевдокод (приклад 2.1).

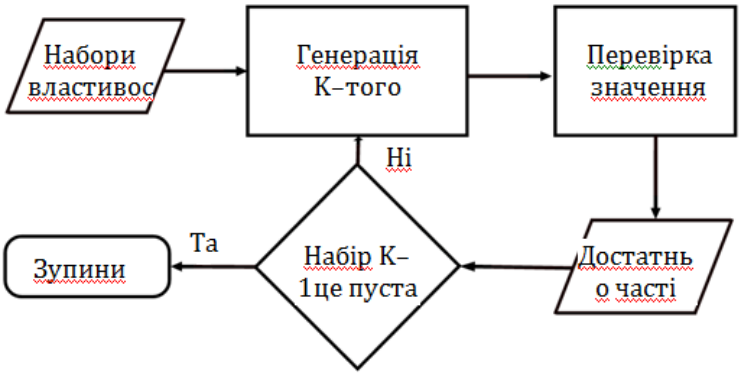

Рисунок 2.3 – Схема алгоритму Apriori

```
forall frequent itemsets l_k, k ≥ 2 do
   call genrules(l_k, l_k);


procedure genrules(l_k: frequent k-itemset, a_m: frequent m-itemset)
1) A := {(m−1)-itemsets a_{m−1}  |  a_{m−1} ⊂ a_m};
2) forall a_{m−1} ∈ A do begin
3)     conf := support(l_k)/support(a_{m−1});
4)     if (conf ≥ minconf) then begin
7)         output the rule a_{m−1} ⇒ (l_k − a_{m−1}), with confidence = conf
               and support = support(l_k);
8)         if (m − 1 > 1) then
9)             call genrules(l_k, a_{m−1});

10)    end
11)end
```

Приклад 2.1 – Алгоритм виведення правил зі зформованих наборів частих властивостей

2.8.2 Виведення асоціативних правил на числових даних

На практиці більшість баз даних не обмежуються категоричними даними, а в основному містять числові дані.

Алгоритм Апріорі, розглянутий у розділі 2.7.1, є досить ефективним і відносно простим у здійсненні рішенням, але його використання обмежується булевими значеннями атрибутів у початковому наборі транзакцій.

Якщо значення атрибутів у наборі джерел транзакції задано числовими значеннями (таблиця 2.4), то для використання алгоритму Апріорі необхідно пробити вибірки атрибутів - тобто вибрати певні інтервали та розподілити значення властивостей.

Вибір діапазонів для вибірки може бути здійснено з використанням відповідних знань предметної області або з використанням методів

кластеризації. Ефективність такого підходу може сильно залежати від обраного методу відбору і може бути значно знижена при наявності статистичних викидів. Можна взяти вибірку даних, перевіривши, чи є прийняте значення інтервалом (табл. 2.5).

Таблиця 2.4 – множина транзакцій з числовими значеннямиатрибутів

Номер транзакції	A	B	C
1	23	78	1
2	14	75	2
3	18	68	5
4	19	67	3

В множині транзакцій I атрибути A, B, C задачі числовими значеннями (таблиця 2.4). Після дискретизації, утворюється нова множина транзакцій I_dв якій всі атрибути мають булеві значення (таблиця 2.5).

Таблиця 2.5 – дискретизовані атрибути

Номер транзакції	A: 10..20	A: 20..30	B: 60..70	B: 70..80	C: 1..3	C: 3..6
1	0	1	0	1	1	0
2	1	0	0	1	1	0
3	1	0	1	0	0	1
4	1	0	1	0	0	1

Підхід до отримання асоціативних правил із числових даних шляхом введення інтервалів має кілька суттєвих недоліків. По-перше, використання діапазонів для розподілу числових даних обмежене, а іноді призводить до неточних результатів. Наприклад, використовуючи такий підхід, ви можете отримати таке правило:

$$\text{Н} \in [100cm, 150cm] \Longrightarrow \text{А} \in [0{,}14](70\%), \tag{2.2}$$

де Н – зріст;
А – вік.

З правила 2.2 випливає, що людина у віці 1 року може мати зріст 100 сантиметрів, хоча це не відображає реальну ситуацію. Цей приклад показує, наскільки чутливим може бути кінцевий результат до вибраних діапазонів вибірки. Крім того, введення діапазонів може призвести до необґрунтовано великої кількості похідних правил, оскільки праву частину правила завжди можна розширити. Ще одним недоліком методу є необхідність попередньо підготувати дані до того, як алгоритм може бути використаний. Це додає додаткової роботи, якщо таке навчання проводиться вручну. Крім того, використання вибірки призводить до часткової втрати даних, тому, насправді, можна дати лише приблизні правила.

2.8.3 Розподільний підхід

Інший підхід до отримання асоціативних правил з кількістю використаних даних роздільних значень. У рамках цього підходу можуть бути подані наступні асоціативні правила:

$$S = \text{female} \Longrightarrow W = 7.90 \text{ p/hr } (m = 9.02), \tag{2.3}$$

де S – стать;

W – середнє значення заробітної платні серед жінок;

m – загальне середнє значення заробітної платні.

Це правило виражає значну різницю в роботі, виконаній середньоплановою середньою групою людей, яка визначила загальне середнє значення, яке має атрибути в базі даних. Окремий підхід розглядає поняття «цікавість» - ефективне розсіювання значних модулів активного підмножини стосовно таких значень для всіх множин. Цей підхід заснований на сучасних методах статистичних результатів для підтвердження обґрунтованості похідних правил. Метод розширюваного тексту не вибірковий, натомість він розглядає вихідні повідомлення як неперевірені значення [9].

Якщо подивитися на категорії даних, "цікавість"можна визначити, як ступінь частоти появи певних атрибутів відносно їх загальної кількості. В такому випадку дані можна зобразити у вигляді списку атрибутів з відповідними частотами появи їх значеннь. Для числових значень, що знаходяться всередині одного, при розширенні середніх та атрибутів дисперсії. Знаючи групи атрибутів, розподіляючи їх, було досягнуто середнє значення вибору, можна сформувати асоціативні правила. Наприклад, для бази даних «тривалість життя» даний підхід дозволить вивести правило, базуючись на значному відхилення по середньому значення тривалості життя серед людей, що не курять та не вживають алкоголь. Таке правило буде виглядати так:

$$S \text{ and } A = 0 \Longrightarrow E = 85 \ (m = 80), \tag{2.4}$$

де S –індикатор куріння;

A –індикатор вживання алкоголю;

E –очікувана тривалість життя.

Одним з недоліків розподільного підходу є формування правил специфічного формату, та неможливість використання декількох числових атрибутів в лівій частині правила.

2.8.4 Оптимізаційний підхід

Оптимізаційний підхід, так само як і розподільний, опрацьовує не лише булеві атрибути, а й числові.

При оптимізаційному підході, асоціативні правила мають таку форму:

$$(B \in [v_1, v_2]) \Longrightarrow (C = yes), \tag{2.5}$$

де B –значення атрибуту лівої сторони правила;

v_1–ліва границя інтервалу;

v_2–права границя інтервалу;

C –значення атрибуту правої сторони правила.

Крім того, в рамках даного підходу вводиться поняття виграшу (Gain) – що визначається як компроміс між значеннями підтримки та впевненості:

$$Gain(A \Longrightarrow B) = Supp(AB) - MinConf * Supp(A), \tag{2.6}$$

де A – атрибут з лівої частини правила;

B – атрибут з правої частини правила.

Основним недоліком даного підходу можна вважати неможливість містити більше двох числових атрибутів.

2.8.5 Алгоритм QuantMiner

QuantMiner - це підхід до виведення асоціативних правил, заснований на використанні генетичних алгоритмів [10]. Цей метод працює безпосередньо з такими поняттями, як шаблони правил - заздалегідь задані формати правил, встановлені вручну або автоматично. Шаблони правил служать орієнтиром для процесу виведення правил. QuantMiner знаходить найкращі інтервали для кожного числового атрибута - приклад 2.2, який присутній у шаблоні за допомогою генетичних алгоритмів [10].

Формально шаблон може бути визначений як набір атрибутів в обох частинах асоціативного правила.

Основною перевагою алгоритму QuantMiner є його спрямованість на роботу з числовими даними, без виконання їх вибірки, що дозволяє досягти кращих результатів [10].

Input: A dataset composed of NbTuples, PopSize,
GenNb, CR, MR, MinSupp, MinConf
Output: Quantitative association rules $\mathcal{R}$
Select a set of attributes
Let $\mathcal{R}_t$ a set of rule templates defined on these attributes
Compute the set of frequent itemsets on categorical attributes in $\mathcal{R}_t$
$\mathcal{R} = \emptyset$
foreach $r \in \mathcal{R}_t$ **do**
  Generate a random population POP of PopSize instantiated rules following the template r
  i=1
  while $i \leq GenNb$ **do**
    Form the next generation of population by mutation and crossover w.r.t. MR and CR.
    Keep PopSize rules in POP with the best Fitness values
    i++
  $\mathcal{R} = \mathcal{R} \cup Argmax_{R \in POP} Fitness(R)$
return $\mathcal{R}$

Приклад 2.2 – Алгоритм виведення асоціативних правил QuantMiner

2.9 Постановка задачі

Розробити iOS додаток для людей, що бажають практикувати іноземні мови.

Додаток має складатись з клієнтської та серверної частин, а також бази даних. Для розробки необхідно використовувати нативні засоби для відповідної платформи.

Додаток має надавати користувачам можливість зареєструватись, вказати деяку особисту інформацію. Користувач має обрати цікавлячу його тему розмови, після чого він може почати практикувати інозмену мову з

підібраним партнером.

Використати методи навчання асоціативних правил для проведення аналізу бази даних додатку, що містить історію обраних для розмов тем кожного користувача.

Результатом аналізу має бути набір асоціативних правил, що описують залежності між виборами певними темами, там між характеристиками користувачів та обраними темами.

Інтегрувати знайдені правила в додаток так, щоб пропоновані користувачу теми відповідали його особистим характеристикам, або були підібрані на основі попередньо обраних тем.

3 АНАЛІЗ ЗАСОБІВ РОЗРОБКИ ТА ПРОЕКТУВАННЯ ДОДАТКУ

3.1 Аналіз засобів розробки

Розробка програмного забезпечення вимагає використання певного набору інструментів та технологій. Як правило, для кожного завдання існує кілька варіантів технологій, за допомогою яких його можна виконати. Кожен варіант має свої переваги та недоліки, які необхідно враховувати при виборі остаточного набору інструментів.

При розробці додатків для iOS існує кілька варіантів стеків технологій - фірмові та сторонніх розробників. Для розробки програми було обрано фірмовий стек технологій, який включає середовище розробки Xcode 11, мову програмування Swift 5 та набір програмних компонентів для розробки під iOS CocoaTouch.

Інтегроване середовище розробки Xcode, містить усі необхідні інструменти для створення нативних програм для iOS, macOS, WatchOS та tvOS платформ. Xcode включає набір компіляторів для мов C / C ++ / Objective-C / Swift, які були розроблені Apple на основі інфраструктурного проекту компілятора LLVM. Крім того, Xcode включає текстовий редактор, колекцію проектів, інструменти графічного інтерфейсу, налагоджувач, профайлер та інші інструменти.

Інтерфейс Xcode IDE складається з безлічі панелей, кожна з яких є інтерфейсом до певного вбудованого інструменту (рис. 3.1).

На панелі проекту відображаються всі файли, проіндексовані в цьому проекті. Ці файли поділяються на групи, які не обов'язково відображають фізичне розташування файлів у файловій системі, і можуть мати будь-яку структуру.

Текстовий редактор дозволяє писати код на мові програмування Swift

або Objective-C. Текстовий редактор, вбудований у середовище Xcode, побудований на основі сучасного компілятора LLVM, дозволяє значно прискорити розробку за рахунок автоматичного заповнення тексту, автоматичного виправлення синтаксичних помилок та інших оптимізацій.

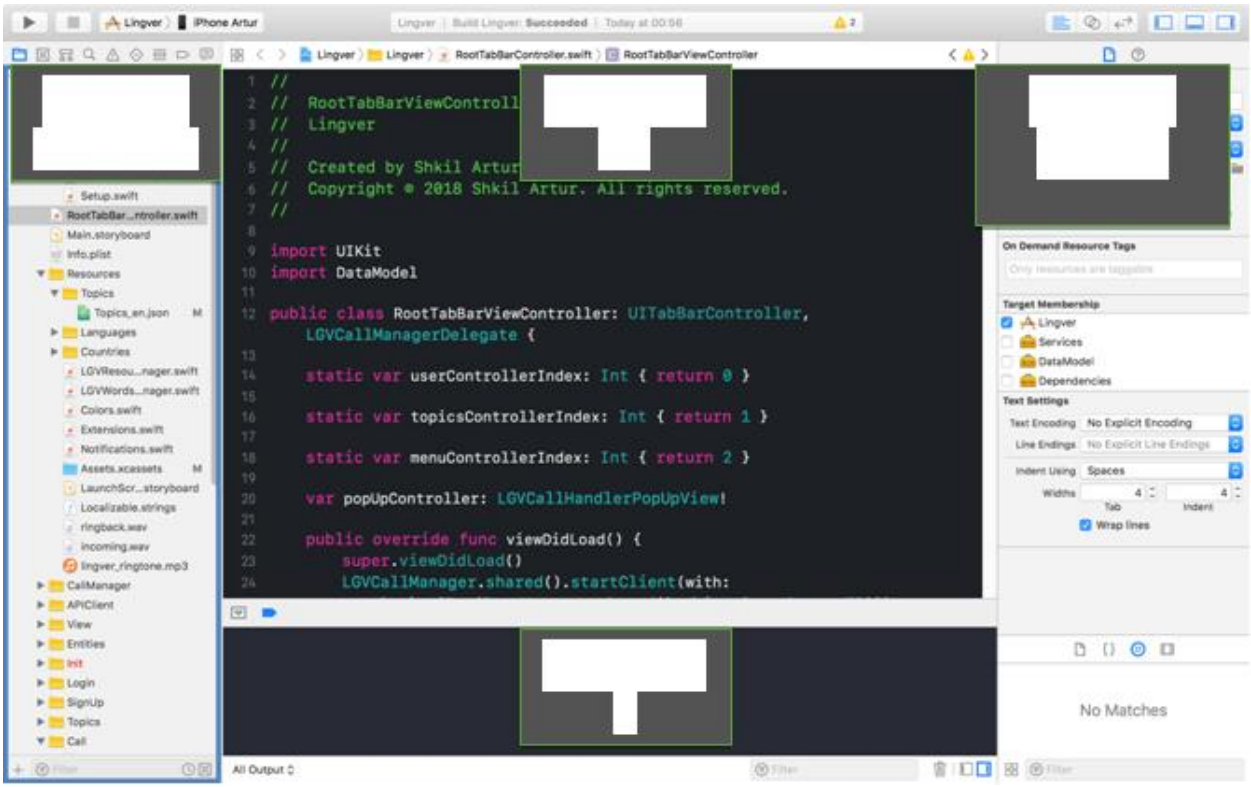

Рисунок 3.1 – Інтерфейс Xcode IDE

Термінал, вбудований в IDE, дозволяє отримувати інформацію про виконання програми та вводити команди.

Панель утиліти дозволяє налаштувати різні властивості, пов’язані з поточно вибраним файлом.

3.1.1 Засоби розробки серверної частини

Оскільки розроблений додаток - це система клієнт-сервер, необхідно вибирати технології для реалізації серверної частини, яка буде взаємодіяти з базою даних.

В даний час існує багато різних варіантів цього завдання, побудованих на різних мовах програмування, включаючи програми для мови програмування Swift.

Оскільки такі інструменти використовують подібні бібліотеки та середовища як клієнтські технології для розробки програм iOS, вибір їх дозволить розробити всю систему, використовуючи єдину мову та середовище, що прискорить розвиток.

Оскільки мова програмування Swift стала доступною як проект з відкритим кодом, для побудови серверної мови на цій мові стали доступні декілька рамок [11].

Вибираючи оптимальне рішення для розробленої програми, необхідно враховувати порівняльні тести, що характеризують їх ефективність за різними показниками (рис. 3.2).

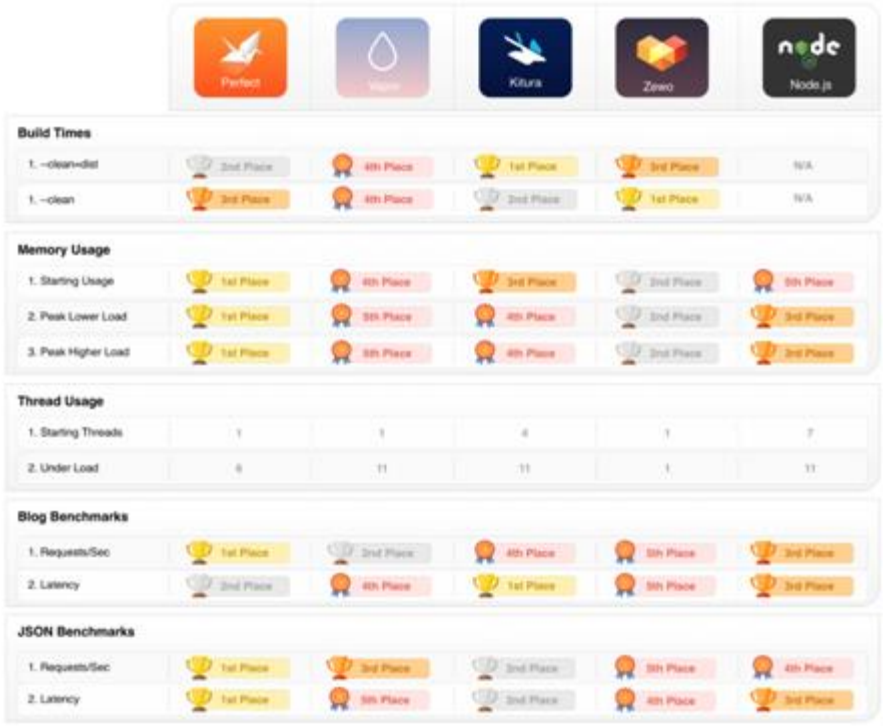

Рисунок 3.2 – Порівняльна харектеристика серверних фрейморків для мови програмування Swift

Використовуючи результати порівняння фреймворків за такими параметрами, як ефективність пам’яті, час компіляції, кількість зайнятих потоків, а також враховуючи якість та повноту програмної документації, ми можемо виділити технологію Vapor.

Vapor побудований на стандартних компонентах, що входять до набору LLVM [12], і тому дозволяє використовувати Xcode IDE для розробки. Цей фреймворк містить усі необхідні компоненти, такі як драйвери доступу до бази даних, обробка http-запитів тощо.

Пара має модульну архітектуру, що складається з декількох рівнів абстракцій та окремих компонентів (рис. 3.3).

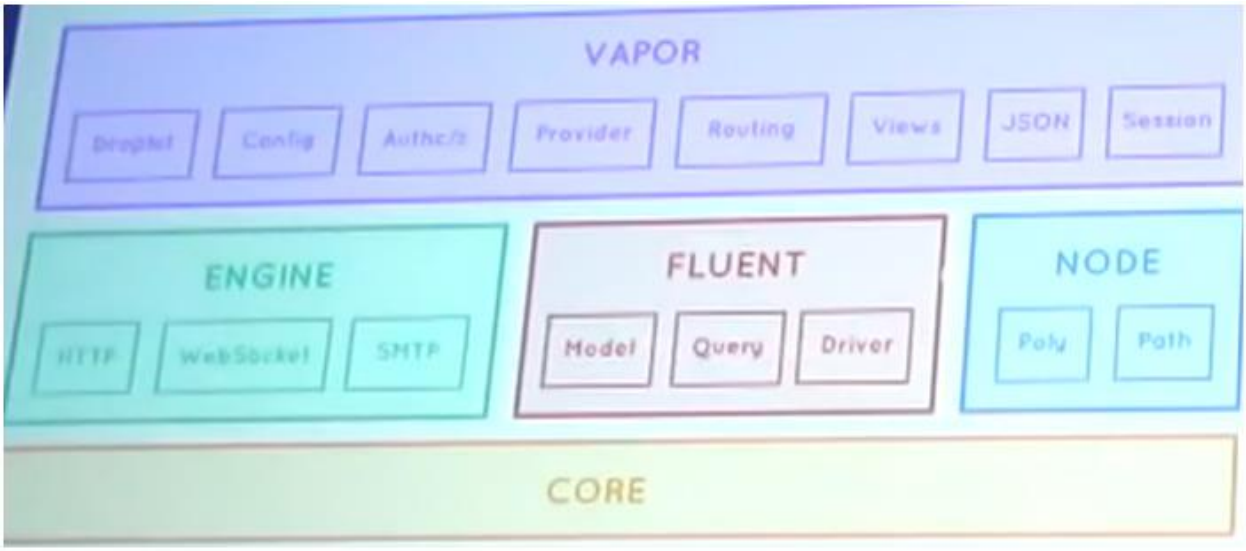

Рисунок 3.3 – Архітектура фрейморку Vapor

Цей підхід дозволяє використовувати Vapor не тільки через інтерфейси найвищого рівня, але і через внутрішні компоненти - для максимальної гнучкості та ефективності.

Як показано на малюнку 3.3, пара має три архітектурні рівні - ядро, сервіси та безпосередньо рівень веб-інтерфейсу.

3.2 Проектування додатку

Під час створення програм важливим кроком є вибір окремих компонентів, які розділяють функціональність на різних рівнях абстрагування (рис. 3.4).

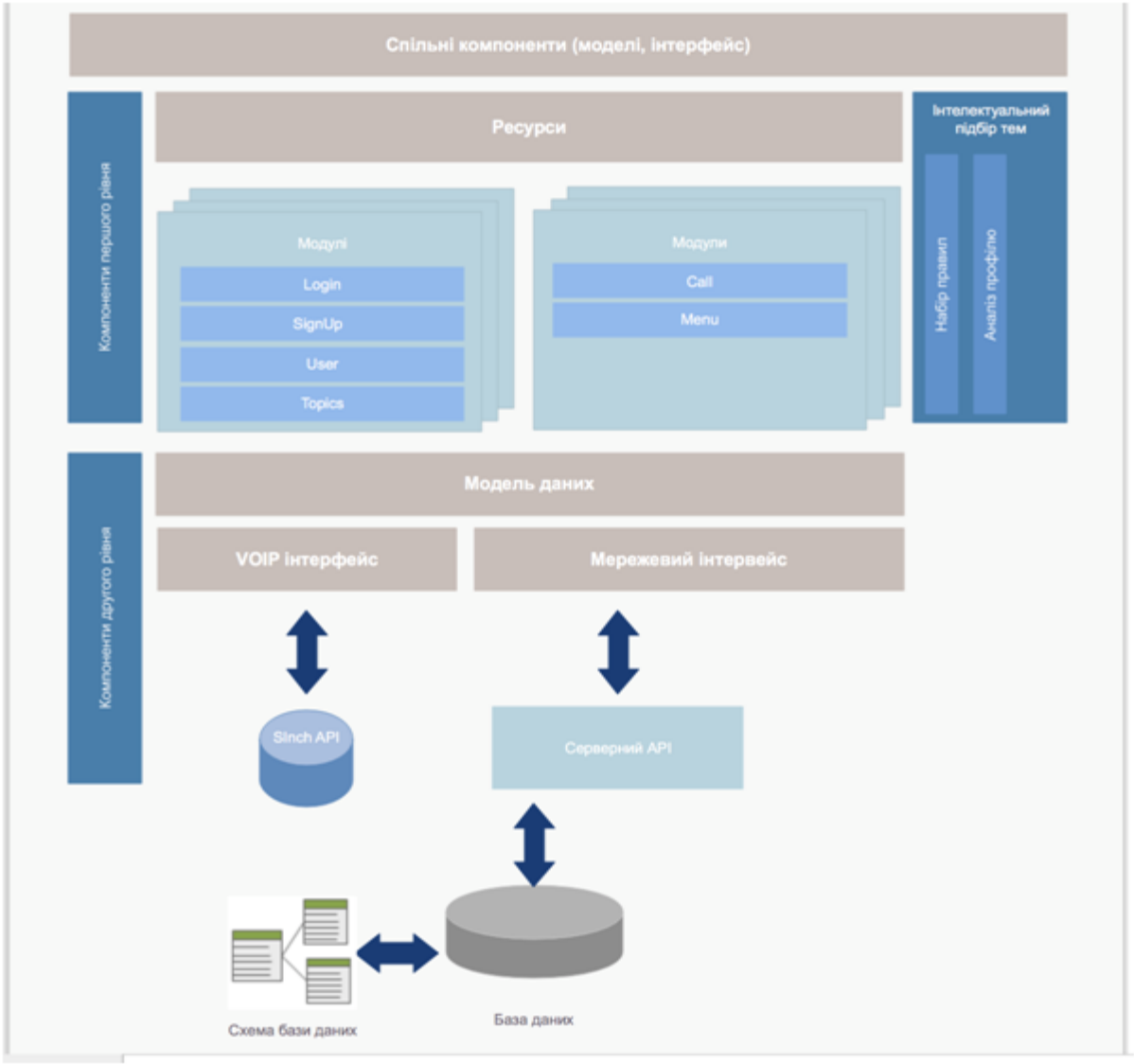

Рисунок 3.4 – Архітектура додатку

Використання модульного підходу дає можливість розробляти кожен компонент окремо незалежно від інших частин системи, що допомагає мінімізувати витрачений час на розробку та оптимізувати структуру проекту за рахунок повторного використання його компонентів. Крім того, модульну архітектуру набагато простіше перевірити та підтримувати в майбутньому, оскільки, якщо винна якась проблема, можливо її локалізувати в межах конкретного модуля.

Дворівнева архітектура була використана при розробці програми - вся функціональність, відповідальна за пряму логіку програми, формує перший рівень, тоді як функціональні компоненти другого рівня відповідають за загальні процедури, такі як мережева взаємодія, взаємодія з інтегрована база даних тощо (рис. 3.5).

Загальні компоненти складаються з різних частин інтерфейсу, моделей даних тощо, які поділяються всіма модулями. Наприклад, у цьому модулі є реалізація основних класів стилів, кнопок та інших елементів.

Ресурси інкапсулюють усі графічні, медіа та інші ресурси. Крім того, для зручності доступу до ресурсів цей модуль реалізує клас, що забезпечує простий інтерфейс для доступу до ресурсів.

Модулі реалізують пряму функціональність програми - тому вони є центральними компонентами всієї архітектури. Кожен модуль інкапсулює всі компоненти, необхідні для запуску певної частини програми, а також використовує інші функціональні компоненти, такі як модель даних для запису та читання даних, мережевий інтерфейс для надсилання запитів тощо.

Ввести функціонал, що забезпечує інтелектуальний аналіз профілю користувача та використання визначених асоціативних правил для оптимізації формування списку запропонованих тем, формує окремий компонент системи, який взаємодіє з рештою програми через свій інтерфейс.

Модель даних включає в себе всю логіку, відповідальну за взаємодію з інтегрованою базою даних, наприклад збереження, оновлення, видалення даних.

Використання інтегрованої бази даних (рис. 3.5) хоч і додає додаткової складності системі, але дає можливість використовувати певний набір функціональних можливостей програми навіть без підключення до Інтернету, оскільки це дозволяє зберегти поточний стан дані

Це означає, що користувач зможе отримати доступ до певного набору

функціональних можливостей програми навіть без активного підключення до Інтернету, що покращує досвід взаємодії користувача.

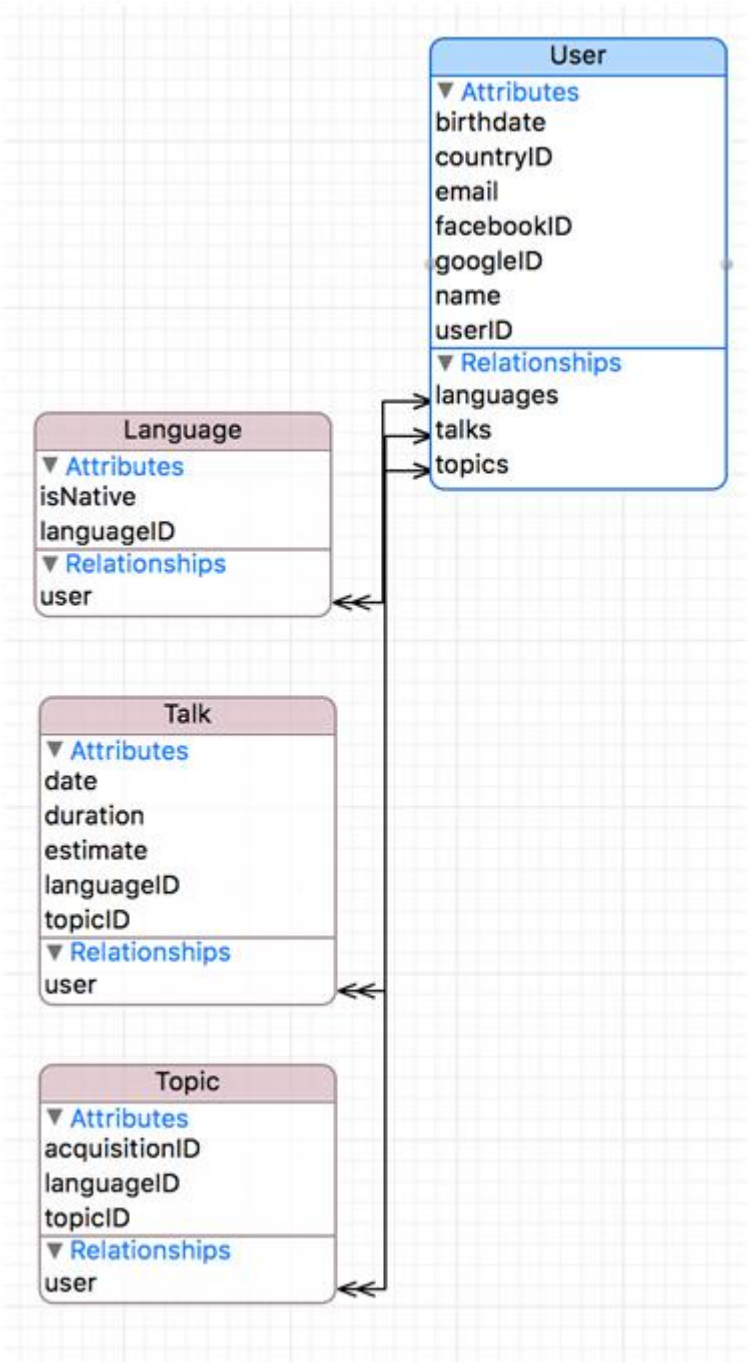

Рисунок 3.5 – Схема даних інтегрованої БД

Важливим елементом системи є центральна база даних (рис. 3.6), в якій зберігаються дані всіх користувачів програми, у тому числі дані, необхідні для роботи алгоритму для вивчення асоціативних правил.

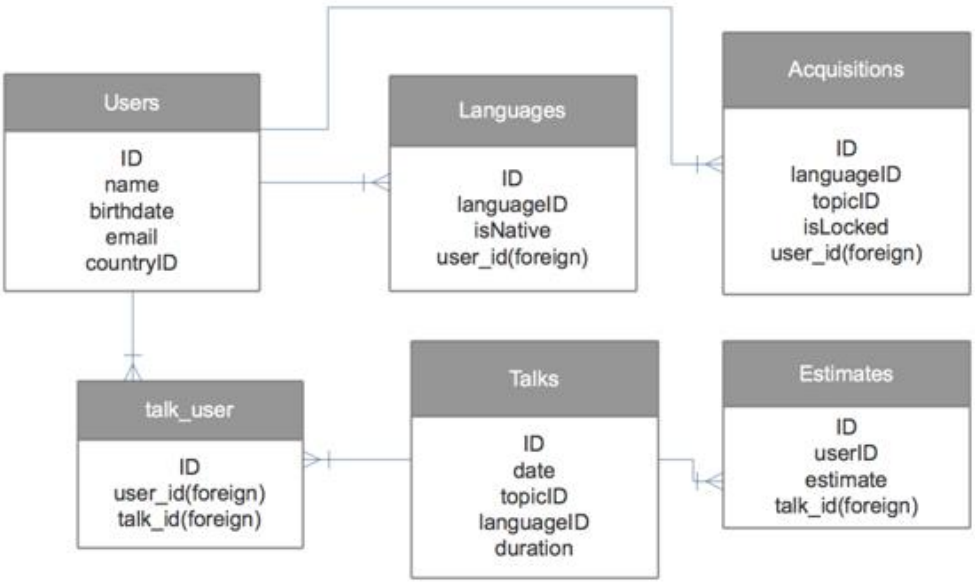

Рисунок 3.6 – Схема центральної бази даних

Елементом, що забезпечує взаємодію між базою даних та додатком, є серверний API програми. Цей елемент системи - це сукупність процедур, що мають чітко визначений комунікаційний інтерфейс. За допомогою цих процедур клієнтська частина може взаємодіяти з центральною базою даних, оновлюючи або отримуючи дані.

Завдяки такому підходу інкапсуляція всієї функціональної логіки, яка відповідає за роботу з центральною базою даних, в окремий структурний компонент.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Розробка додатку

Перевага використання модульної архітектури (рис. 3.2) полягає в тому, що кожен компонент може розроблятися незалежно від інших. Таким чином, система складається з набору окремих компонентів, кожен з яких має свою внутрішню структуру, і взаємодіє з іншими частинами через певні інтерфейси (рис. 4.1).

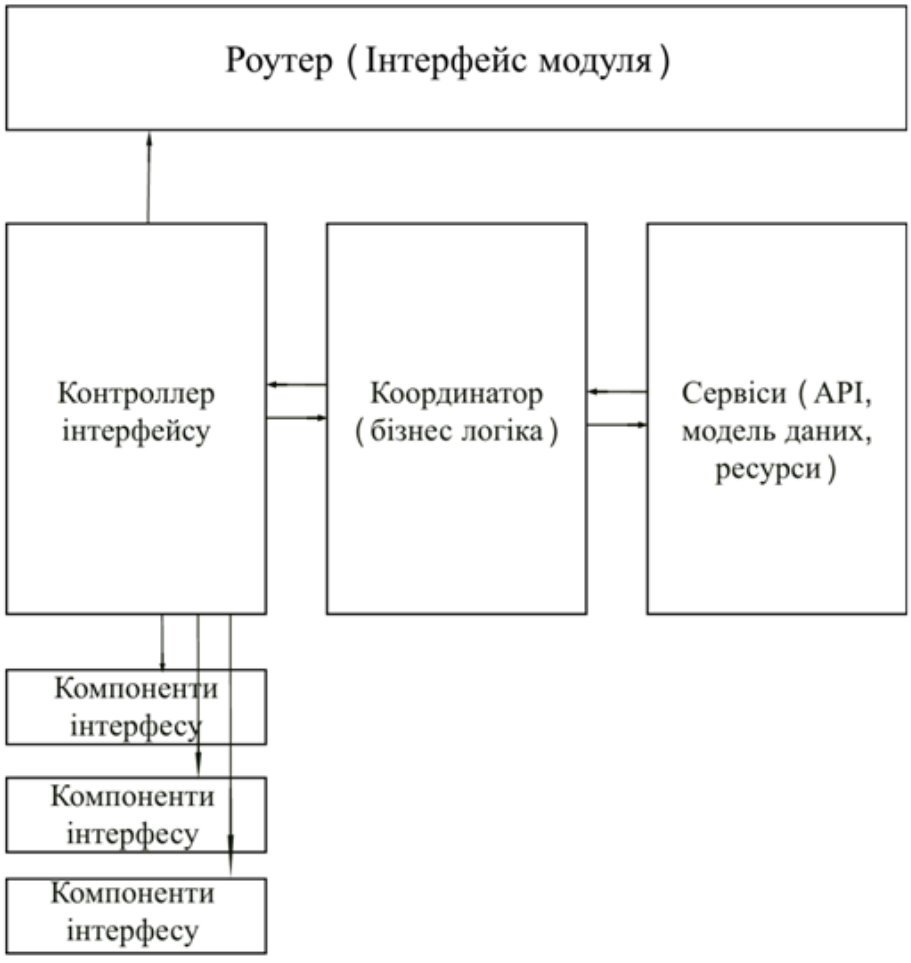

Рисунок 4.1 – Архітектура модуля додатку

Кожен модуль програми відповідає за реалізацію деяких логічно пов'язаних функціональних можливостей - реєстрації, авторизації тощо. Структура модуля складається з окремих об'єктів, які є окремими класами (рис. 4.1).

Кожен модуль, насамперед, має маршрутизатор - цей клас виконує функцію інтерфейсу, через який цей модуль може взаємодіяти з іншими (приклад 4.1).

```
class LoginRouter {

    private var storyboard = UIStoryboard(name: "Main", bundle: nil)

    //Routes:

    func presentInitialViewController(with window: UIWindow?) {
        let initialViewController = storyboard.instantiateViewController(withIdentifier: loginController) as! LoginViewController
        initialViewController.router = self
        window?.rootViewController = initialViewController
        window?.makeKeyAndVisible()
    }
```

Приклад 4.1 – Метод інтерфейсу між двома модулями

Кожен інтерфейс визначає необхідні параметри для взаємодії з іншими модулями, такими як дані, що передаються. Маршрутизатор є центральним компонентом будь-якого модуля, він завжди існує в одному екземплярі і відповідає за створення інших об'єктів.

Кожен модуль може мати кілька екранів - кожен з яких, у свою чергу, відображає інтерфейс користувача, пов’язаний з деяким етапом програми - таким як екран реєстрації / авторизації (рис. 4.2).

На малюнку 4.2 показаний початковий екран програми. З дисплея, якщо користувач відкриває програму вперше, або якщо вже зареєстрований користувач вирішив використовувати функцію для виходу зі свого облікового запису.

Цей екран складається з деяких декоративних елементів інтерфейсу, таких як анімація, підписи та функціональні кнопки.

Усі ці елементи є об’єктами класу, показаними на рисунку 4.1 як компоненти інтерфейсу. Реалізуючи компоненти, важливо використовувати принцип єдиного обов'язку, згідно з яким кожен клас повинен виконувати лише те завдання, для якого він створений [13].

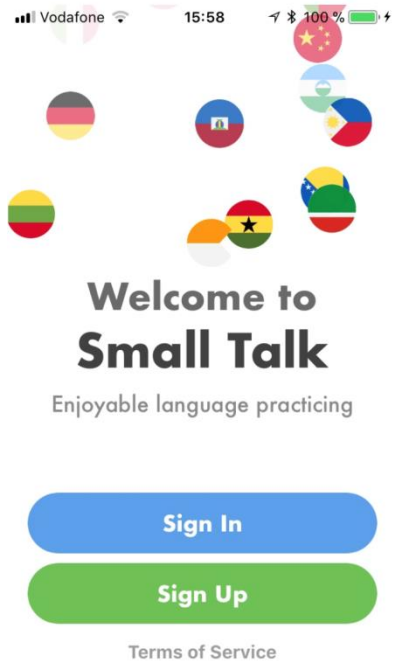

Рисунок 4.2 – Екран авторизації/реєстрації

Тому класи компонентів інтерфейсу повинні відповідати виключно за вказівку візуальних характеристик, таких як шрифт, розмір, колір та інші стилістичні елементи (приклад 4.2).

```
public class LGVRoundedButton: UIButton {

    public override func layoutSubviews() {
        super.layoutSubviews()
        clipsToBounds = true
        layer.cornerRadius = frame.height / 2
        setTitleColor(UIColor.white, for: .normal)
        titleLabel?.font = UIFont(name: "Futura-Bold", size: 18)
    }
}
```

Приклад 4.2 – Реалізація кнопки – інтерактивного елементу інтерфейсу

Це означає, що елементи інтерфейсу не містять ніяких логічних механізмів - цю частину роботи виконує контролер. Призначення контролера походить від його назви - керувати інтерфейсом.

Наприклад, якщо користувач натискає одну з кнопок на екрані авторизації / реєстрації (рис. 4.2), контролер повинен відповісти, оновивши інтерфейс та представивши нові компоненти на екрані (приклад 4.3).

```
private func presentPopupController(with labelTitle: String) {
    let titleLabel = LoginWithLabel()
    let descriptionLabel = LoginDescriptionTextView()
    titleLabel.text = labelTitle
    let facebookButton = LoginWithFacebookButton(width: view.frame.width - 16)
    facebookButton.selectionHandler = loginWithFacebookTapped
    popupController = CNPPopupController(contents: [titleLabel, facebookButton, descriptionLabel])
    popupController.theme.shouldDismissOnBackgroundTouch = true
    popupController.theme.maxPopupWidth = view.frame.width - 16
    popupController.present(animated: true)
}
```

Приклад 4.3 – Метод контроллера, що викликається у відповідь на дії користувача

Крім того, контролер виконує ще одну життєздатну функцію - надання даних компонентам інтерфейсу. У більшості випадків інтерфейс конкретного

екрана відображає пов'язані дані (рис. 4.4).

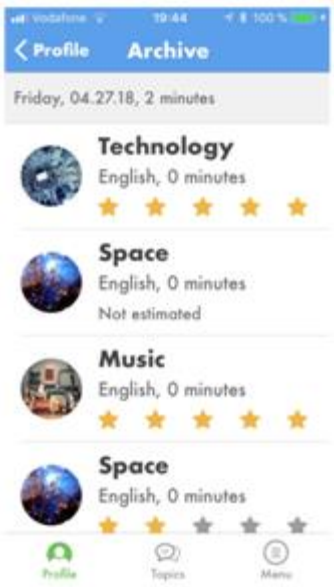

Рисунок 4.4 – Екран додатку, що відображає історію практик користувача

Завданням контролера в цьому контексті є з'єднання даних з компонентами інтерфейсу. Іноді дані, що відображаються, спочатку потрібно отримати з бази даних або завантажити з Інтернету та відформатувати певним чином. Цю роботу виконує координатор (рис. 4.1). Цей клас реалізований для кожного контролера і містить методи виконання бізнес-логіки. Наприклад, координатор реалізує сортування тем, пропонованих користувачеві, на основі пов'язаних асоціативних правил, отриманих з аналізу центральної бази даних користувачів.

4.1.1 Реалізація підсистеми реєстрації та авторизації

Для доступу до функцій програми користувач повинен спочатку зареєструвати обліковий запис. Для спрощення цього процесу механізм

реєстрації реалізується через соціальні мережі (рис. 4.5).

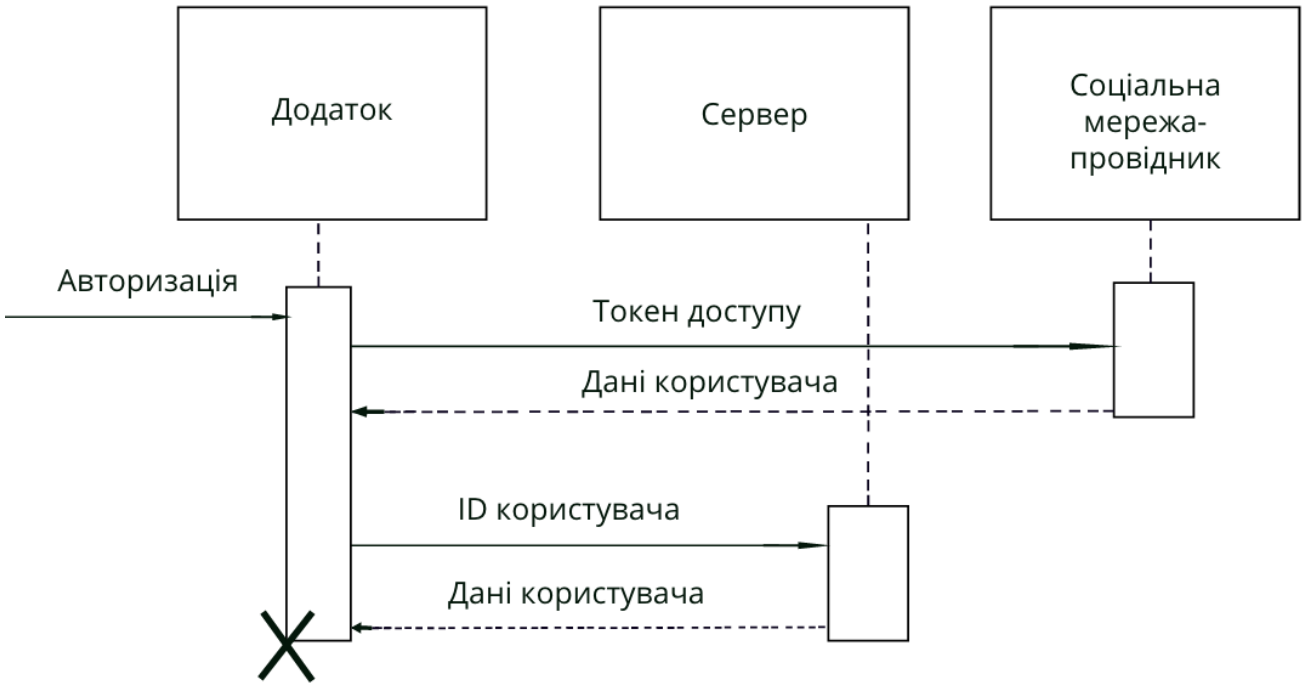

Рисунок 4.5 – Діаграма послідовності процесу авторизації

По-перше, під час входу в систему потрібно перевірити, чи користувач уже зареєстрований. Для цього потрібно зробити запит до бази даних, надіславши на сервер унікальний ідентифікатор користувача - в додатку це ідентифікатор користувача, отриманий від провідника соціальної мережі. Тому перший крок у процесі авторизації - це надіслати запит до такої соціальної мережі з метою отримання можливих даних (приклад 4.4).

```
func loginWithFacebook(with responce: @escaping (LoginCoordinatorResponce) -> Void) {
    FBClient.shared().delegate = self
    FBClient.shared().getFacebookUser(from: controller, success: { (facebookUserData) in
        if let facebookUser = facebookUserData {
            let facebookID = facebookUser["id"] as! String
```

Приклад 4.4 – Отримання даних користувача з соціальної мережі-провідника

Отримавши дані користувача, які включають, зокрема, унікальний ідентифікатор, ви можете виконати наступну частину процесу - відправлення запиту на сервер додатків (рис. 4.5). За умови, що користувач вже зареєструвався, його обліковий запис знайде отриманий ідентифікатор. Усі його збережені дані будуть надсилатися додатково - процес авторизації закінчений.

Якщо користувач ще не зареєстрований, до процесу додається кілька додаткових кроків (рис. 4.6).

Перш за все, при отриманні повідомлення від сервера про те, що користувач не знайдений, модуль авторизації передає управління модулем реєстрації - це означає представлення відповідного інтерфейсу, де користувач може вводити або редагувати раніше отримані з соціальних даних дані мережевий провідник. Крім того, мій користувач може вибирати мови, якими він хоче займатися.

Після збору даних і переконання, що вони введені правильно, програма надсилає їх на сервер, де формує новий обліковий запис користувача (рис. 4.6). Після успішного завершення цієї процедури програма отримує повідомлення про збереження нового користувача, а також унікальний ідентифікатор створеного облікового запису - процедура реєстрації завершена.

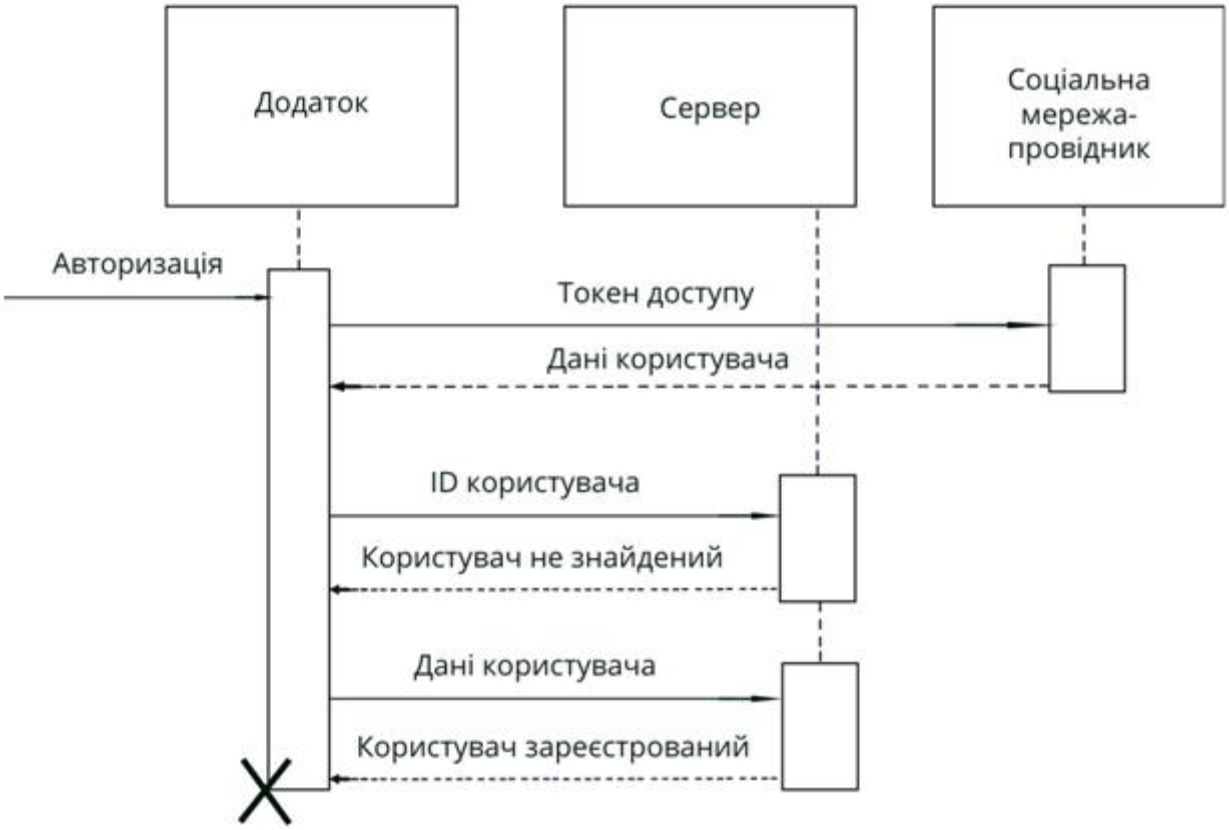

Рисунок 4.6 – Діаграма послідовності процесу реєстрації

Після проходження послідовності авторизації або реєстрації користувач опиняється в головному модулі програми, де відображаються його особисті дані, статистика та надається можливість вибору теми, щоб почати практикувати мови.

Варто зазначити, що з точки зору безпеки головну роль у реалізованій схемі авторизації / реєстрації відіграє доступ користувача до його облікового запису в соціальній мережі - всі інші деталі аутентифікації передаються їм.

4.1.2 Реалізація головного модуля додатку

Основний модуль програми - відповідає за основну функціональність

програми і складається з декількох екранів (додаток В). На екрані профілю користувач може переглядати інформацію про свій обліковий запис, а також оновлювати його, перейшовши на екран налаштувань. Крім того, користувач може перейти на екран «архіву» та переглянути історію всіх своїх минулих практик, а також оцінки, які він отримав за них від партнерів (рис. 4.4).

Основна функціональність програми - використання мов з обраними партнерами через вбудований механізм Інтернет-голосових дзвінків починається з вибору теми - для цього є екран "Теми", на якому відображаються теми, запропоновані користувачеві для розмова. Перелік тем, що відображаються на цьому екрані, формується з урахуванням асоціативних правил, інтегрованих у додатку, які слід спочатку отримати шляхом аналізу бази даних за допомогою методів асоціативного навчання.

4.2 Знаходження асоціативних правил методом асоціативного навчання

Щоб отримати правила з бази даних, потрібно спочатку вибрати технологію, яка дозволить вам використовувати алгоритми для вивчення правил асоціативного навчання. Дані, що містяться в базі даних, мають в основному числові атрибути - кількість тем, обраних кожним користувачем. Крім того, існують і категоричні ознаки - характеристики самих користувачів, такі як вік, стать, країна походження.

Тому необхідно вибрати алгоритм, який повинен бути оптимізований для числових і категоричних атрибутів - з розділу 1 видно, що ідеальним кандидатом є алгоритм QuantMiner, оскільки він дозволяє використовувати шаблони правил для цих типів даних.

Першим етапом відбору асоціативних правил є підготовка даних. Алгоритм QuantMiner може обробляти файли у двох форматах: csv та db [3]. Слід зазначити, що для зручності та ясності формат csv є більш вигідним,

тому дані були переведені у цей формат (рис. 4.7). На рисунку 4.7 показано зразок даних, згенерованих тестовими користувачами програми - кожен рядок відповідає профілю одного користувача та показує, скільки разів він вибрав тему для практики.

Technology	Space	Music	Books	Movies	Video games	Sport	Travel	Learning	Cooking	Age	Gender
45	34	5	39	1	5	3	4	12	2	24	male
19	36	4	22	1	3	1	1	28	4	34	male
14	17	3	32	3	3	2	5	20	1	36	female
54	54	2	18	2	3	5	3	30	2	39	male
36	19	2	33	4	4	2	5	48	4	51	female
17	36	4	22	2	5	5	2	15	4	54	male
36	34	1	14	3	3	3	2	12	1	15	female
33	17	5	10	3	1	1	5	30	3	11	male
36	34	2	10	3	4	2	1	18	2	48	female
17	12	3	45	4	2	4	1	38	5	33	male
34	34	3	32	4	5	2	4	26	1	17	male
36	28	5	48	2	4	1	1	12	2	12	male
54	13	3	32	5	2	5	4	10	3	36	male
32	30	4	42	4	1	1	1	22	1	51	female

Рисунок 4.7 – Вибірка даних в форматі csv

За допомогою даних тесту можна безпосередньо використовувати алгоритм QuantMiner. Для вибору асоціативних правил необхідно вказати шаблон правила [3] - тобто вказати бажані атрибути для лівої та правої сторін правила (рис. 4.8).

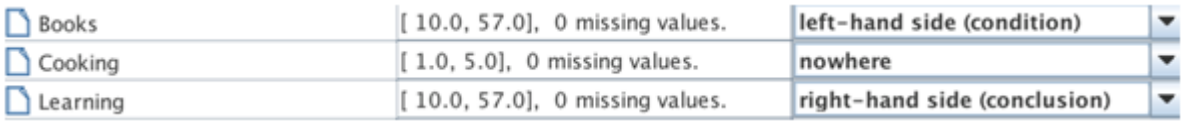

Рисунок 4.8 – Задання шаблону правила в QuantMiner

Встановивши шаблон правила, необхідно визначити такі параметри, як мінімальне значення підтримки, мінімальне значення довіри (рис. 4.9), після чого алгоритм почне аналізувати дані.

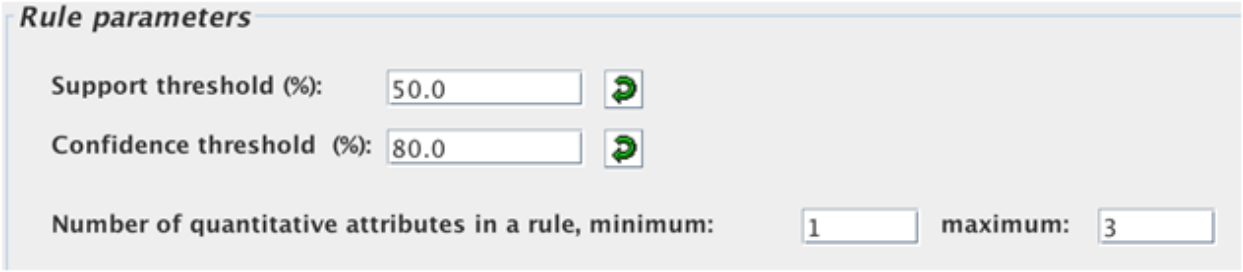

Рисунок 4.9 – Задання параметрів пошуку

Після завершення аналізу даних алгоритм QuantMiner видає правила, що відповідають усім раніше встановленим вимогам (рис. 4.10).

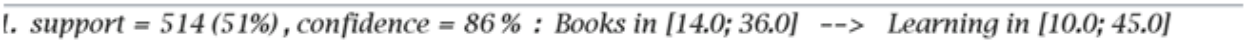

Рисунок 4.10 – Приклад знайденого асоціативного правила

Як видно з рисунку 4.10 у тестовому зразку із заданим шаблоном та заданими параметрами, було знайдено асоціативне правило, що ілюструє залежність атрибута Books та атрибута Learning. У цьому випадку ви бачите, що користувачі, які часто обирали практикувати тему Книги, також часто

обирали тему Навчання.

Якщо ви зміните шаблон і встановите атрибут Technology в його лівій частині, а атрибут Cooking - у правій частині, зберігаючи всі раніше встановлені параметри, алгоритм QuantMiner дасть нове знайдене правило (рис. 4.11).

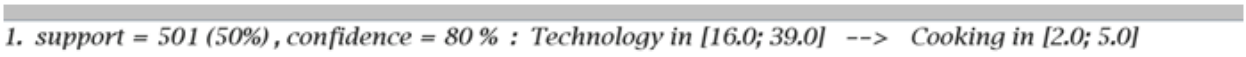

Рисунок 4.11 – Приклад знайденого асоціативного правила

У цьому випадку знайдене правило свідчить про іншу залежність - користувачі, які цікавляться темою технології, майже не вибирають тему приготування.

Варто зазначити, що виділення правил за допомогою лише одного шаблону одночасно може зайняти багато часу, але QuantMiner дозволяє вказувати відразу кілька шаблонів. У цьому випадку кількість знайдених асоціативних правил різко зростає (рис. 4.12).

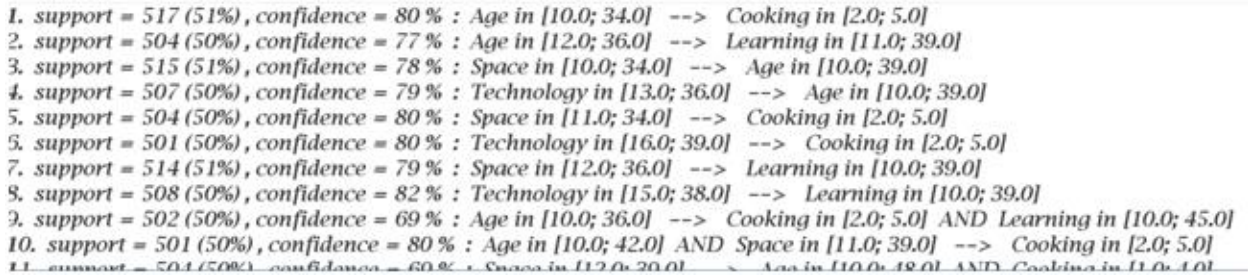

Рисунок 4.12 – Група знайдених асоціативних правил.

Як видно з рисунку 4.12, QuantMiner також може виділити складні

правила - з кількома атрибутами в лівій або правій частинах.

Використовуючи алгоритм QuantMiner на вибірці тестових даних, описаній вище, ви можете отримати необхідні асоціативні правила, що дозволяють оцінити взаємозв'язки між темами, вибраними користувачами. Наступним кроком є використання порушених асоціативних правил у самій програмі, для здійснення інтелектуального підбору тем.

Для цього спочатку потрібно зберегти знайдені правила в окремому файлі, який слід включити до ресурсів програми (рис. 3.2). Таким чином координатор зможе отримати доступ до бази даних правил через відповідний інтерфейс (рис. 4.1). Крім того, координатор може отримати дані з інтегрованої бази даних - шляхом визначення атрибутів, що характеризують користувача. Після цього залишається лише використовувати алгоритм пошуку асоціативних правил, які відповідали б параметрам користувача, та відповідно сформувати список запропонованих тем (приклад 4.5).

```
private func performViewModel(languageID: String) {
    if let rules = LGVResourcesManager.getAssociativeRules() {
        if let topics = LGVResourcesManager.getLocalizedTopics() {
            for (index, topic) in topics.enumerated() {
                let topicImage = UIImage(named: "Topic_\(index)")!
                let topicTitle = topic["title"]!
                let topicDescription = topic["description"]!
                viewData.append(TopicsTopicViewModel(id: index, picture: topicImage, title: topicTitle, topicDescription: topicDescription))
                arrange(viewData, with: rules)
            }
        }
```

Приклад 4.5 – Формування списку тем

Функція розташування (приклад 4.5) відповідає за прямий пошук правил серед списку, що виходить з виклику getAssociativeRules. Таким чином, список тем, пропонованих користувачеві, формується з урахуванням його раніше вибраних тем та деяких інших характеристик. Це також означає, що, як ви продовжуєте використовувати додаток, оскільки дані конкретного користувача змінюються, список тем також змінюватиметься (додаток В).

Використовуючи методи асоціативного навчання, таким чином, вдалося вирішити важливу проблему для забезпечення оптимальної роботи програми в одній з найважливіших її частин.

4.3 Реалізація модуля голосових дзвінків

Після того як користувач вибирає тему для розмови, програма перевіряє, чи є в базі даних відповідний партнер - для цього всі необхідні параметри надсилаються на сервер. Якщо партнера не знайдено, сервер чату програми створює нову запис, що містить інформацію про потрібний формат розмови. Якщо партнер знайдений - його додаток повертається до програми, після чого ви можете здійснити голосовий дзвінок. Отримавши необхідні дані про партнера, контроль переходить до модуля бесіди (приклад 4.6).

Координатор цього модуля забезпечує функцію голосових викликів за допомогою інтегрованого набору інструментів розвитку Sinch, який, в свою чергу, взаємодіє з хмарною платформою.

```
topicsCoordinator.acquire { (response) in
    switch response {
    case .error:
        UIApplication.shared.endIgnoringInteractionEvents()
        SVProgressHUD.showError(withStatus: NSLocalizedString("Topics.AcquireError", comment: "Error"))
    case .notFound:
        UIApplication.shared.endIgnoringInteractionEvents()
        SVProgressHUD.dismiss()
        self.presentAcquiredPopUp()
    case .found(let acquisitionData):
        UIApplication.shared.endIgnoringInteractionEvents()
        SVProgressHUD.dismiss()
        self.router.presentCallModule(from: self, with: acquisitionData)
    }
```

Приклад 4.6 – Передача керування до модуля розмови

Для спрощення використання SDK Sinch була створена оболонка, яка з архітектурної точки зору знаходиться на рівні послуг (рис. 3.2). Оскільки

Sinch SDK реалізований в мові програмування Objective-C, оболонка навколо нього також повинна бути створена на цій мові.

Завантажуючи модуль бесіди, його координатор активує функцію Sinch через відповідний створений інтерфейс.

Перша операція полягає у виклику методу ініціювання виклику (приклад 4.7).

```
- (void) makeCallTo: (NSString *) contact
            topicID: (NSString *) topicID
         languageID: (NSString *) languageID
             userID: (NSString *) userID
             talkID: (NSString *) talkID
      acquisitionID: (NSString *) acquisitionID;
```

Приклад 4.7 – Інтерфейс ініціації дзвінка

Для управління поточним викликом також був створений протокол, підтримуючи який координатор цього модуля може отримувати відповідні повідомлення. Після того, як дзвінок був надісланий та отриманий іншою стороною, координатор отримує повідомлення "talkStarted". У відповідь на це повідомлення координатор оновлює інтерфейс користувача, запускає таймер виклику.

Якщо дзвінок було відхилено, координатор отримає повідомлення "talkDenied", в кінці дзвінка - "talkEnded".

Крім оновлення інтерфейсу, у відповідь на повідомлення менеджера викликів координатор також використовує деякі методи серверної частини програми для оновлення стану розмови в центральній базі даних.

На початку розмови слід створити відповідний запис у базі даних (приклад 4.8).

```
func talkStarted() {
    APIClient.shared.talkAdd(userID: Int(DataManager.shared().objectGrapth.userID), talkID:
        controller.callData.talkID)
    if controller.callData.callState == .caller {
        APIClient.shared.topicDelete(acquisitionID: controller.callData.acquisitionID)
    }
    timer = Timer.scheduledTimer(timeInterval: 1, target: self, selector:
        #selector(CallTalkCoordinator.updateTimer), userInfo: nil, repeats: true)
    getPartnerData()
}
```

Приклад 4.8 – Метод відповіді координатора модуля розмови на помідомлення про початок розмови

В кінці розмови вам потрібно оновити раніше збережену інформацію новими даними. Це необхідно насамперед для збереження тривалості дзвінка, яку, очевидно, можна визначити лише після його завершення (приклад 4.9).

```
func talkEnded() {
    timer.invalidate()
    if controller.callData.callState == .caller {
        APIClient.shared.talkUpdate(talkID: controller.callData.talkID, duration: duration)
    }
    controller.performEndView()
    controller.presentSummaryView()
}
```

Приклад 4.9 – Метод відповіді координатора модуля розмови на помідомлення про завершення розмови

Крім того, подібні методи із специфічною логікою існують для ще двох сценаріїв припинення виклику - відхилення та скасування.

У разі відхилення необхідно видалити всю інформацію про розмову, що була надіслана та збережена раніше (приклад 4.10).

```
func talkDenied() {
    if controller.callData.callState == .caller {
        APIClient.shared.topicDelete(acquisitionID: controller.callData.acquisitionID)
        APIClient.shared.talkDelete(talkID: controller.callData.talkID)
        controller.performEndView()
        controller.presentDeniedView()
    }
}
```

Приклад 4.10 – Метод відповіді координатора модуля розмови на помідомлення про відхилення розмови

Якщо користувач ініціював дзвінок, а потім чомусь передумав, координатор отримає повідомлення про скасування дзвінка.

У відповідь на це повідомлення буде викликано два способи серверного інтерфейсу, які повинні повернути базу даних у стан, в якому вона була до початку виклику (приклад 4.11).

```
func talkCancelled() {
    if controller.callData.callState == .caller {
        APIClient.shared.topicLockout(acquisitionID: controller.callData.acquisitionID)
        APIClient.shared.talkDelete(talkID: controller.callData.talkID)
        controller.performEndView()
```

Приклад 4.11 – Метод відповіді координатора модуля розмови на помідомлення про відміну розмови

Основне завдання розробленого додатку - дати можливість користувачеві знайти партнера і забезпечити можливість голосових дзвінків для мовної практики. Однією з основних концепцій програми є анонімність користувачів щодо своїх партнерів до початку розмови.

Після початку розмови користувачі отримують певну особисту

інформацію про свого партнера - його фото, ім’я та деякі статистичні дані.

Розмова закінчиться, як тільки один із користувачів натисне відповідну кнопку.

У цьому циклі, який складається з етапів вибору теми, пошуку партнера і розмова починається з початку.

В кінці розмови користувачі можуть оцінювати один одного за допомогою п'ятибальної шкали. Пізніше користувач може переглянути отриману оцінку на екрані «архіву», де, крім того, також можна побачити тривалість та інші деталі всіх минулих розмов.

ВИСНОВКИ

У цій роботі показано, як використання методів штучного інтелекту може оптимізувати продуктивність програмного забезпечення та покращити роботу користувачів.

Був проведений детальний аналіз такої галузі обміну даними, як асоціативне навчання. Цей метод дозволяє викривати приховані шаблони у великих масивах даних, що показують зв’язки між атрибутами.

Аналіз різних алгоритмів асоціативного вивчення правил, що працюють з різними типами даних - булевими, категоричними, числовими.

Результатом аналізу став вибір одного з алгоритмів навчання - QuantMiner, який використовує генетичні алгоритми та працює на основі шаблонів - форматів правил, які містять цільові атрибути з лівого та правого боку правила.

Проведено аналіз технології голосових дзвінків через Інтернет. Досліджено протоколи, які можна використовувати для реалізації цієї технології. Крім того, було проведено аналіз доступних рішень, таких як бібліотеки з відкритим кодом або хмарні платформи, які значно полегшують реалізацію функціональних можливостей голосових дзвінків через Інтернет.

Була розроблена інформаційна система, що складається з декількох незалежних компонентів - додатку для мобільних пристроїв, серверної частини та центральної бази даних. За допомогою цієї системи користувачі можуть автоматично знаходити партнерів для занять іноземними мовами та безпосередньо практикувати вбудовану функціональність голосових дзвінків.

За допомогою алгоритму QuantMiner було проаналізовано базу даних розробленої інформаційної системи. У цьому аналізі були виявлені приховані зв’язки між вибором тем, зробленими користувачами. Отримані правила дозволили вирішити одне з важливих завдань у розробці додатку -

формування списку тем, пропонованих користувачеві для практики.

Таким чином, асоціативні правила відображають закономірності вибору різних тем, а також деякі характеристики самих користувачів.

Ця робота продемонструвала, як програмні продукти можуть отримати користь від використання методів штучного інтелекту. Крім того, враховуючи сучасний темп розвитку галузі, для високих вимог користувачів використання таких методів вже насправді є необхідністю.

ПЕРЕЛІК ПОСИЛАННЬ

1. Language exchange // Вільна інтернет-енциклопедія Wikipedia. URL : https://en.wikipedia.org/wiki/Language_exchange. (Дата звернення 20.05.20).

2. Tanenbaum A. S. Computer Networks. Amsterdam : Vrije Universiteit, 2011. 905 с.

3. Johnson A. Understanding the Session Initiation Protocol. Norwood, 2016. 493 с.

4. PJSIP – Documentation. URL : https://trac.pjsip.org/repos. (Дата звернення 01.06.20).

5. Sinch – Documentation. URL : https://www.sinch.com/docs/voice/ios/.(Дата звернення 02.06.20).

6. Tandem – Інформація з сайту платформи. URL : https://www.tandem.net.(Дата звернення 03.06.20).

7. HelloTalk – Інформація з сайту платформи. URL : https://www.hellotalk.com.(Дата звернення 03.06.20).

8. Agrawal R., Imielinski T., Swami A. Mining associating rules between sets of items in large databases // Conference on Management of Data. Washington, 1993. P. 207–216.

9. Agrawal R., Imielinski T., Swami A.Mining Associating Rules between Sets of Items in Large Databases. IBM Almaden Research Center, 1993. 10 p.

10. Ansaf S. QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules. Columbia University, 2007. 6p.

11. Mata J., Alvarez J.L., Riquelme J.C. Mining Numeric Association Rules with Genetic Algorithms. Prague, 2001. P. 264 – 267.

12. A Statistical Theory for Quantitative Association Rules // Journal of Intelligent Information Systems. 2007. 83 p.

13.Takeshi F.Mining Optimized Association Rules for Numeric Attributes.

Tokyo Research Laboratory, IBM Research, 1996. P. 12–23.

14. Vapor –Інформація з сайту фреймворка.URL: https://vapor.codes(Дата звернення 02.06.20).

15. Comparison of the Swift frameworks.URL: https://www.netguru.co/codestories/server–side–swift–frameworks–comparison(Дата звернення 02.06.20).

16. Принцип єдиного обов’язку.URL: ttps://uk.wikipedia.org/wiki/Принцип_єдиного_обов%27язку(Дата звернення 02.06.20)