

Структура документа ТЗ:

1. Введение

- 1.1 Цель документа
- 1.2 Обзор проекта
- 1.3 Определения и аббревиатуры

2. Общее описание

- 2.1 Контекст приложения
- 2.2 Функции приложения
- 2.3 Характеристики пользователей
- 2.4 Ограничения

3. Требования

- 3.1 Функциональные требования
- 3.2 Нефункциональные требования (производительность, надежность, доступность)
- 3.3 Интерфейсные требования
- 3.4 Требования к архитектуре и технологическому стеку

4. Архитектура системы и технологический стек

- 4.1 Обзор архитектуры
- 4.2 Фронтенд
- 4.3 Бэкенд
- 4.4 База данных
- 4.5 Внешние интеграции (API)
- 4.6 Безопасность

5. Детальное описание функционала

- 5.1 Экраны приложения
- 5.2 Интеграция с API
- 5.3 Аутентификация и авторизация
- 5.4 Управление подписками и платежами

6. Тестирование

- 6.1 Методы тестирования
- 6.2 Тестирование

7. План разработки

- 7.1 Этапы и сроки

8. Приложения

- 8.1 Макеты экранов
- 8.2 Технические спецификации API
- 8.3 Шаблоны данных

9. Управление версиями и обновлениями

- 9.1 Стратегия управления версиями

10. Поддержка и обслуживание

- 10.1 Стратегия поддержки пользователей
- 10.2 План обслуживания

11. Юридические аспекты

- 11.1 Соблюдение законодательства

12. Заключение

- 12.1 Следующие шаги

Технологический стек:

- Фронтенд: React Native (для обоих Android и iOS), для создания нативных приложений с единой кодовой базой.
- Бэкенд: Node.js с использованием Express.js для создания RESTfull API.
- База данных: MongoDB или FirebaseDB(определить перед началом разработки) для гибкого хранения данных.
- Аутентификация: Firebase Auth для интеграции с AppleID, GoogleID, и Facebook.
- API интеграции: OpenAI для генерации текста, изображений и аудио.
- Платежная система: Apple/Google для управления подписками и платежами.
- Аналитика: Google Analytics для мобильных приложений, с возможностью подключения Firebase A/B testing, Firebase predictions.
- Push-уведомления: Firebase Cloud Messaging, Firebase In-App messaging.
- Безопасность: SSL /TLS для шифрования данных, OAuth для аутентификации.

1. Введение

1.1 Цель документа

Определить технические и функциональные требования к разработке мобильного приложения "Magic Tales" для генерации сказок с использованием AI, ориентированного на детскую аудиторию.

1.2 Обзор проекта

"Magic Tales" — это приложение для iOS и Android, которое позволяет пользователям создавать персонализированные детские сказки с помощью искусственного интеллекта. Приложение предлагает интерактивный и интуитивно понятный интерфейс для выбора элементов сказки и получения уникального рассказа.

1.3 Определения и аббревиатуры

- AI: Искусственный интеллект
- API: Программный интерфейс приложения
- UI/UX: Интерфейс пользователя / Опыт пользователя

2. Общее описание

2.1 Контекст приложения

Приложение интегрируется с OpenAI для генерации текстов, изображений и аудио, предоставляя уникальный контент для каждого пользователя.

2.2 Функции приложения

- Регистрация и вход через социальные сети и email
- Выбор параметров сказки (персонажи, тематика, мораль)
- Генерация текста, изображений и аудио сказки
- Сохранение ранее сгенерированного контента с привязкой к пользователю
- Управление подписками и платежами

2.3 Характеристики пользователей

Целевая аудитория — родители с детьми в возрасте от 3 до 10 лет, заинтересованные в образовательном и развлекательном контенте.

3. Требования

3.1 Функциональные требования

- ФТ1: Регистрация и аутентификация через AppleID, GoogleID, Facebook, email+pass регистрация/вход.
- ФТ2: Интеграция с API OpenAI для генерации сказок, изображений и голоса.
- ФТ3: Возможность выбора элементов сказки пользователем (герой, мораль, возраст и тд).
- ФТ4: Сохранение и просмотр истории сгенерированных сказок в приложении пользователем.
- ФТ5: Пользователи должны иметь доступ к своим данным, их исправление, удаление («право быть забытым»), ограничение обработки и передачу данных.

3.2 Нефункциональные требования

- НФТ1: .Оптимизированный и расширяемый код, отклик не более 30 сек
- НФТ2: Поддержка мультиязычности (Английский, Украинский, Испанский), возможность менять и/или добавлять языки в конфиге и в приложении).
- НФТ4: возможность и доступность простого и гибкого изменения Prompt для генерации текста и изображений со стороны заказчика.
- НФТ5: РЕализовать возможность добавления и редактирования условий подписок (Пример подписки - 10\$ за 10 историй)

3.3 Интерфейсные требования

- ИТР1: Интуитивно понятный и дружелюбный к детям интерфейс.
- ИТР2: Адаптивный дизайн для различных размеров экранов, ориентаций и платформ.

3.4 Требования к архитектуре и технологическому стеку

- AP1: Микросервисная архитектура для обеспечения масштабируемости и надежности.
- TC1: Использование Flutter для разработки кроссплатформенного приложения.

4. Архитектура системы и технологический стек

4.1 Обзор архитектуры

Система строится на микросервисной архитектуре с использованием контейнеризации для гибкости развертывания и масштабирования.

4.2 Фронтенд

- Технологии: flutter,
- Особенности: Компонентный подход, адаптивный дизайн.

4.3 Бэкенд

- Технологии: python fast api, .
- База данных: postgresql
- Внешние интеграции (API)
- OpenAI для генерации текста, изображений и аудио.

4.4 Безопасность

- Протоколы OAuth для аутентификации, SSL /TLS для шифрования данных.
- Реализовать процесс проверки на “стоп-слова”, после которых сказка(или ее часть) отправляется на повторную генерацию. После нескольких попыток уходит в ошибку.

5. Детальное описание функционала

5.1 Экраны приложения

- Экран знакомства: Представление основных функций и преимуществ приложения через серию вводных экранов. Включает кнопку "Продолжить", ведущую к экрану регистрации, и опцию пропуска.

- Экран регистрации/входа: Предоставляет возможность входа через социальные сети (AppleID, GoogleID, Facebook, email - регистрация, вход, восстановление пароля). Дизайн должен быть простым и интуитивно понятным. На этом этапе мы должны получить согласие на terms & conditions
- Главный экран (генерация сказки): Интерактивная форма для сбора информации о ребенке (имя, пол, возраст). Предусмотреть возможность просмотра, удаления и редактирования сохраненных профилей и просмотр ранее сгенерированных сказок.
- Экран тематики: Позволяет пользователю выбрать одну тематику для сказки из предложенного списка. Использовать визуализацию в виде плиток для удобного выбора.
- Экран выбора героев: Аналогично экрану тематики, предоставляет выбор главного героя из списка в виде плиток.
- Экран морали: Пользователь выбирает основную идею, или урок сказки. Представление в виде плиток для единообразия интерфейса.
- Экран генерации сказки: Анимированная заставка, отображающая процесс создания сказки. Включает информационные и мотивационные иллюстрированные сообщения для пользователя.
- Экран просмотра сказки: Отображение сгенерированной сказки с иллюстрациями и возможностью прокрутки. Должна быть предусмотрена возможность получения/покупки аудио версии сказки и возврата к главному экрану. В конце сказки дать возможность написать отзыв, с оправкой его на email заказчика приложения и с сохранением в базе данных.
- Экран истории сказок: Список ранее сгенерированных сказок с возможностью их повторного просмотра/прослушивания/покупки аудио, и отзыва.
- Экран настроек: Управление настройками приложения, включая выбор языка, просмотр статуса подписки(с возможностью удаления подписки и всех данных пользователя), доступ к обратной связи и управление профилями.

5.2 Интеграция с API

Для генерации сказок, изображений и аудио использовать API OpenAI, обеспечивая безопасность и конфиденциальность пользовательских данных. Необходима корректная обработка ответов API и их интеграция в контент приложения. Смотреть детальную блок-схему №1 отдельно.

5.3 Аутентификация и авторизация

Реализовать систему аутентификации через социальные сети с использованием протоколов OAuth для безопасного входа без передачи паролей и безопасного входа при авторизации через почту/пароль. Предусмотреть возможность восстановления утерянного пароля.

5.4 Управление подписками и платежами

Интегрировать revenue cat для управления подписками и одноразовыми покупками внутри приложения.

6. Тестирование

6.1 Методы тестирования

Реализовать интеграцию с **Firebase Test Lab & Performance Monitoring**.

6.2 Тестирование

Провести тестирование приложения перед релизом и устранить ошибки.

8. Приложения

8.1 Макеты экранов

Предоставить и согласовать с заказчиком детализированные макеты всех экранов приложения для визуализации дизайна и пользовательского интерфейса.

8.2 Технические спецификации API

Описать спецификации запросов в `swagger ui`

8.3 Шаблоны данных

Определить структуру данных, используемых в приложении, включая форматы хранения пользовательских профилей, сказок и истории покупок и других данных получаемых от пользователя или генерируемых для пользователя

9. Управление версиями и обновлениями

9.1 Стратегия управления версиями

Определить систему управления версиями для отслеживания изменений в коде и документации проекта. (github)

10. Поддержка и обслуживание

10.1 Стратегия поддержки пользователей

Создать систему поддержки пользователей, включая часто задаваемые вопросы (FAQ), возможность связи через отправку email.

Приложения:

Приложение А: Глоссарий

AI (Искусственный интеллект) - технологии, позволяющие машинам выполнять задачи, которые обычно требуют человеческого интеллекта, такие как обработка естественного языка, распознавание образов и принятие решений.

API (Application Programming Interface, Программный интерфейс приложения) - набор определений и протоколов для создания и интеграции программного обеспечения. API позволяет разным системам и приложениям взаимодействовать друг с другом.

UI/UX (User Interface/User Experience, Интерфейс пользователя / Опыт пользователя) - относится к дизайну и организации пользовательского интерфейса приложений, а также к общему опыту пользователя при взаимодействии с этим интерфейсом.

React Native - фреймворк для разработки кроссплатформенных мобильных приложений на JavaScript, позволяющий использовать одну и ту же кодовую базу для создания приложений для iOS и Android.

Node.js - программная платформа, основанная на движке V8 (открытый исходный код Google), позволяющая использовать JavaScript для создания серверных приложений.

Express.js - легковесный фреймворк для приложений Node.js, предоставляющий ряд мощных функций для создания веб-приложений и API.

MongoDB - NoSQL база данных, ориентированная на документы, предоставляющая высокую производительность, высокую доступность и легкую масштабируемость.

FirebaseDB - облачная NoSQL база данных для хранения и синхронизации данных в реальном времени между пользователями.

Firebase Auth - сервис для аутентификации пользователей, поддерживающий вход через социальные сети, такие как Google, Facebook и Twitter, а также через электронную почту и пароль.

Google Analytics - сервис для сбора и анализа статистики посещений веб-сайтов и мобильных приложений.

Firebase Cloud Messaging (FCM) - кроссплатформенный сервис обмена сообщениями, который позволяет надежно передавать сообщения и уведомления между серверами и клиентскими приложениями на Android, iOS и веб-платформах.

SSL/TLS (Secure Sockets Layer / Transport Layer Security) - криптографические протоколы, обеспечивающие защищенное соединение между клиентом и сервером в интернете.

OAuth - открытый стандарт для авторизации, который позволяет пользователям предоставлять сторонним приложениям ограниченный доступ к своим учетным записям на других сервисах, без необходимости раскрывать свои пароли.

Схема №1: Дополнительные материалы

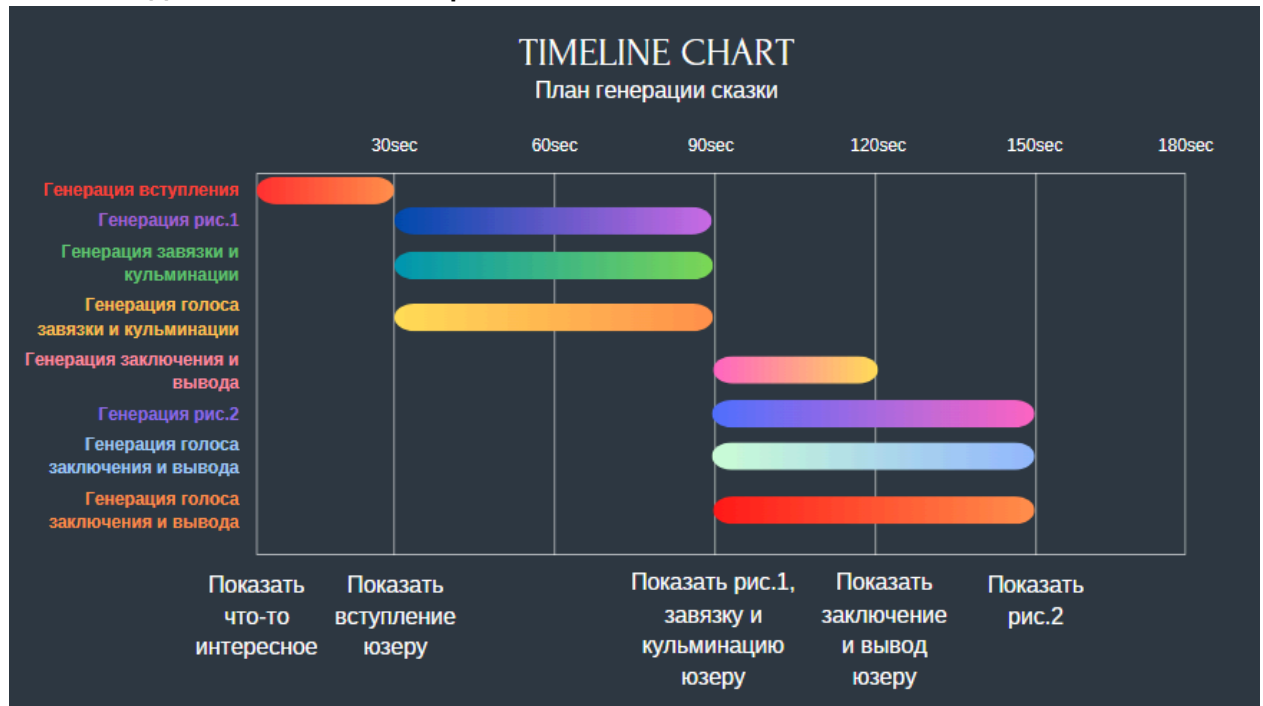


Схема №2: Путь клиента в приложении и примеры экранов и взаимосвязи между ними

https://miro.com/app/board/uXjVKbiNblw=?share_link_id=919953225503

Варианты подписок

1. **Бесплатная** - входит 2 генерации новой сказки + аудио озвучка
2. **10 сказок** - цена 9.99\$ (без аудио озвучки)
3. **30 сказок** - цена 29.99\$ (с аудио озвучкой)
4. **50 сказок** - цена 44.99\$ (с аудио озвучкой)