**CS330 Software Engineering**

**Software Requirements Specification (SRS) Template**

Items that are intended to stay in as part of your document are in **bold**; explanatory comments are in *italic* text. Plain text is used where you might insert wording about your project.

The document in this file is an annotated outline for specifying software requirements, adapted from the IEEE Guide to Software Requirements Specifications (Std 830-1993).

Tailor this to your needs, removing explanatory comments as you go along. Where you decide to omit a section, keep the header, but insert a comment saying why you omit the data.

Software system for teaching foreign words

SmartCore Developers

Miach Kateryna

Viktoriia Serdiuk

**Software Requirements Specification**

**Document**

**Version: 1.0** **Date: 17.06.2025**

**Зміст**

**1. ВСТУП**

1.1 Огляд продукту

1.2 Мета

1.3 Межі

1.4 Посилання

1.5 Означення та абревіатури

**2. ЗАГАЛЬНИЙ ОПИС**

2.1 Перспективи продукту

2.2 Функції продукту

2.3 Характеристики користувачів

2.4 Загальні обмеження

2.5 Припущення й залежності

**3. КОНКРЕТНІ ВИМОГИ**

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

3.1.2 Апаратний інтерфейс

3.1.3 Програмний інтерфейс

3.1.4 Комунікаційний протокол

3.1.5 Обмеження пам’яті

3.1.6 Операції

3.1.7 Функції продукту

3.1.8 Припущення й залежності

3.2 Властивості програмного продукту

3.3 Атрибути програмного продукту

3.3.1 Надійність

3.3.2 Доступність

3.3.3 Безпека

3.3.4 Супроводжуваність

3.3.5 Переносимість

3.3.6 Продуктивність

3.4 Вимоги бази даних

3.5 Інші вимоги

1 ВСТУП

1.1 Огляд продукту

У межах цього проєкту буде створено інтерактивний веб-застосунок для вивчення англійської мови. Застосунок працюватиме через веб-браузер і буде орієнтований на користувачів, які бажають покращити свій словниковий запас, граматику та навички спілкування. Однією з ключових особливостей системи стане впровадження чат-бота, що базуватиметься на технологіях штучного інтелекту, який зможе вести діалоги з користувачем, ставити запитання, виправляти помилки та мотивувати до навчання. Застосунок матиме низку функціональних сторінок: реєстрацію, вхід, профіль, словник, сторінку міні-тестів, сторінки вправ: вибір правильної відповіді та побудова речення, а також взаємодію з чат-ботом. Передбачається, що користувач зможе проходити короткі тести для самоперевірки, переглядати вивчені слова, а також отримувати зворотний зв’язок від AI. Реалізація продукту планується з використанням такого стека технологій:

* Frontend: HTML, CSS, JavaScript, React;
* Backend: Node.js з Express.js;
* База даних: Sequelize ORM з SQLite.

1.2 Мета

Головна мета – розробити ефективний онлайн-інструмент для самостійного вивчення англійської мови з використанням інноваційних підходів до навчання. Продукт має забезпечити зручність, доступність та гнучкість у навчальному процесі, завдяки можливості навчатися у будь-який час і з будь-якого пристрою. Система буде спрямована на: адаптацію навчального матеріалу до рівня користувача; інтеграцію чат-бота, який допомагатиме у вивченні мови в режимі реального часу, генеруватиме відповіді, ставитиме питання та надаватиме індивідуальні рекомендації.

1.3 Межі

На першому етапі розробки застосунок матиме обмежений функціонал, орієнтований на основні потреби користувача, а саме:

* підтримка лише англійської мови;
* відсутність повноцінної адміністративної панелі для керування контентом;
* база даних SQLite, яка обмежена у продуктивності при високому навантаженні;
* історія проходження тестів не зберігатиметься;
* профіль користувача не міститиме повної статистики про активність;
* використання локального середовища для запуску.

1.4 Посилання

Для розробки проєкту можна скористатися такими офіційними джерелами:

React – бібліотека для створення інтерфейсів користувача (https://react.dev/)

Node.js – серверна платформа для JavaScript (https://nodejs.org/)

Express.js – фреймворк для створення веб-серверів (https://expressjs.com/)

Sequelize ORM – ORM для роботи з базами даних (https://sequelize.org/)

SQLite – легка вбудована реляційна база даних (https://sqlite.org/)

Google Gemini – платформа штучного інтелекту від Google (https://gemini.google.com/)

1.5 Означення та абревіатури

AI (Artificial Intelligence) – Штучний інтелект

JWT (JSON Web Token) – формат для безпечної передачі токенів авторизації

ORM (Object-Relational Mapping) – технологія для взаємодії з базами даних у вигляді об’єктів

SQLite Легка реляційна база даних, що не потребує окремого сервера

GitHub – Платформа для спільної розробки та зберігання коду

2 ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

Проєктований інтерактивний веб-додаток буде самостійним програмним продуктом, призначеним для кінцевих користувачів, що вивчають англійську мову. Він не є частиною більшої системи, проте його архітектура має передбачати можливість майбутньої інтеграції з іншими освітніми платформами або розширення функціонала.

Продукт буде позиціонуватися на ринку як комплексна екосистема для самостійного навчання, що охоплює всі етапи: від визначення рівня знань до щоденної практики. У майбутньому перспективи продукту включатимуть розробку повноцінних мобільних додатків, розширення бази мов для вивчення та інтеграцію з системами управління навчанням.

2.2 Функції продукту

Система буде надавати користувачам наступний набір основних функцій:

* головна сторінка: виконує роль інформаційного лендингу, що презентує основні можливості та переваги додатку;
* реєстрація користувача: користувачі можуть створювати новий обліковий запис, надаючи свої дані;
* автентифікація: зареєстровані користувачі можуть входити до системи за допомогою email та пароля;
* профіль користувача: доступ до особистого профілю з можливістю редагування даних і рівня складності;
* вибір вправ: сторінка з вибором трьох типів інтерактивних вправ – «вибір правильної відповіді», «побудова речення» та «переклад слів»;
* проходження адаптивних вправ: автоматичний підбір завдань відповідно до рівня користувача;
* інтерактивний ai-чат: можливість вести текстовий діалог з ai-асистентом на вільну або обрану тему;
* ведення персонального словника: можливість додавати нові слова до особистого словника під час виконання вправ.

2.3 Характеристики користувачів

Продукт орієнтований на широку аудиторію користувачів, які прагнуть вивчити або покращити англійську мову. Система розрахована на два основні типи користувачів: нового користувача, який може ознайомитися з головною сторінкою, та зареєстрованого користувача, якому доступний весь функціонал. Програмне забезпечення буде підтримувати учнів з різним рівнем підготовки, від початкового до просунутого, адаптуючи контент відповідно до вказаного ними рівня. Для взаємодії з системою від користувача очікується лише базовий рівень комп'ютерної грамотності, що включає вміння користуватися веб-браузером. Основною мотивацією цільової аудиторії є бажання вивчити мову "з нуля", покращити існуючі навички для кар'єрного росту або підготуватися до подорожей.

2.4 Загальні обмеження

При розробці та функціонуванні системи необхідно враховувати низку загальних обмежень. Перш за все, існують технологічні обмеження, оскільки продукт є веб-додатком і вимагатиме від користувача наявності пристрою з доступом до мережі Інтернет та сучасного веб-браузера. Важливим обмеженням є залежність від зовнішніх сервісів, адже функціонал AI чат-бота буде повністю залежати від стабільності та умов надання сервісу Google Gemini API. Також слід враховувати обмеження обраної бази даних: СУБД SQLite, хоча і є зручною для розробки, може мати обмеження продуктивності при високому навантаженні. Нарешті, на поточному етапі існують мовні обмеження, оскільки система призначена виключно для вивчення англійської мови україномовними користувачами.

2.5 Припущення й залежності

Проєктування системи базується на кількох ключових припущеннях. Припускається, що користувач матиме стабільне інтернет-з'єднання, необхідне для коректної роботи всіх мережевих функцій, особливо взаємодії з AI-асистентом. Система має низку залежностей, що впливають на її функціонування. Проєкт критично залежить від стабільності роботи зовнішнього сервісу Google Gemini API. Крім того, він залежить від сторонніх програмних бібліотек та пакетів (npm-пакетів), таких як React, Express та Sequelize, майбутні оновлення яких можуть вимагати адаптації кодової бази. Доступність додатку для кінцевих користувачів буде залежати від надійності та стабільності роботи обраного хостинг-провайдера.

3. КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

Інтерфейс користувача повинен бути розроблений з урахуванням принципів UX/UI, забезпечуючи легкість навігації, приємний дизайн і зручність користування. Він має бути адаптивним і коректно відображатися на різних типах пристроїв та в різних браузерах. Основні компоненти інтерфейсу включають:

* Форми реєстрації та авторизації із валідацією введених даних та повідомленнями про помилки;
* Особистий кабінет користувача з можливістю перегляду та редагування персональної інформації та зміни пароля;
* Сторінки словника для додавання, видалення та повторення слів;
* Сторінка з міні-тестами із таймером, підрахунком балів та відображенням результатів;
* Сторінки вправ, що включають вибір правильної відповіді та побудову речення, що допомагають закріпити граматичні та лексичні навички;
* Чат-бот на базі AI для інтерактивної взаємодії та навчання користувача.

3.1.2 Апаратний інтерфейс

Програмний продукт буде працювати на стандартних пристроях з операційними системами Windows, macOS, Linux. Не передбачається використання спеціального обладнання. Всі необхідні обчислювальні ресурси для роботи обробляються на сервері, клієнтською частиною є веб-браузер. Мінімальні апаратні вимоги для клієнтського пристрою:

* Процесор з тактовою частотою не менше 1.5 ГГц;
* Оперативна пам’ять – не менше 2 ГБ;
* Доступ до стабільного інтернет-з’єднання для взаємодії з сервером.

3.1.3 Програмний інтерфейс

Програмне забезпечення буде реалізоване за принципом клієнт-серверної архітектури. Клієнтська частина створюватиметься з використанням бібліотеки React, що дає змогу розробити динамічний і зручний інтерфейс користувача. Серверна частина працюватиме на платформі Node.js із фреймворком Express.js, який дозволяє ефективно обробляти HTTP-запити та легко масштабувати систему. Обмін даними між клієнтом і сервером відбуватиметься через REST API, що забезпечить чітку структуру запитів і стабільну взаємодію компонентів. Формат передавання даних — JSON, який є загальноприйнятим і зручним для веб-розробки. Для роботи з базою даних буде використано ORM Sequelize, який спрощує доступ до даних завдяки об’єктно-орієнтованому підходу. Також планується використання бази даних SQLite.

3.1.4 Комунікаційний протокол

Програмне забезпечення буде реалізовано за принципом взаємодії клієнтської та серверної частин через протокол HTTP або HTTPS. Для обміну даними між клієнтом і сервером використовуватиметься REST API з передачею інформації у форматі JSON. Планується впровадження механізму автентифікації користувачів за допомогою токенів (JWT), що дозволить забезпечити безпечний доступ до системи без повторного входу. З метою захисту персональних даних паролі користувачів будуть хешуватися за допомогою алгоритму bcrypt перед збереженням у базі даних. У фінальній версії, при розгортанні в мережі Інтернет, система використовуватиме безпечний протокол HTTPS для шифрування всього трафіку між клієнтом і сервером.

3.1.5 Обмеження пам’яті

Програма орієнтована на роботу на сучасних пристроях з мінімальними апаратними вимогами, що включають не менше 2 ГБ оперативної пам’яті на клієнтському пристрої. Серверна частина буде розгорнута з урахуванням обмежень доступної пам’яті на хостингу або сервері, з можливістю масштабування при збільшенні кількості користувачів. Для бази даних на початковому етапі планується використання SQLite, що має невеликі вимоги до ресурсів, з подальшим переходом на більш потужні СУБД, наприклад, MySQL чи PostgreSQL.

3.1.6 Операції

У процесі взаємодії з системою користувач зможе здійснювати такі дії:

* Зареєструвати новий обліковий запис, ввівши необхідні дані;
* Авторизуватися в системі з використанням електронної пошти та пароля;
* Переглядати, редагувати свій профіль, змінювати пароль або видаляти обліковий запис;
* Переглядати сторінку словника з можливістю перекладу та вивчення слів;
* Проходити інтерактивні вправи, такі як: вибір правильної відповіді, переклад слова, побудова речення;
* Отримувати результати виконання вправ;
* Користуватися функціями штучного інтелекту для покращення навчання.

3.1.7 Функції продукту

Програмний продукт повинен забезпечити реалізацію таких основних функцій:

* Реєстрація нових користувачів та вхід до особистого кабінету;
* Керування профілем: редагування особистих даних, зміна пароля, видалення облікового запису;
* Доступ до модуля словника для перегляду, вивчення та перекладу лексики;
* Проходження інтерактивних вправ, зокрема: вибір правильної відповіді, переклад слова, побудова речень;
* Обробка відповідей користувача та надання зворотного зв’язку щодо правильності;
* Взаємодія між клієнтською та серверною частинами через REST API з обміном даними у форматі JSON;
* Інтеграція з AI-сервісом (Google Gemini) для адаптації навчального контенту та підвищення ефективності навчання.

3.1.8 Припущення й залежності

* Користувач має доступ до сучасного веб-браузера, що підтримує JavaScript та актуальні вебтехнології;
* Пристрій користувача відповідає мінімальним технічним вимогам, необхідним для стабільної роботи вебзастосунку;
* Забезпечено стабільне інтернет-з’єднання для коректної взаємодії клієнтської частини з сервером у режимі реального часу;
* На етапі розробки база даних реалізується на SQLite;
* Програмне забезпечення включає інтеграцію з AI-сервісами для реалізації інтелектуальних функцій: адаптації матеріалу до рівня користувача, надання розумних підказок тощо;
* Усі взаємодії з системою (реєстрація, вхід, виконання вправ) реалізуються через REST API з використанням формату JSON та механізмів автентифікації.

3.2 Властивості програмного продукту

Програмний продукт, що розробляється, повинен володіти низкою ключових властивостей, які визначають його якість та відповідність очікуванням користувачів. По-перше, система повинна бути гнучкою та адаптивною. Це означає, що навчальний контент, зокрема вправи та складність діалогів з AI-асистентом, має динамічно підлаштовуватися під рівень знань, вказаний користувачем. По-друге, продукт має бути інтерактивним та залучаючим. Взаємодія з системою не повинна зводитися до пасивного споживання інформації. Користувач повинен активно брати участь у процесі: виконувати різноманітні завдання, вести діалог, самостійно формулювати відповіді та отримувати миттєвий зворотний зв'язок. По-третє, система повинна бути надійною та стабільною. Це передбачає коректну обробку помилок, таких як втрата інтернет-з'єднання або недоступність зовнішніх сервісів, та надання користувачеві зрозумілих повідомлень про стан системи. По-четверте, продукт має бути розширюваним. Архітектура повинна дозволяти в майбутньому легко додавати нові типи вправ, нові теми для обговорення в чаті та підключати інші мови для вивчення без необхідності кардинальної перебудови всієї системи.

3.3 Атрибути програмного продукту

3.3.1 Надійність

Система повинна залишатися працездатною при виникненні непередбачуваних ситуацій. Вона має коректно обробляти помилки, що надходять від зовнішнього Google Gemini API, і відображати користувачеві відповідне повідомлення, не перериваючи роботу всього додатку. У випадку помилок при збереженні даних у базу, система повинна інформувати про це користувача, а не завершуватися з критичною помилкою.

3.3.2 Доступність

Веб-додаток повинен бути доступним для користувачів 24 години на добу, 7 днів на тиждень, за винятком планових технічних робіт. Очікуваний показник доступності для продуктивного середовища має становити не менше 99.5%.

3.3.3 Безпека

Безпека є критично важливим атрибутом. Система повинна забезпечувати:

* конфіденційність даних користувача: всі дані, що передаються між клієнтом та сервером, повинні шифруватися за допомогою протоколу HTTPS;
* безпечне зберігання паролів: паролі користувачів не повинні зберігатися у відкритому вигляді, вони мають хешуватися за допомогою надійного алгоритму (наприклад, bcrypt);
* автентифікація та авторизація: доступ до персональних даних (словник, профіль) та захищених функцій (проходження вправ, чат) має надаватися лише автентифікованим користувачам, для цього використовується механізм на основі JWT (JSON Web Tokens);

3.3.4 Супроводжуваність

Кодова база проєкту має бути структурованою та добре документованою. Архітектурне рішення з чітким розділенням на клієнтську та серверну частини, а також модульна структура на обох сторонах (компоненти в React, маршрутизатори в Express) спрощують внесення змін, виправлення помилок та додавання нового функціонала в майбутньому.

3.3.5 Переносимість

Серверна частина, розроблена на Node.js, є кросплатформною і може бути розгорнута на будь-якій операційній системі (Windows, macOS, Linux). Клієнтська частина є веб-додатком і вимагає лише наявності сучасного веб-браузера, що робить її незалежною від платформи користувача.

3.3.6 Продуктивність

Система повинна забезпечувати комфортний час відгуку. Час завантаження основних сторінок не повинен перевищувати 3 секунди на середньостатистичному інтернет-з'єднанні. Відповідь сервера на запити клієнта (крім запитів до AI) не повинна перевищувати 500 мілісекунд. Час відповіді від AI-чату залежить від зовнішнього сервісу, але очікується в межах 2-5 секунд.

3.4 Вимоги бази даних

Перш за все, система повинна гарантувати цілісність даних. Це досягається шляхом використання механізмів первинних (Primary Key) та зовнішніх (Foreign Key) ключів у рамках реляційної моделі.

Другою вимогою є структуроване зберігання. База даних повинна мати чітко визначену схему, що складається з основних таблиць, які відповідають ключовим сутностям предметної області. Поля цих таблиць повинні використовувати відповідні типи даних, такі як INTEGER, VARCHAR, TEXT та DATETIME, для забезпечення коректності та типізації збереженої інформації, що є важливим для подальшої роботи ORM Sequelize.

Третя вимога – ефективність виконання запитів. Структура таблиць та їх індексація, зокрема автоматичне індексування первинних та унікальних ключів (наприклад, id та email у таблиці Users), повинні забезпечувати швидке виконання основних операцій. До таких операцій належать: вибірка користувача за ідентифікатором, пошук питань за рівнем та типом, отримання персонального словника, а також операції вставки та оновлення даних.

Четвертою вимогою є масштабованість. Хоча на початковому етапі для спрощення розробки та розгортання використовується файлова СУБД SQLite, спроєктована схема даних є універсальною. Завдяки використанню ORM Sequelize, система спроєктована таким чином, щоб у майбутньому можна було легко мігрувати на більш потужні клієнт-серверні СУБД. Вимоги до бази даних мають охоплювати всю систему, а не тільки такі як PostgreSQL або MySQL, без значних змін у бізнес-логіці додатку.

П'ятою вимогою є забезпечення можливості резервного копіювання. Повинна бути передбачена процедура для створення регулярних резервних копій файлу бази даних, що є критично важливим для запобігання втраті даних у разі апаратних збоїв чи інших непередбачуваних ситуацій.

3.5 Інші вимоги

Усі компоненти проєкту мають відповідати встановленим технічним вимогам. Програмний код повинен бути написаний згідно зі стандартами стилю для відповідної мови, бути структурованим, читабельним і містити коментарі в складних місцях для полегшення підтримки. Розробка має вестися з використанням системи контролю версій Git, а також передбачати наявність віддаленого репозиторію для командної роботи, резервного копіювання та відстеження змін. Серверна частина повинна реалізовувати логування ключових подій та помилок, що дозволить ефективно моніторити систему й вчасно реагувати на збої. Оскільки проєкт використовує зовнішній сервіс Google Gemini API, необхідно подбати про безпечне зберігання API-ключа через файл .env, не допускаючи його потрапляння у відкритий клієнтський код.