**1. Краткое описание проекта (суть)**

Название мобильного приложения: WordUp

Описание: Создание кроссплатформенного мобильного приложения для эффективного изучения иностранных слов с использованием методики интервального повторения.

Цель:

* Предоставить пользователю интуитивно понятный интерфейс для изучения слов
* Реализовать систему прогресса и мотивации
* Обеспечить работу в офлайн-режиме
* Создать базу данных с словарным запасом

Аналоги: Doulingo

**2. Основные требования (Функциональные)**

**2.1. Пользовательские роли**

✅ Регистрация нового пользователя

✅ Авторизация существующего пользователя

✅ Восстановление пароля

✅ Вход через социальные сети (Google/Apple)

✅ Запомнить меня

**2.2. Базовый функционал (MVP)**

✅ Выбор языка изучения (Английский, Армянский, Русский и др.)

✅ Система категорий (Путешествия, Еда, Работа и т.д.)

✅ Карточки слов с переводом и произношением

✅ Система интервального повторения

✅ Отметки "Знаю"/"Повторить"

✅ Прогресс изучения по категориям

### **Модуль практики**

✅ Тесты на выбор правильного перевода

✅ Игра "Word Rain" (выбор падающих слов)

✅ Статистика правильных ответов

✅ Таймер на ответы

### **Модуль статистики**

✅ Количество выученных слов

✅ Серия дней (streak)

✅ Прогресс по дням/неделям

✅ Достижения и бейджи

✅ Графики прогресса

**2.3. Нефункциональные требования**

* Время запуска приложения: < 3 секунд
* Время перехода между экранами: < 0.5 секунд
* Поддержка 1000+ слов в базе
* Работа на устройствах с 2GB RAM

**3. Предложенный стек технологий (рекомендация для сдачи в сроки)**

* Фронтенд:
* Kivy/KivyMD - фреймворк UI
* Python 3.8+ - основной язык
* KV Language - разметка интерфейса

Бэкенд варианты:

* SQLite - локальная база данных
* MySQL + REST API - облачное решение
* Flask - веб-сервер

**4. Рекомендуемые инструменты разработки**

* Pycharm.
* Figma — прототипы / дизайн.
* GitHub — исходники + Actions.
* Flutter DevTools — отладка.
* Postman / REST Client — если есть API.

**5. Архитектура и данные**

**5.1. Архитектурная схема (вкратце)**

MVVM Pattern:

Model - database.py, api\_client.py

View - \*.kv files

ViewModel - main.py (Screen classes)

**5.2. Структура БД**

****Таблицы:****

* users - пользователи
* languages - языки
* categories - категории
* lessons - уроки
* words - слова
* user\_progress - прогресс
* user\_stats - статистика

### **6.Требования к производительности**

* Время запуска приложения: < 3 секунд
* Время перехода между экранами: < 0.5 секунд
* Поддержка 1000+ слов в базе
* Работа на устройствах с 2GB RAM

### **6.1. Поддерживаемые платформы**

✅ Android 8.0+

✅ iOS 12+ (через Kivy-iOS)

✅ Windows 10/11

✅ macOS 10.14+

### **6.2. Адаптивность**

* Поддержка portrait ориентации
* Разрешения: 720x1280 - 1440x2960
* Плотность пикселей: mdpi - xxxhdpi

### **7. Защита данных**

* Хеширование паролей (bcrypt)
* HTTPS для API запросов
* Локальное хранилище чувствительных данных
* Валидация входных данных

### **8.  Виды тестирования**

* **Модульное тестирование** - тесты отдельных функций
* **Интеграционное тестирование** - тесты взаимодействия модулей
* **UI тестирование** - тесты интерфейса
* **Нагрузочное тестирование** - тесты производительности

### **8.2. Критерии приемки**

* Приложение не крашится при стандартном использовании
* Все кнопки и навигация работают корректно
* Данные сохраняются между сессиями
* Интерфейс отзывчивый и плавный

**9. Релиз**

* Репозиторий: Git (feature-branch workflow).
* Pull Request template + code review.
* GitHub Actions: запускает lint, unit tests, билд apk/ipa.
* Автоматизированные сборки для тестовых релизов (Internal testing).
* Release: Google Play / RuStore.

**10. Безопасность и приватность**

* Все запросы по HTTPS.
* Токены в безопасном хранилище (Flutter secure\_storage).
* Контроль доступа на уровне БД (Firestore security rules).
* Ограничение загрузки файлов (size/type), сканирование расширений.
* Возможность удаления аккаунта / данных.

## **11. Этапы разработки**

### **Этап 1: Базовый функционал (2 недели)**

* Прототип интерфейса
* Навигация между экранами
* Локальная база SQLite
* Система аутентификации

### **Этап 2: Основные функции (3 недели)**

* Модуль изучения слов
* Система прогресса
* База слов и категорий
* Статистика и достижения

### **Этап 3: Дополнительные возможности (2 недели)**

* Игры и практика
* REST API интеграция
* Уведомления и напоминания
* Cloud синхронизация

### **Этап 4: Тестирование и публикация (1 неделя)**

* Тестирование на реальных устройствах
* Исправление багов
* Публикация в магазины приложений



