КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота
на здобуття освітнього рівня бакалавра
за спеціальністю 121 Інженерія програмного забезпечення
на тему:

РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ ВИВЧЕННЯ АНГЛІЙСЬКОЇ МОВИ, ВИКОРИСТОВУЮЧИ ХМАРНУ БАЗУ ДАНИХ FIREBASE

Виконала студентка 4-го курсу
Таїсія ФЕНЗ _____
(підпис)

Науковий керівник:
доцент, кандидат фіз.-мат. наук
Максим ВЕРЕС _____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студентка _____
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри інтелектуальних програмних систем «25» травня 2022 р.,

протокол № 10

Завідувач кафедри

Олександр ПРОВОТАР _____
(підпис)

Київ - 2022

РЕФЕРАТ

Кваліфікаційна робота «Реалізація веб-застосунку для вивчення англійської мови, використовуючи хмарну базу даних Firebase» складається з переліку умовних скорочень, вступу, основної частини, що містить 3 розділи, висновків і списку використаних джерел. Загальний обсяг роботи – 45 сторінки. Робота містить 19 рисунків та 2 таблиці.

У кваліфікаційній роботі проводиться опис сучасних Frontend технологій та дослідження можливостей платформи Firebase. Метою роботи було, розробити веб-додаток для індивідуального вивчення англійської мови на їх основі. Було використано базу даних реального часу Firebase realtime data base. Передбачається, що застосунок буде використовуватись студентами на викладачами для полегшення їх комунікації.

Ключові слова: Angular, Firebase, Realtime DB, «EasyEnglish», мережа Петрі.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

ПЗ	Програмне забезпечення;
HTML	(HyperText Markup Language) – мова розмітки гіпертексту;
CSS	(Cascading Style Sheets) – каскадні таблиці стилів;
VSCode	Visual Studio Code – середовище розробки;
IDE	(Integrated development Environment) – інтегроване середовище розробки;
БД	База даних;
DB	(Data Base) – база даних;
JS	JavaScript – мова програмування;
DOM	(Document Object Model) – об'єктна модель документа;
XML	(Extensible Markup Language) – розширювана мова розмітки;
MVC	(Model-View-Controller) – модель-представлення-контролер;
SSL	(Secure Sockets Layer) – рівень захищених сокетів;
HTTPS	(HyperText Transfer Protocol Secure) – захищений протокол передачі гіпертексту;

ВСТУП

У часи глобалізації стає дедалі зрозуміліше, що знання міжнародної мови, це не просто побутове бажання, а нагальна потреба. Це пов’язано з тим що, більшість технічної документації, міжнародних новин, важливих документів створюється саме англійською мовою. Попит на вивчення та покращення рівня мови, створює безліч опцій для вирішення цієї проблеми – як у класах з викладачем, так і за допомогою технічних пристроїв. Не дивлячись на переваги живого спілкування, події у сучасному світі дають зрозуміти наскільки важливим є перехід до онлайн навчання. З розвитком технологій кількість платформ які пропонують такий формат освіти неухильно зростає. Чат-боти, програми на комп’ютер, мобільні застосунки, і звичайно ж – веб-додатки. Останні здобули особливого розповсюдження, оскільки доступні по посиланню у веб-браузері і не потребують встановлення додаткового ПЗ. Саме тому постає питання про створення нового веб-застосунку для вивчення англійської мови з використанням передових технологій.

Мета роботи: ознайомлення з сучасними Frontend технологіями та дослідження можливостей платформи Firebase. Розробка веб-додатку для вивчення іноземної мови на їх основі.

Завдання:

- підбір та аналіз актуальних Frontend технологій;
- огляд можливостей використання Firebase;
- дослідження особливостей і переваг впровадження та використання Realtime Firebase DB;
- опис і документування концепції та можливостей продукту, розробка веб-застосунку на їх основі;
- опис сценаріїв додатку за допомогою мереж Петрі.

Актуальність:

- використання сучасних та прогресивних веб-технологій, які дозволяють створювати проект у відповідності з основними принципами програмування;
- використання строгої типізації та реактивного програмування;
- інтеграція з Backend-as-a-Service (BaaS) платформою Firebase;
- впровадження Realtime Data Base;
- програма створена для запровадження індивідуального вивчення англійської мови; автоматизує та полегшує взаємодію студента та вчителя.

РОЗДІЛ 1

FRONTEND ТЕХНОЛОГІЇ ДЛЯ РОЗРОБКИ ВЕБ-ЗАСТОСУНКУ

Для розробки веб-застосунку було використано наступні технології:

- HTML для верстки сторінок веб-застосунку, SASS для створення каскадних таблиць стилів;
- JavaScript, TypeScript – залучені мови програмування;
- фреймворк[1] Angular 13-ї версії;
- бібліотека RxJS для створення запитів до БД;
- бібліотека компонент Angular Material UI.

Проект створювався за допомогою Angular CLI – інструменту інтерфейсу командного рядка, який програміст використовує для ініціалізації, розробки, зборки та підтримки Angular застосунків безпосередньо з командного рядка. Середовищем розробки обрала VSCode, тому що це швидка і безкоштовна IDE, яка пропонує своїм користувачам безліч функцій для полегшення написання коду.

1.1. HTML, SASS

Мова розмітки гіпертексту або HTML є стандартною мовою розмітки для документів, призначених для відображення у веб-браузері. Цьому можуть допомогти такі технології, як каскадні таблиці стилів (CSS) і мови скриптів[2], такі як JavaScript.

CSS – це спеціалізована мова стилю сторінок, яка використовується для опису зовнішнього вигляду сторінок. CSS потужний інструмент, але таблиці стилів стають більшими, складнішими та їх стає складніше підтримувати. Саме такі проблеми допомогають вирішувати препроцесори. Один із них – SASS [2],

[1] Фреймворк (англ. Framework, платформа, каркас, структура) – інфраструктура програмних рішень, яка полегшує розроблення складних систем.

[2] Скрипт (англ. Script) – програма, що автоматизує якесь завдання, яке б без скрипту користувач робив би вручну.

має функції, яких ще не існує в CSS, таких як, вкладення, наявність змінних, успадкування та інші чудові переваги, які допомагають писати надійний, зручний для підтримки CSS.

Отже, SASS – це метамова на основі CSS, призначена для збільшення рівня абстракції CSS коду та спрощення файлів каскадних таблиць стилів.

1.2. JavaScript, TypeScript

JavaScript (JS) [3] — динамічна, об'єктно-орієнтована прототипна мова програмування, реалізація стандарту ECMAScript[1]. Найчастіше використовується для створення скриптів веб-сторінок, які надають можливість взаємодіяти з користувачем на боці клієнта (пристрої кінцевого користувача), обмінюватися даними з сервером асинхронно, керувати браузером, змінювати зовнішній вигляд та структуру веб-сторінки. Він дотримується правил програмування на стороні клієнта, тому працює у веб-браузері користувача без потреби в будь-яких ресурсах веб-сервера.

TypeScript — це сучасний JavaScript. Це статично скомпільована мова для написання зрозумілого та простого коду JavaScript. Його можна запустити на Node js або будь-якому браузері, який підтримує ECMAScript 3 або новіші версії. TypeScript надає додаткову статичну типізацію, класи та інтерфейс.

Ключова різниця між JavaScript і TypeScript:

- JavaScript – це мова скриптів, яка допомагає створювати інтерактивні веб-сторінки, тоді як TypeScript — це JavaScript плюс додаткові можливості;
- код TypeScript потрібно компілювати, а код JavaScript – ні;
- порівнюючи TypeScript і JS, TypeScript підтримує функцію прототипування, в той час як JavaScript не має такої переваги;

[1] ECMAScript – це стандарт JavaScript, призначений для забезпечення сумісності веб-сторінок у різних веб-браузерах.

- TypeScript використовує такі поняття, як типи та інтерфейси для опису даних, тоді як у JavaScript такої концепції не має;
- TypeScript – це потужна система типів, що включає загальні для всіх мов програмування функції та функції JS для проектів великого і середнього розміру, тоді як JavaScript — ідеальний варіант для невеликих програм.

З цього випливає що, для проекту з багатьма сторінками, запитами до БД та високою необхідністю обробляти та типізувати отримані результати, застосування TypeScript може дати більш надійний процес розробки та можливість легко підтримувати та нарощувати програмний продукт в майбутньому.

1.3. Загальний огляд Angular

Фреймворком для розробки застосунку було обрано Angular [4]. Angular – це платформа та фреймворк для створення односторінкових клієнтських програм, що використовують HTML і TypeScript. Angular написаний на TypeScript. Він реалізує «core»[1] функціонал та надає можливість підключати додаткові функції у вигляді набору бібліотек TypeScript, які розробник імпортує у свою програму.

Як платформа, Angular включає:

- компонентний фреймворк для створення масштабованих веб-додатків;
- колекція добре інтегрованих бібліотек, які охоплюють широкий спектр функцій, включаючи маршрутизацію, керування формами, зв’язок клієнт-сервер, тощо;
- набір інструментів для розробників, які допоможуть вам розробляти, створювати, тестувати та оновлювати код;

[1] Core – укр. ядро, осердя, серцевина

Основною причиною для використання фреймворку є те, що вони загалом підвищують ефективність і продуктивність веб-розробки, забезпечуючи послідовну структуру, щоб розробникам не доводилося перебудовувати код з нуля. Фреймворки заощаджують час і пропонують розробникам безліч додаткових функцій, які можна додати до програмного забезпечення без зайвих зусиль.

JavaScript є найбільш часто використовуваною мовою програмування на стороні клієнта. Він записаний у документи HTML, щоб забезпечити взаємодію з веб-сторінками багатьма унікальними способами. Як відносно проста для вивчення мова з гарною підтримкою коду, вона добре підходить для розробки сучасних програм. Але, не дивлячись на вищеперераховані переваги, JavaScript не підходить для розробки односторінкових програм, які вимагають модульності, можливості якісно тестувати код та максимальної продуктивності розробників.

Сьогодні було створено різноманітні фреймворки та бібліотеки, розроблені для надання альтернативних рішень. Що стосується інтерфейсної веб-розробки, Angular вирішує багато, якщо не всі проблеми, з якими розробники стикаються під час самостійного використання JavaScript.

«Angular» — це загальний термін для різних версій фреймворку. Angular був розроблений в 2009 році, і в результаті було створено багато версій.

З самого початку існування ідеї цього фреймворку був створений оригінальний Angular, який називався Angular 1 і з часом він став відомий більше як AngularJS. Потім з’явилися нові версії, такі як - Angular 2, 3, 4, 5, 11, 12 поки, нарешті, поточна версія, Angular 13, випущена 04.11.2021. Найважливіша відмінність між AngularJS та Angular пізніших версій полягає в тому, що AngularJS заснований на JavaScript, а Angular - на TypeScript. При цьому, кожна наступна версія Angular покращує свою попередню, виправляючи помилки,

вирішуючи проблеми та пристосовуючись до зростаючої складності сучасних платформ.

1.4. Особливості та переваги Angular

До особливостей Angular [5] [6] можна віднести:

а) Об'єктна модель документа.

DOM (об’єктна модель документа) розглядає XML або HTML-документ як деревовидну структуру, в якій кожен вузол представляє частину документа (Рисунок 1.1).

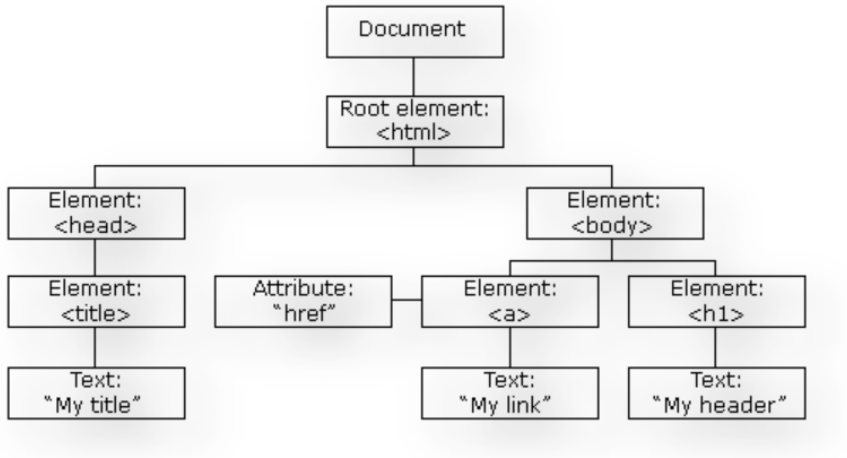


Рисунок 1.1 — Приклад DOM [5]

Angular використовує звичайний DOM. Під час розробки треба враховувати, що якщо було зроблено декілька оновлень, то вони будуть застосовані на одній HTML-сторінці. Замість того, щоб змінювати ті, які вже були зачіплені попередніми змінами, Angular оновить всю деревовидну структуру тегів HTML.

б) TypeScript.

TypeScript визначає набір типів для JavaScript, що допомагає користувачам писати код JavaScript, який легше зрозуміти. Весь код TypeScript компілюється за допомогою JavaScript і може безперебійно працювати на будь-якій платформі. TypeScript не є обов'язковим для

розробки програми Angular. Однак це стала рекомендація, оскільки він пропонує кращу синтаксичну структуру, а також полегшує розуміння та подальше обслуговування коду.

в) Прив'язка даних.

Прив’язка даних – це процес, який дозволяє користувачам маніпулювати елементами веб-сторінки через веб-браузер. Він використовує динамічний HTML і не вимагає написання складних скриптів для обробки подій, які створює користувач. Прив’язка даних використовується на веб-сторінках, які містять інтерактивні компоненти, такі як кнопки, калькулятори, навчальні посібники, форуми та ігри. Це також дозволяє краще відображати веб-сторінку, коли вона містять велику кількість даних.

Angular використовує двостороннє прив’язування. Моделі в коді відображають будь-які зміни, внесені у відповідні елементи інтерфейсу користувача. І навпаки, елементи UI відображають будь-які зміни в стані моделі. Такий функціонал дозволяє фреймворку підключати DOM до даних моделі через контролер.

г) Renderer Ivy встановлений за замовчуванням.

Новий двигун Angular налічує безліч покращень, таких як оптимізовані розміри пакетів і швидше завантаження компонентів. З Ivy renderer компанії можуть отримати незрівнянний процес дебагінгу[1] коду та зручну роботу з додатком. Крім того, це зменшує розміри файлів, що робить фреймворк більш доступним і надає можливість програмістам ще простіше додавати новий функціонал для розробки складних додатків.

[1] Дебагінг (англ. debugging) – налагодження програми, процес пошуку та зменшення числа дефектів або помилок у програмі з метою отримання очікуваної поведінки.

д) Декларативний інтерфейс користувача.

Фреймворк Angular використовує HTML, який у порівнянні з JavaScript є менш складною мовою. HTML також популярний як декларативна та інтуїтивно зрозуміла мова, яка позбавляє від необхідності витрачати багато часу на виконання програм і планування того, що завантажувати першим. Розробники Angular просто намічають, що потрібно відобразити, а все інше робить фреймворк.

е) Підтримується Google.

Google є однією з найбільших технологічних компаній, має талановиту команду розробників Google і пропонує довгострокову підтримку (LTS) для Angular, щоб розширити розробку корпоративних програм Angular. Крім того, багато великих брендів, таких як Netflix, Gmail, YouTube TV, Upwork та інші, використовують фреймворк Angular.

Саме ці переваги відіграли вирішальну роль у виборі фреймворку для створення проекту.

1.5. Архітектура Angular

Angular — це повноцінний фреймворк з патерном модель-представлення-контролер (MVC). Він надає чіткі вказівки щодо структурування програми (Рисунок 1.2) та пропонує двонаправлений потік даних, забезпечуючи реальну DOM.

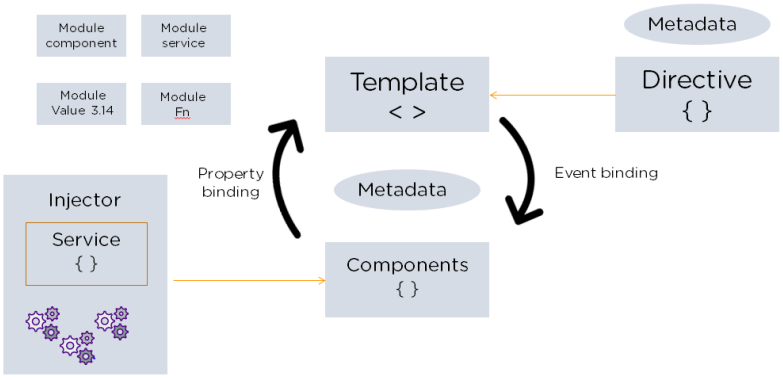


Рисунок 1.2 — Структура Angular проекту [5]

Нижче наведено вісім будівельних блоків програми Angular [7]:

а) Модулі.

Додаток Angular має кореневий модуль під назвою AppModule, який забезпечує механізм завантаження для запуску програми.

б) Компоненти.

Кожен компонент програми визначає клас, який містить логіку програми та дані. Компонент зазвичай визначає частину інтерфейсу користувача (UI).

в) Шаблони.

Шаблон Angular поєднує код Angular з HTML, щоб змінити елементи HTML перед їх відображенням. Існує два типи прив’язки даних:

- прив’язка подій: дозволяє програмі реагувати на поведінку користувача на сайті, оновлюючи дані програми;
- прив’язка властивостей: дозволяє користувачам інтерполювати значення, обчислені з даних програми, у HTML.

г) Метадані.

Метадані повідомляють Angular, як обробляти клас. Вони використовуються для декорування класу, щоб він міг налаштувати очікувану поведінку компоненти.

д) Сервіси.

Якщо у є дані або логіка, які не пов’язані з представленням, але мають бути спільними між компонентами, створюється клас сервісу. Клас завжди підв’язується з декоратором @Injectible.

е) Ін'єкція залежностей.

Ця функція дозволяє підтримувати класи компонентів чіткими та ефективними. Програма не отримує дані із сервера, не перевіряє введені користувачем дані та не реєструється безпосередньо на консолі. Натомість вона делегує такі завдання сервісам.

1.6. Додаткові бібліотеки

Реактивне програмування — це парадигма асинхронного програмування, що спеціалізується на потоках даних і поширенню змін. RxJS (Reactive Extensions for JavaScript) [8] — це бібліотека для реактивного програмування з використанням Observables[1], які полегшують створення асинхронного коду або коду на основі зворотного виклику.

Основні переваги використання RxJS [9]:

- RxJS можна використовувати з іншими бібліотеками та фреймворками Javascript. Він підтримується JavaScript, а також TypeScript. Наприклад часто використовується у зв’язках з Angular, ReactJS, Vuejs, nodejs тощо;
- RxJS — це чудова бібліотека, коли справа доходить до обробки асинхронних завдань. RxJS використовує спостережувані елементи для роботи з реактивним програмуванням, яке має справу з асинхронними викликами даних, зворотними викликами та обробкою подій.
- RxJS пропонує величезну колекцію операторів у категоріях математики, перетворення, фільтрації, умов, обробки помилок, комбінації, що полегшує життя при роботі з реактивним програмуванням.

[1] Observable – (укр. наглядач) сутність в RxJS, яка представляє собою ідею викликаної колекції майбутніх значень чи подій.

Angular Material [10] — це реалізація Material Design Specification (2014-2017). Ця бібліотека надає набір повторно використовуваних, добре перевірених і доступних компонентів інтерфейсу для розробників Angular.

Отже, було зроблено огляд сучаних Frontend технологій, які будуть використовуватись для створення веб-застосунку «EasyEnglish».

РОЗДІЛ 2

ОСОБЛИВОСТІ ХМАРНОЇ БАЗИ ДАНИХ FIREBASE

У висококонкурентному середовищі, що розвивається, компанії прагнуть удосконалити свої стратегії залучення та утримання клієнтів для досягнення кращих результатів бізнесу. Вони прагнуть персоналізувати свої маркетингові зусилля, щоб привернути увагу користувачів, перетворити їх у клієнтів, а пізніше і в клієнтів на все життя.

Отже, важливо розуміти, що важливо для споживачів і як компанії можуть привернути їхню увагу. По-перше, це зручний веб-сайт, який забезпечує позитивний досвід клієнтів. Споживачі хочуть отримати доступ до веб-сайту з різних пристроїв, включаючи планшети та смартфони. Створення різних версій веб-сайту змушує бренди вкладати додаткові ресурси в платформи розробки додатків та інструменти аналітики. На щастя, у Google є потужне рішення, яке містить усе вищезазначене в одному місці під назвою Google Firebase.

2.1. Огляд Google Firebase

Рішення Google спрощує всю роботу, починаючи з розробки додатків. Він містить усі необхідні інструменти в одному місці, включаючи хостинг додатків, базу даних та аналітику додатків. Він також надає багато розширених функцій, які можуть вивести керування додатком на новий рівень.

Google Firebase [11] — це програмне забезпечення платформи Backend-as-a-Service (BaaS), яке допомагає розробляти найкращі міжплатформні програми. Це нереляційна (NoSQL) хмарна база даних, що дозволяє авторам додатків зберігати та синхронізувати дані між кількома клієнтами.

Платформа додатків Firebase інтегрується з програмами на Android та iOS, API (інтерфейсом програмного забезпечення) для програм Java, Node.js, Objective-C і JavaScript. Також надає можливість працювати з базою даних, як-от REST, з фреймворками JavaScript, включаючи angular, react, vue.js.

До найвідоміших популярних програми, які використовують Firebase, відносяться [12]:

– Alibaba;
– The New York Times;
– Le Figaro;
– eBay Motors.

Можна виділити 10 основних переваг використання Firebase:

– можна безкоштовно почати працювати з Firebase;
– швидко розвивається;
– платформа для повної розробки додатків;
– підтримується Google;
– розробники можуть зосередитися на розробці інтерфейсу;
– він не потребує серверів;
– пропонує можливості машинного навчання;
– генерує трафік до програм;
– надає можливість моніторингу помилок;
– безпечний.

Платформа Firebase — це сервер, база даних, платформа хостингу та інструмент аутентифікації в єдиному рішенні. Отже, основні функції Firebase [13]:

а) Хмарне сховище Firebase.

Firebase також виконує роль сховище файлів. Firebase Cloud Storage забезпечує надійне завантаження та вивантаження файлів для програми. Google Cloud Storage підтримує хмарне зберігання відео, аудіо чи файлів будь-якого іншого типу. Його система безпеки надійно захищає вміст хмарного сховища.

б) Система аутентифікації Google Firebase.

Розробляти систему аутентифікації щоразу з нуля економічно неефективно. Натомість система аутентифікації Google Firebase Auth працює з паролями та електронними листами. Вона підтримує публічний протокол аутентифікації OAuth 2.0, який використовують такі великі компанії, як Google, Twitter і Facebook. Система автентифікації Google Firebase інтегрується безпосередньо в базу даних.

в) Хостинг.

Firebase Hosting забезпечує швидкий хостинг для веб-програм, вміст кешується в мережах доставки вмісту по всьому світу. Firebase Hosting приймає статичні файли програм, які підтримують JavaScript, HTML, CSS та інші записи. Хмарні функції надають динамічну підтримку Node.js. Передача файлів завершується через мережу доставки вмісту за безпечними протоколами SSL і HTTPS.

г) Firebase Messaging.

Firebase Messaging — це кросплатформне рішення, яке дозволяє надсилати повідомлення на пристрої користувачів із програми. Додаток може надсилати сповіщення на будь-який пристрій, включаючи персональні комп’ютери (ПК). Це можуть бути певні користувачі, групи користувачів або всі пристрої з встановленим додатком. Крім того, є можливість розділити сповіщення на окремі теми.

Опція Firebase Messaging легко масштабується і дозволяє швидко надсилати величезну кількість сповіщень. Розробник також може налаштувати повідомлення, навіть беручи до уваги часовий пояс одержувача.

д) Аналітика для мобільних додатків Firebase Analytics.

Інструмент аналітики мобільних додатків Firebase Analytics служить основою для різних замірів активності у додатку і мобільної аналітики.

Цей інструмент корисно використовувати для того щоб сприяти розвитку як програми, так і бізнесу.

е) Це база даних реального часу.

2.2. Realtime Firebase DB

Олним із найпотужніших засобів Firebase є база даних реального часу. Дані в режимі реального часу – це шлях у майбутнє. Ніщо з цим не порівняється. Більшість баз даних вимагають здійснення постійних HTTP-викликів, щоб отримати та синхронізувати нові дані. Більшість баз даних надають дані лише тоді, коли додаток просить їх і не слідкують за тим коли дані обновлюються.

Коли розробник підключає свій додаток до Firebase, він має можливість не підключатися через звичайний HTTP. Він може підключитися через WebSocket, і йому не треба буде кожен раз створювати нові запити для отримання даних, що в результаті буде швидше, ніж HTTP. Застосунку не потрібно здійснювати окремі виклики WebSocket, тому що одного підключення достатньо. Усі дані автоматично синхронізуються через один WebSocket так швидко, наскільки це може передати мережа клієнта. Firebase надсилає вам нові дані, щойно вони оновлюються. Коли клієнт зберігає змінені дані, усі підключені клієнти отримують оновлення майже миттєво. Але при цьому всьому Firebase залишає можливість робити HTTP запити, коли розробник вважає їх необхідними.

WebSocket [14] — це стандартний протокол для двосторонньої передачі даних між клієнтом і сервером. Протокол Web Socket побудований на основі TCP. Веб-сокети в основному використовуються для надсилання повідомлень клієнту в режимі реального часу. Це протокол із збереженням стану, що означає, що з'єднання між клієнтом і сервером буде існувати, поки не завершиться будь-якою стороною (клієнтом або сервером).

HTTP — це протокол зв'язку всесвітньої павутини. Http працює як протокол запит-відповідь у обчислювальній моделі клієнт-сервер. Це односпрямований протокол, за яким клієнт надсилає запит, а сервер —

відповідь. Кожен запит асоціюється з відповідною відповіддю, а після надсилання відповіді з'єднання закривається.

Переваги WebSocket над HTTP:

- використовує менше ресурсів і швидше – для 5000 запитів у 5-7 разів швидше ніж звичайний HTTP (Рисунок 2.1);
- потокова передача запитів і відповідей;
- вищий показник ефективності;
- усунення проблем із затримкою;
- підтримує дуплексне підключення;
- міжплатформна сумісність.

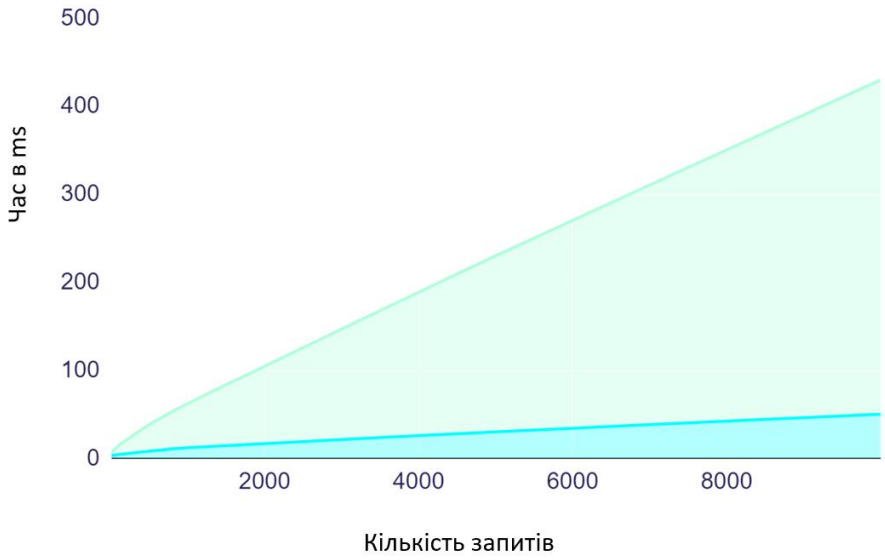


Рисунок 2.1 – Графік залежності часу від кількості запитів для WebSocket і HTTP

2.3. Особливості впровадження і використання Firebase DB у веб-застосунку

Для того щоб почати роботу з Firebase, необхідно виконати наступні кроки:

- Створити акаунт в Firebase;
- В новому акаунті, за допомогою інтерфейсу користувача, створити Firebase проект;

– В проекті веб-застосунку слід встановити Firebase NPM, для цього можна використати команду «npm install firebase –save»;

Для того щоб безперешкодно користуватися Firebase DB треба розуміти її певні особливості. Дані Firebase представляють об’єкти JSON. Якщо користувач відкриває програму з інформаційної панелі Firebase, він може додати дані вручну, натиснувши знак «+» (Рисунок 2.2). Також інтерфейс користувача, надає можливість видалити обраний об’єкт. Для цього необхідно натиснути на «кошик».

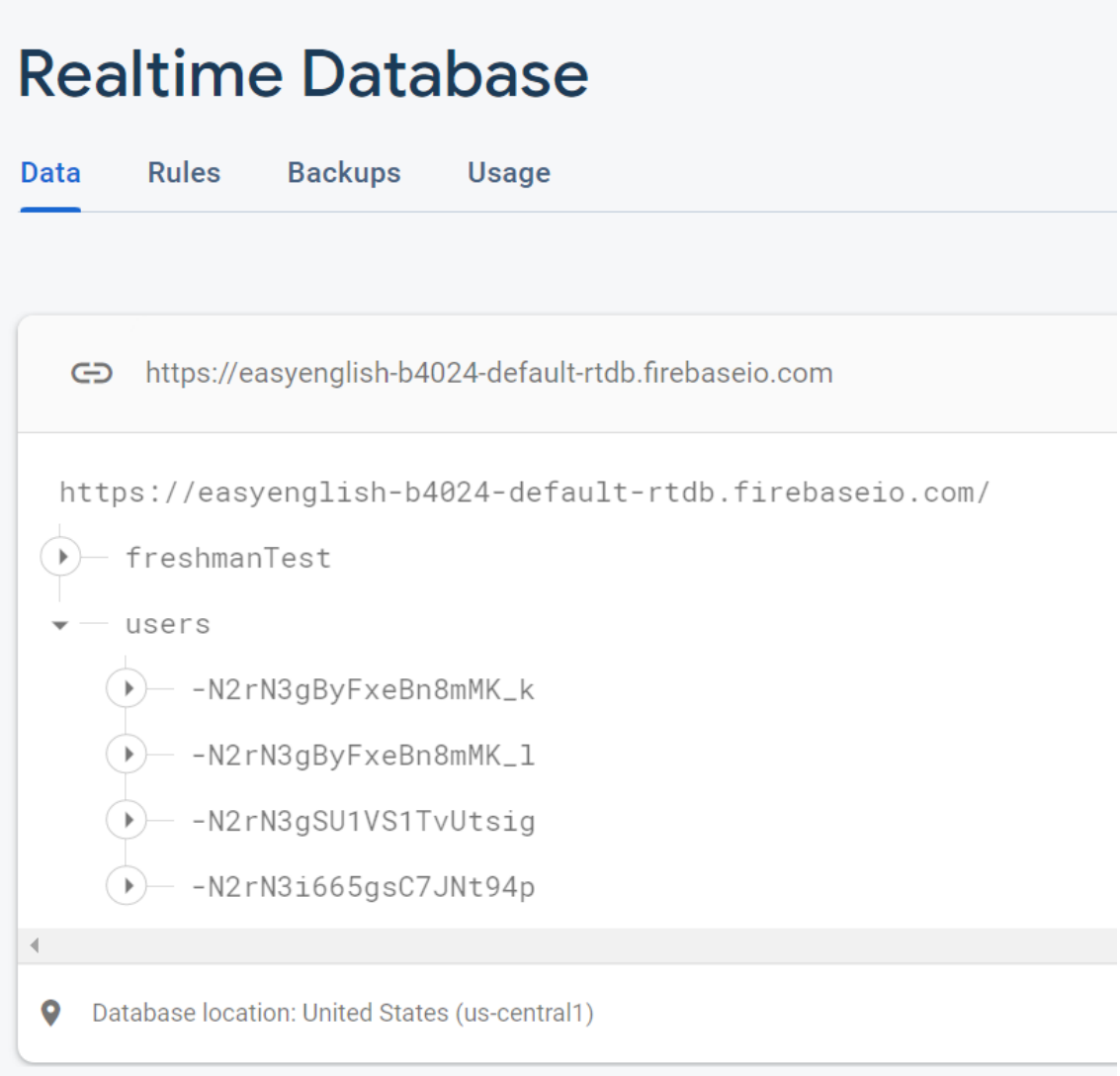


Рисунок 2.2 — Приклад даних в Firebase DB

Також з огляду на те що, це база даних реального часу, Firebase DB має ряд особливостей, які важливо розуміти перед початком роботи з нею. Наприклад,

для того, щоб створити таку структуру даних (Рисунок 2.3), необхідно надіслати дерево JSON — ['Alica', 'Bob'] до колекції USERS.

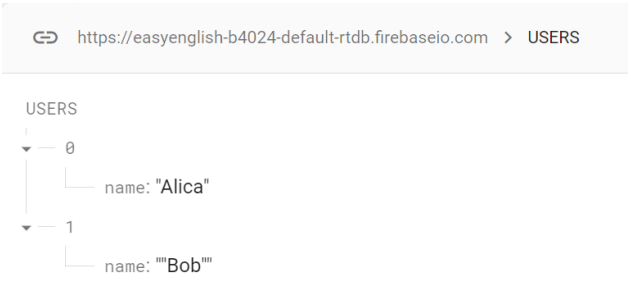


Рисунок 2.3 — Дерево JSON для масиву ['Alica', 'Bob']

Це тому, що Firebase не підтримує масиви напряму, але створює список об’єктів із цілими числами як іменами ключів.

Причина невикористання масивів полягає в тому, що Firebase діє як база даних у режимі реального часу, і якщо кілька користувачів будуть маніпулювати масивами одночасно, результат може бути непередбачуваним, оскільки індекси масивів постійно змінюються.

Те, яким чином Firebase вирішує цю проблему, дозволяє ключам (індексам) завжди залишаться незмінними. Користувачі могли б видалити Алісу, а Боб все ще мав би ключ (індекс) 1.

Іншим важливим аспектом є те що, такий підхід має ряд відмінностей у використанні, у порівнянні з загальновідомим HTTP. Крім способу яким Firebase додається у проект, він відрізняється власними типами, шляхом яким він отримує та обробляє дані. На прикладі отримання списку користувачів додатку буде проілюстровано яким саме чином відбуваються вищезгадані процеси при підключенні до Firebase DB. Всі методи які створені для взаємодії з базою даних знаходяться у main.service.ts. У конструктор сервісу Angular необхідно підключити AngularFireDatabase клас (Рисунок 2.4 а)), який допомагає взаємодіяти з Firebase. Перед цим слід проініціалізувати змінну usersRef

(Рисунок 2.4 б)), яка буде мати тип з простору Firebase. Перед завантаженням сторінок проекту цій змінній надається значення відповідного референсу у БД.

```
constructor(
  private db: AngularFireDatabase) {
    this.usersRef = db.list('/users');
}
```

а

```
public usersRef: AngularFireList<proUser[]>;
```

б

Рисунок 2.4 – Ініціалізація змінної usersRef: а – запровадження у сервіс AngularFireDatabase та надання значення usersRef; б – ініціалізація usersRef

Наступним етапом необхідно створити методи для отримання значень цих змінних (Рисунок 2.5).

```
getUsersListRef(): AngularFireList<proUser[]> {
  return this.usersRef;
}
```

Рисунок 2.5 – Метод для отримання значення usersRef

Для отримання поточного значення масиву користувачів використовується метод getUsersList() (Рисунок 2.6). Метод з рисунка 2.5 отримує результат у форматі AngularFireList, що є потоком даних і не може бути використаний як звичайний масив. Для того щоб отримати дані у звичному форматі необхідно після виклику цього методу, викликати snapshotChanges() який зробить зліпок з поточного значення списку користувачів. Наступним кроком слід обробити отриманий результат за допомогою операторів RxJS – pipe і map. В кінці буде отримано масив користувачів у звичному форматі, який можна використовувати для відображення на сторінці.

```
getUsersList() {
  return this.getUsersListRef().snapshotChanges().pipe(
    map(changes =>
      changes.map(c => ({key: c.payload.key, ...c.payload.val()})
      )
    )
  )
}
```

Рисунок 2.6 – Метод для отримання поточного значення масиву користувачів

Отже, на прикладі списку користувачів веб-додатку «EasyEnglish» було проілюстровано яким чином можна взаємодіяти з Realtime Firebase DB.

РОЗДІЛ 3

МОЖЛИВОСТІ ПРОГРАМНОГО ПРОДУКТУ

3.1. Концепція веб-застосунку

Обрана назва застосунку – EasyEnglish – українською перекладається як «Проста англійська». Система «EasyEnglish» призначена для вивчення англійської мови за індивідуальним планом викладача. Програма придатна для студентів які вже мають знання мови, але хочуть покращити свій рівень. За допомогою індивідуального плану студент може зрозуміти теми, які він з тих чи інших причин не знає або забув, та відпрацювати їх. Застосунок може спростити студентам та викладачам пошук один одного. Також система створена для того, щоб полегшити та налагодити комунікацію сторін.

Після авторизації у застосунку, студент проходить тест на рівень знання англійської та потряпляє у список “вільних” студентів. Вчитель переглядає цей список та обирає учнів яких планує вести. У кожного учня допустимий лише один вчитель, який буде призначати завдання.

Програма має експлуатуватися користувачами у разі бажання чи необхідності покращити знання з англійської мови та закрити недоліки в уже вивчених темах, що досить складно зробити під час групових занять. Використання програми полегшує комунікацію між студентами і викладачами та допомагає підібрати ідеальну пару «викладач — студент». Оскільки викладач зосереджений над проблемами у знаннях кожного студента окремо та визначає завдання які допоможуть саме йому, учень швидше засвоює матеріал та отримує знання яких йому бракувало.

Отже, для студента система «EasyEnglish» надає можливість вивчення англійської мови по індивідуальному плану викладача, а викладачу надає можливість зручно вести індивідуальний план для декількох студентів одразу, та відслідковувати їх прогрес.

3.2. Обґрунтування доцільності створення системи «EasyEnglish»

Зі збільшенням попиту на онлайн платформи, швидко росте кількість додатків, які пропонують свої послуги у сфері освіти. Курси по програмуванню, кулінарії, саморозвитку – все це маленькими кроками переходить у світ онлайн. Проте одним із найбільш популярних напрямів у освіті за допомогою комп'ютера стає саме вивчення іноземних мов, зокрема англійської. Тому що це загальноприйнята міжнародна мова, знання якої надає можливість розуміти більшість інших курсів доступних у мережі.

Саме тому існує незлічена кількість онлайн платформ по вивченню англійської мови, які створювали компанії з усього світу. На перший погляд може здатися, що ця ніша вже зайнята і неможливо розробити щось нове, з іншим підходом, проте це не так. Перед створенням системи «EasyEnglish» було проаналізовано аналогічні додатки та розроблено власну концепцію застосунку. Отже, переваги додатку «EasyEnglish» порівняно з іншими подібними:

- індивідуальний підхід до кожного студента (в Duolingo.com використовуються наперед визначені алгоритми, відсутні вчителі та зворотній зв'язок, до яких можна звернутися в разі необхідності, що буває досить часто при вивченні іноземної мови);
- наявні вчителі, які складають завдання, виходячи зі знань студента та допомагають доопрацювати «втрачені» теми, через які студент не може зрозуміти більш складні конструкції (в Lingualeo.com наявні гарні матеріали для опрацювання, проте досить часто студенту самостійно важко зрозуміти де у нього недоліки у знаннях);
- в системі є можливість не починати вивчення іноземної мови з 0 (завдяки вхідному тесту), а продовжувати з того рівня на якому зараз знаходиться студент (Duolingo.com, наприклад, більше підходить для починаючих);
- у вчителів є можливість створювати структуровані завдання для своїх учнів (на Englishspeak.com добре пропрацьовані ситуації, проте відсутня можливість комплексно покращувати граматику);

– привабливий та зрозумілий інтерфейс (не секрет, що сьогодні однією із рушійних сил для освоєння нової мови може стати гарне візуальне відображення та впровадження «ігрового» аспекту у процес навчання. На Esl.fis.edu та learn-english-today.com представлені чудові тести по різним темам та структурована граматика, проте відсутня хороша візуалізація та зручний інтерфейс);
– викладачі надають правила перед виконанням завдань (Englishlearner.com, Englishteststore.net — ресурс з тестами по граматиці, проте відсутні достатні граматичні правила).

Отже, «EasyEnglish» — платформа з індивідуальним підходом для вивчення англійської мови. Вона враховує більшість кращих аспектів вище зазначених сайтів, що зумовлює необхідність у створенні подібної системи.

3.3. Опис і документування концепції та можливостей веб-застосунку «EasyEnglish»

Для роботи з веб застосунком потрібно перейти в браузері на його відповідну адресу. У користувача який потрапив на сайт, є можливість авторизуватися за 2 ролями – студент та викладач. Це і є актори[1] системи (Таблиця 3.1).

Таблиця 3.1 – Короткий опис акторів системи

Актор	Короткий опис
Студент	Рядовий користувач системи. Має можливість пройти тест на визначення рівня знань, переглянути список доступних завдань, виконати доступні завдання.
Викладач	Має можливість створювати завдання, набирати студентів, переглядати списки студентів та призначати їм завдання.

[1] Актор — це роль, що виконується сутностями (людина, інша система тощо), які взаємодіють з системою.

Актори системи мають ряд можливостей, які надає їм застосунок. У обох ролей є спільні наміри, проте більшість функціоналу у них різна і специфікована саме для їх ролі у застосунку.

Список намірів для студента:

- зареєструватись у системі;
- увійти в систему;
- пройти початковий тест;
- переглянути список завдань;
- виконати завдання.

Список намірів для викладача:

- зареєструватись у системі;
- увійти в систему;
- створити завдання;
- переглянути список вільних студентів;
- додати студента на навчання;
- переглянути список власних студентів, які проходять навчання;
- додати завдання студенту.

Наміри обох акторів можна виконати на сторінках додатку. Список сторінок Веб-застосунку:

a) Login – сторінка авторизації для входу в систему (Рисунок 3.1). Наявна валідація полів: поле пошти обов’язкове та потребує введення у форматі пошти, поле пароля обов’язкове.

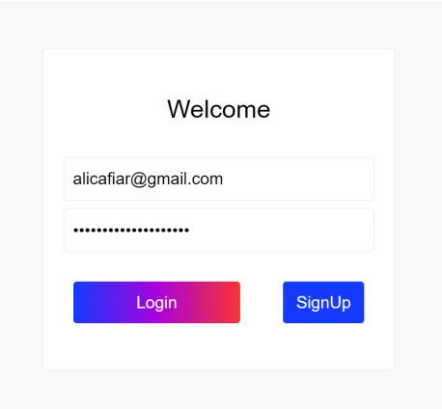


Рисунок 3.1 – Сторінка логіну з введеними даними

б) Register – сторінка реєстрації (Рисунок 3.2). Валідація найвна на усіх полях у формі.

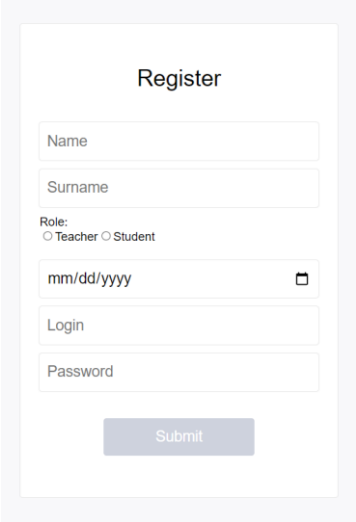


Рисунок 3.2 – Сторінка реєстрації

в) FreshmanTest – сторінка, на якій присутній тест, для визначення початкового рівня володіння мовою (Рисунок 3.3). Якщо користувач не обирає відповідь на питання у тесті і натискає кнопку «Save answers» то питання зараховується неправильним.

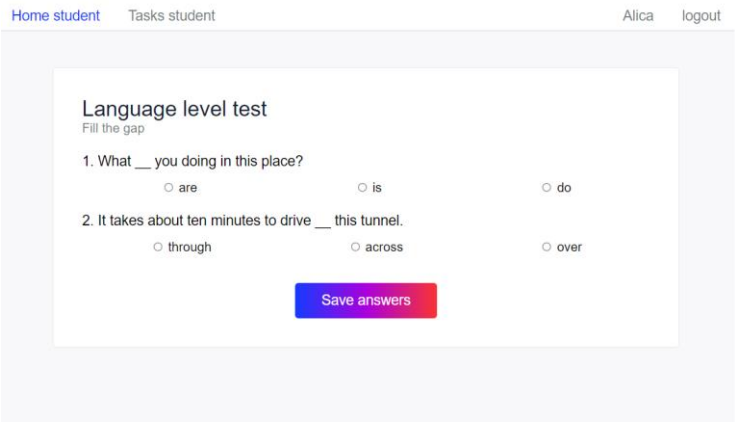
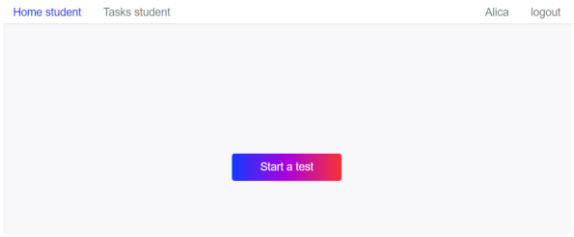
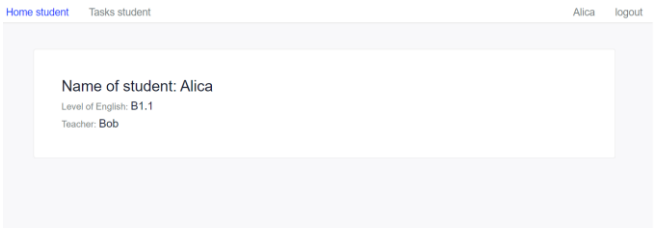


Рисунок 3.3 – Сторінка початкового тесту на рівень мови

г) HomeStudent – домашня сторінка студента. Якщо студент ще не має рівня англійської мови, то він бачить сторінку, яка пропонує йому пройти тест на рівень мови (Рисунок 3.4 а)). Коли користувач вже має рівень англійської, то він бачить інформаційне вікно (Рисунок 3.4 б)). У цьому вікні відображається ім’я студента, його рівень мови та, якщо викладач уже обрав студента, ім’я викладача.



а

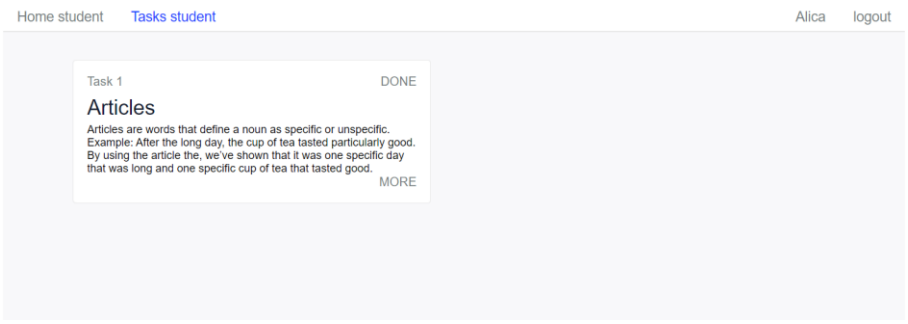


б

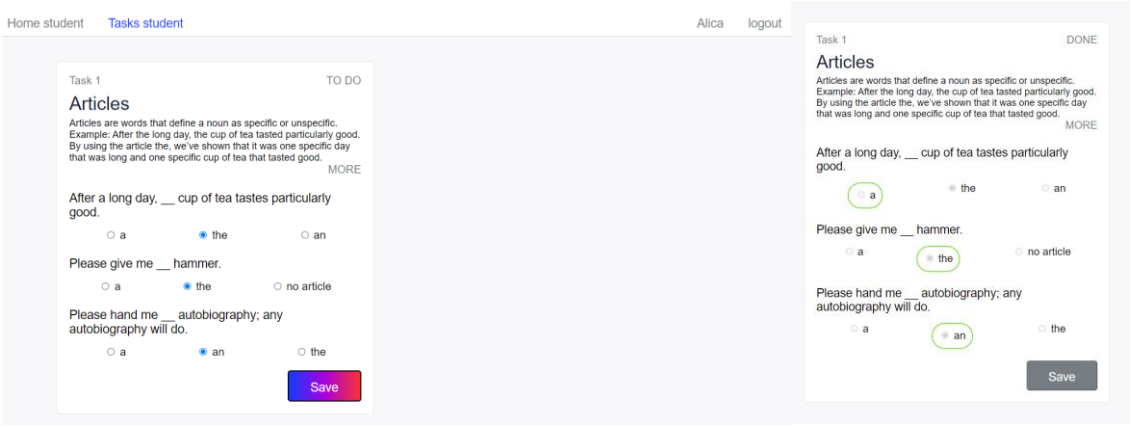
Рисунок 3.4 - Домашня сторінка студента: а – вигляд сторінки, якщо студент ще не пройшов тест; б – інформаційне вікно після проходження тесту

д) TasksStudent – сторінка на якій відображаються доступні для студента завдання (Рисунок 3.5 а)), є можливість виконувати їх

(Рисунок 3.5 б)) та переглядати правильні результати (Рисунок 3.5 в)).



а



б в

Рисунок 3.5 – Сторінка «TasksStudent»: а – вигляд сторінки; б – відкрите завдання; в – пройдене завдання з правильними відповідями

е) HomeTeacher – домашня сторінка викладача.

ж) AvaliableStudents – список вільних студентів, які вже пройшли тест на рівень мови та ще не мають викладача (Рисунок 3.6).

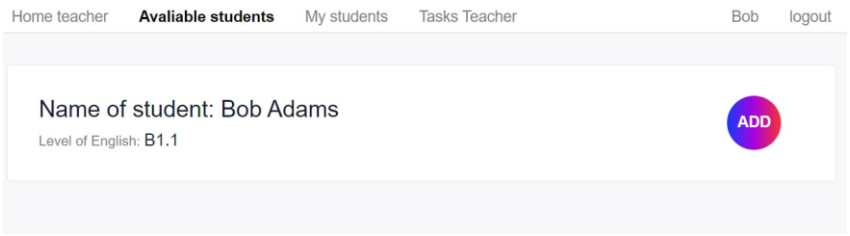


Рисунок 3.6 – Сторінка доступних студентів

з) MyStudents – сторінка зі списком студентів викладача, на ній користувач може переглядати вже призначені студенту завдання та призначити нові.

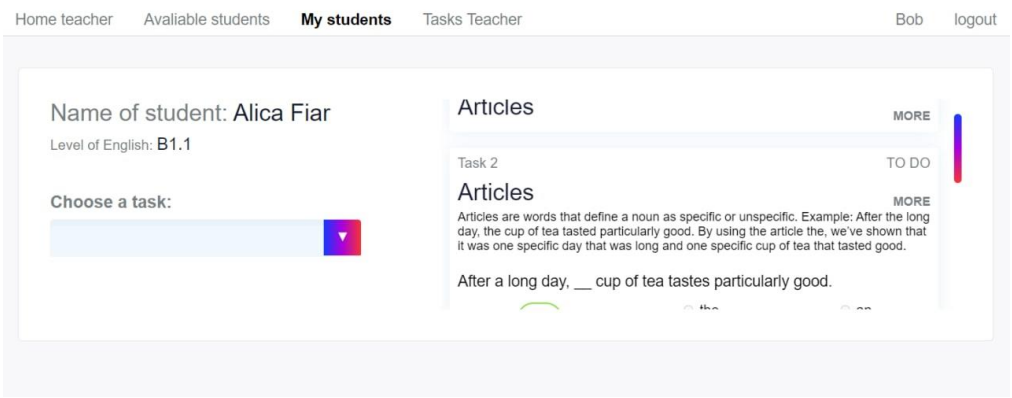
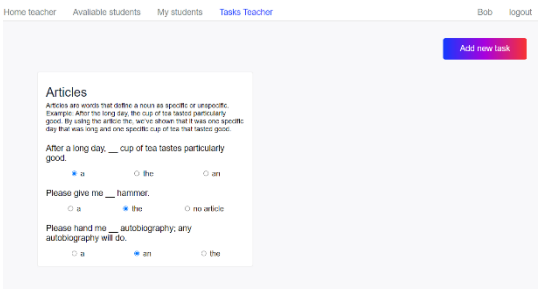
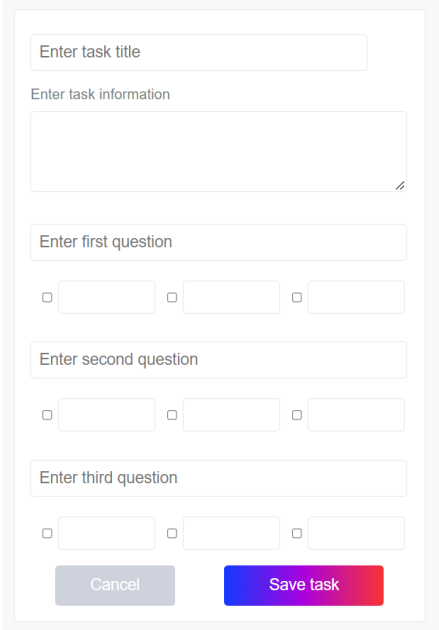


Рисунок 3.7 – Сторінка студентів викладача

и) TasksTeacher - сторінка, на якій доступний список завдань викладача і є конструктор для створення нових завдань.





а б

Рисунок 3.8 – Сторінка завдань вчителя: а – сторінка з уже створеним завданням; б – конструктор завдань

Для сегментації та визначення у якому порядку і які сторінки буде бачити користувач можна скористатись user flow діаграмою (Рисунок 3). На ній зображені сторінки та основний функціонал доступний на них.

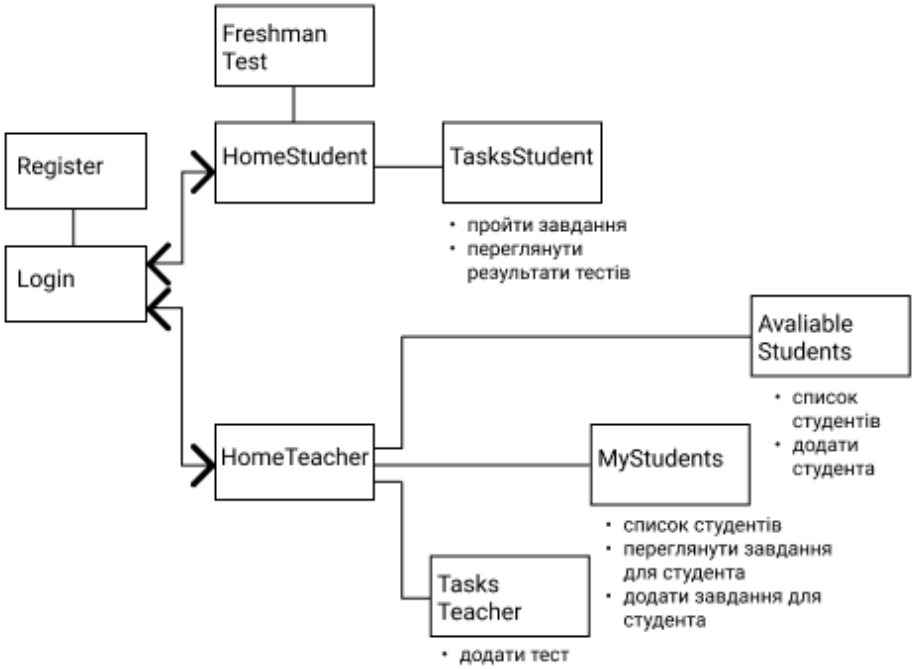


Рисунок 3.9 – Подання User flow у форматі task flow для всіх акторів Веб-застосунку

Але цього може бути недостатньо для розуміння які саме події призведуть до виконання того чи іншого сценарію. Для успішного виконання програми необхідно розуміти послідовність дій, які приведуть до виконання кожного вищезазначеного наміру для кожного актора. Для виконання спільних намірів акторів необхідно слідувати наступним алгоритмам:

а) реєстрація у системі:
   1) на головній сторінці “login” натиснути кнопку “signUp”;
   2) ввести відповідні дані в кожне поле (Ім’я, Фамілія, Роль, Дата народження, Логін, Пароль);
   3) натиснути кнопку “SignUp”;

б) вхід у систему:
   1) зареєструватися в систему, якщо користувача ще немає у системі;

2) на головній сторінці “login” ввести логін та пароль;
3) натиснути кнопку “Login”;

Для виконання намірів доступних студенту:

в) пройти початковий тест:
    1) пройти реєстрацію як студент, якщо користувача ще немає у системі;
    2) авторизуватися;
    3) на домашній сторінці студента натиснути “Start a test”;
    4) відповісти на кожне питання початкового тесту;
    5) натиснути кнопку “Submit”;
    6) на домашній сторінці студента переглянути результат тесту - рівень англійської;
г) переглянути список завдань:
    1) пройти реєстрацію як студент, якщо користувача ще немає у системі;
    2) авторизуватися;
    3) пройти початковий тест;
    4) почекати та отримати свого вчителя;
    5) перейти на вкладку “TasksStudent”;
    6) переглянути список доступних завдань;
д) виконати завдання:
    1) пройти реєстрацію як студент, якщо користувача ще немає у системі;
    2) авторизуватися;
    3) пройти початковий тест;
    4) почекати та отримати свого вчителя;
    5) перейти на вкладку “TasksStudent”;
    6) переглянути список доступних завдань;

7) обрати завдання, яке учень бажає освоїти та пройти;
8) натиснути на кнопку “MORE” для відкриття повного завдання;
9) прочитати теоритичну відомість та позначити правильну відповідь на кожне питання;
10) натиснути на кнопку “Save”;
11) переглянути правильні відповіді до тесту;

Для виконання намірів доступних викладачу:

е) створити завдання:
1) пройти реєстрацію як викладач, якщо користувача ще немає у системі;
2) авторизуватися;
3) перейти на вкладку “Tasks Teacher”;
4) натиснути на кнопку “Add new task”;
5) заповнити поля нового завдання (заголовок тесту, теоритична відомість, питання з варіантами відповідей);
6) натиснути кнопку “Save task”;

ж) переглянути список вільних студентів:
1) пройти реєстрацію як викладач, якщо користувача ще немає у системі;
2) авторизуватися;
3) перейти на вкладку “AvaliableStudents”;
4) переглянути список доступних студентів (які вже пройшли тест та ще не мають викладача);

з) додати студента на навчання:
1) пройти реєстрацію як викладач, якщо користувача ще немає у системі;
2) авторизуватися;
3) перейти на вкладку “AvaliableStudents”;

4) переглянути список доступних студентів (які вже пройшли тест та ще не мають викладача);
5) натиснути кнопку “Add student”;

и) переглянути список власних студентів, які проходять навчання:
1) пройти реєстрацію як викладач, якщо користувача ще немає у системі;
2) авторизуватися;
3) якщо студентів ще немає, додати їх;
4) перейти на вкладку “MyStudents”;
5) переглянути список студентів (яких веде цей викладач);

к) додати завдання студенту:
1) пройти реєстрацію як викладач, якщо користувача ще немає у системі;
2) авторизуватися;
3) якщо студентів ще немає, додати їх;
4) перейти на вкладку “MyStudents”;
5) переглянути список студентів (яких веде цей викладач);
6) вибрати студента;
7) вибрати завдання студенту із переліку запропонованих.

Для того щоб, детальніше описати об’єкти взаємодії, завдання та діалоги нижчого рівня в інтерфейсах користувача можна використати методи моделювання користувацьких інтерфейсів. Використання моделей як частини розробки користувальницького інтерфейсу може допомогти врахувати вимоги користувача, чітко визначити взаємозв'язки між різними частинами інтерфейсу та їх ролями. Тому для системи «EasyEnglish» було розроблено модель вмісту інтерфейсу користувача (Таблиця 3.2).

Таблиця 3.2 – Модель вмісту інтерфейсу користувача для системи «EasyEnglish»

Ім’я сторінки/намір	Варіант використання		Елемент даних для виконання варіанта	Операція з даними
	Код	Найменування		
Register	ST1	Зареєструватись у системі	Ім’я Фамілія Роль Дата народження Логін Пароль	Введення
Login	ST2	Увійти в систему	Логін Пароль	Введення
FreshmanTest	S1	Пройти початковий тест	Базові знання англійської мови	Обирання checkboxes
TasksStudent	S2	Вирішити завдання	Базові знання англійської мови	Обирання checkboxes
TasksStudent	S3	Переглянути результати завдань	Список результатів	Перегляд
TasksTeacher	T1	Створити завдання	Завдання для студента	Введення
AvaliableStudents	T2	Переглянути список студентів	Список студентів	Перегляд

Кінець таблиці 3.2

Ім’я сторінки/намір	Варіант використання		Елемент даних для виконання варіанта	Операція з даними
	Код	Найменування		
AvaliableStudents	T3	Додати студента в свою групу	Картка студента	Натиснути «Add»
MyStudents	T4	Переглянути список студентів	Список студентів	Перегляд
MyStudents	T5	Переглянути список доступних студенту завдань	Список завдань	Перегляд
MyStudents	T6	Додати завдання студенту	Список тестів	Вибрати завдання з dropdown

3.4. Опис сценаріїв застосунку з використанням мереж Петрі

Для спрощення реалізації веб-застосунку кращого розуміння його наповненості та доступного функціоналу, може бути використаний ще один інструмент – мережі Петрі.

Мережа Петрі – це графічний і математичний засіб, який допомагає у моделюванні систем та процесів. Це орієнтований граф, який має чотири базових елементи: вузо або позиція, переходи, дуги і маркери (Рисунок 3.10).

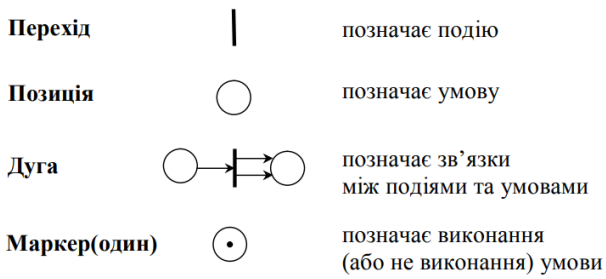


Рисунок 3.10 – Графічне представлення елементів мережі Петрі

Вузли визначають стан, в якому знаходиться або може знаходитись мережа або її частина. Переходи – це такі елементи мережі, які позначають дії, які виконуються під час спрацювання переходів. Для того щоб перехід активізувався, необхідне виконання деяких умов, які будуть визначатися наявністю маркерів у вузлах мережі, що з'єднані з переходом.

Для системи «EasyEnglish» було розроблено мережу Петрі (Рисунок 3.11). Початкова подія тут – клієнт зайшов на сторінку сайту. Якщо маркер з'явиться у умові «натискання кнопки «register»», то відбудеться подія – «клієнт на сторінці реєстрації». Наступна подія після цієї буде – «клієнт зареєструвався», виконання якої поверне користувача назад на сторінку авторизації, яку першою побачив користувач.

Коли у умові «правильно введені дані існуючого клієнта» з'явиться маркер, подія «клієнт авторизувався» буде виконана. Далі, в залежності від наявності маркера в лівій чи правій гілці мережі Петрі клієнт потрапить на сторінку викладача чи студента. Аналогічно описуються і інші події у моделі мережі Петрі.

Коли клієнт потрапляє на сторінку викладача, є декілька позицій в які може потрапити маркер і кожна з них приведе до виконання однієї з 3 подій: клієнт на сторінці вільних студентів, клієнт на сторінці власних студентів, клієнт на сторінці тестів. У кожної з цих подій є по 2 додаткових позиції, що приведуть їх на сусідню сторінку.

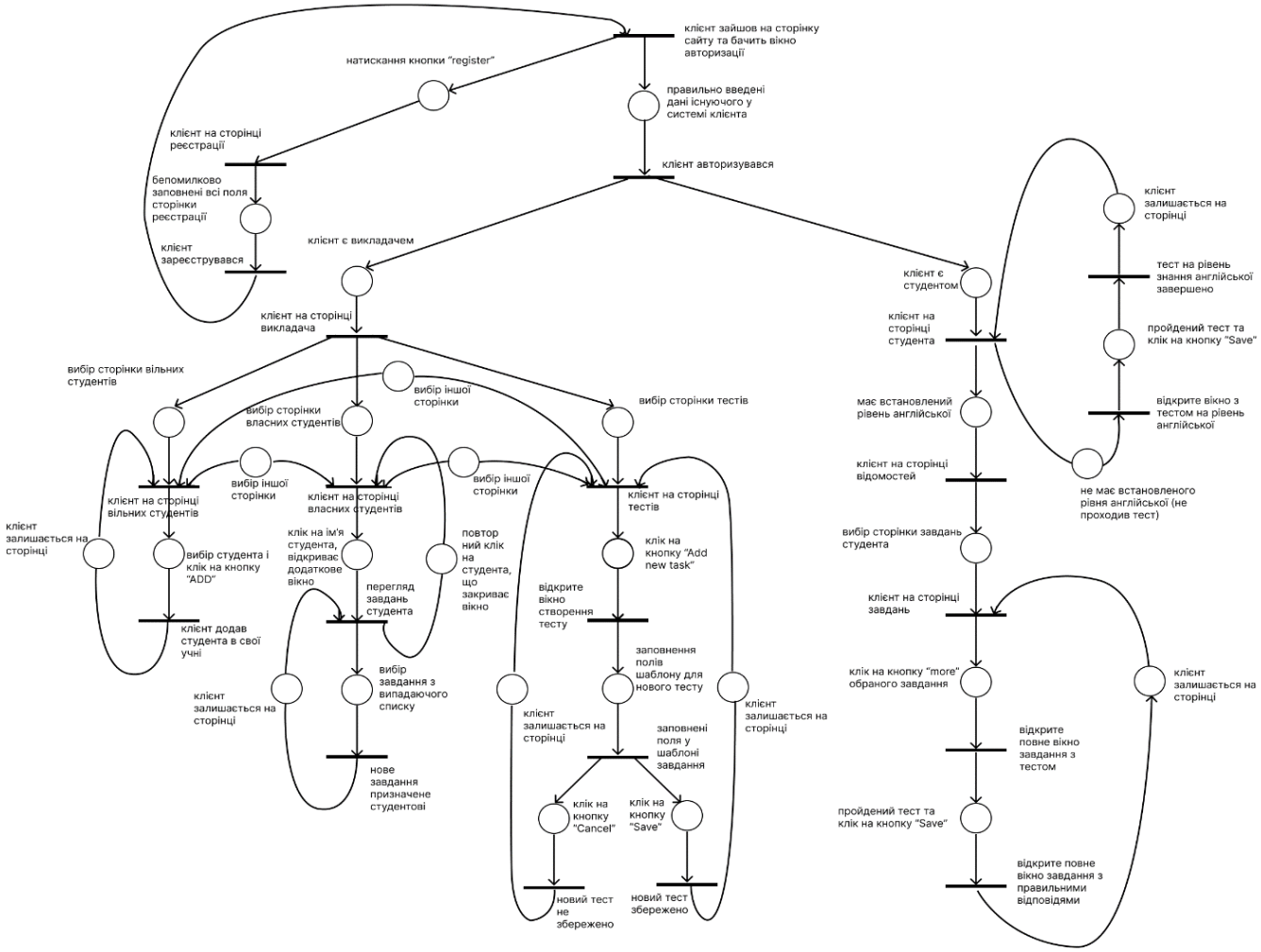


Рисунок 3.11 – Мережа Петрі для подій веб-застосунку «EasyEnglish»

Якщо клієнт знаходиться на сторінці вільних студентів, йому доступна позиція «вибір студента і клік на кнопку «Add»», яка при наявності у ній маркера приведе його до пореходу «клієнт додав студента у свої учні». Після цього клієнт циклічно потрапляє до переходу «клієнт на сторінці вільних клієнтів».

На переході «клієнт на сторінці власних студентів», якщо маркер у умові «клік на ім'я студента», то відбувається перехід «перегляд завдань студента». Далі в залежності від виконаної умови доступні 2 переходи: «нове завдання призначене студентові» та повернення до переходу «клієнт на сторінці власних студентів». Після виконання 1-го переходу клієнт може повернутися на перехід «перегляд завдань студента».

У останній гілці, доступній викладачу, після переходу «клієнт на сторінці тестів» доступна позиція «клік на кнопку «Add new task»», маркер в якій призведе до переходу «відкрите вікно створення тесту». Наступний перехід «заповнені поля у шаблоні завдання» має 2 доступні клієнту позиції: «клік на кнопку «Cancel»» та «клік на кнопку «Save»». Кожна з яких приведе до збереження чи не збереження нового тесту.

Потрапляючи на сторінку студента, клієнт має можливість виконати 2 переходи, в залежності від наявності маркеру у позиціях «клієнт має встановлений рівень англійської» та «не має рівня англійської». Якщо маркер знаходиться у 2-ій позиції, то наступний перехід доступний клієнтові це – «відкрите вікно з тестом на рівень мови», з якого відкривається позиція «пройдений тест та клік на кнопку «Save»». Результат наявності маркеру у ній – доступний перехід «тест на рівень англійської мови завершено», з якого можна повернутися до переходу «клієнт на сторінці студента».

При умові ж що маркер поміщений у позицію «має встановлений рівень англійської», клієнт потрапляє на сторінку відомостей про студента, з якої можна потрапити до сторінки завдань. Наступний доступний перехід це – «відкрите повне вікно з завданням тесту», з якого доступна позиція – «пройдений тест та клік на кнопку «Save»». Активізація якої призведе до переходу «відкрите повне вікно завдання з правильними відповідями».

Таким чином було описано сценарції застосунку «EasyEnglish» за допомогою мереж Петрі.

ВИСНОВКИ

Отже, у кваліфікаційній роботі було підібрано та проаналізовано сучасні Frontend технології, на основі яких було розроблено веб-застосунок. Серед основних технологій – HTML, SASS, TypeScript, фреймворк Angular, бібліотеки RxJS та Angular Material. Також було проаналізовано можливості та переваги, які надає платформа Google Firebase. В результаті огляду, у проекті було використано один із найпотужніших засобів Firebase – базу даних реального часу, Realtime Firebase DB.

У останньому розділі було проаналізовано та враховано переваги відомих аналогічних додатків для вивчення англійської мови. На основі отриманих даних, створено концепцію застосунку, описано сторінки, наміри, їх алгоритми для усіх акторів додатку та розроблено модель вмісту інтерфейсу користувача. У результаті створених моделей, вдалося безперешкодно реалізувати веб-застосунок «EasyEnglish». Також, для спрощення процесу розробки та кращого розуміння сценаріїв застосунку, їх було описано за допомогою мереж Петрі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. І.В. Стеценко Моделювання систем / І.В. Стеценко – К.: «М-во освіти і науки України, Черкас. держ. технол. ун-т. – Черкаси», 2010. – 399 с.
2. Sass documentation [Електронний ресурс]. – Режим доступу: URL: – https://sass-lang.com/guide.
3. Javascript vs Typescript [Електронний ресурс]. – Режим доступу: URL: – https://www.guru99.com/typescript-vs-javascript.html.
4. Angular documentation [Електронний ресурс]. – Режим доступу: URL: – https://angular.io/guide/what-is-angular.
5. What is Angular?: Architecture, Features, and Advantages [Електронний ресурс]. – Режим доступу: URL: – https://www.simplilearn.com/tutorials/angular-tutorial/what-is-angular.
6. Six reasons to choose angular for web development [Електронний ресурс]. – Режим доступу: URL: – https://softwebsolutions.com/resources/why-angularjs-should-be-used-for-development.html#:~:text=Unlike%20some%20of%20the%20other,updated%20automatically%20in%20real-time.
7. Angular архітектура [Електронний ресурс]. – Режим доступу: URL: – https://angular.io/guide/architecture.
8. RxJS documentation [Електронний ресурс]. – Режим доступу: URL: – https://rxjs.dev/guide/overview.
9. RxJS – Overview [Електронний ресурс]. – Режим доступу: URL: – https://www.tutorialspoint.com/rxjs/rxjs_overview.htm#:~:text=The%20full%20form%20of%20RxJS,other%20Javascript%20libraries%20and%20frameworks.
10. Angular material [Електронний ресурс]. – Режим доступу: URL: – https://material.angularjs.org/latest/.
11. Firebase Wikipedia [Електронний ресурс]. – Режим доступу: URL: – https://en.wikipedia.org/wiki/Firebase.

12. What is Firebase? All the secrets [Електронний ресурс]. – Режим доступу: URL: – https://blog.back4app.com/firebase/
13. What is Google Firebase [Електронний ресурс]. – Режим доступу: URL: – https://delvedeeper.com/articles/what-is-google-firebase/
14. Web Socket vs HTTP [Електронний ресурс]. – Режим доступу: URL: – https://ipwithease.com/web-socket-vs-http/.