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

Название мобильного приложения: EIC

Описание: мобильное приложение для студентов, объединяющее в себе расписание, задания, даты экзаменов и сообщения из eljur и Telegram.

Цель: создать удобный и безопасный инструмент для получения и хранения необходимой информации в одном месте.

Аналоги: eljur, Telegram, Todoist и Google Tasks.

Дата сдачи — 28.09.2025

**2. Основные требования (Функциональные)**

**2.1. Пользовательские роли**

* Пользователь: вход, настройка интеграций (Элжур, Telegram), просмотр расписания, управление заданиями, просмотр экзаменов и сообщений.
* Администратор: отсутствует.

**2.2. Базовый функционал (MVP)**

1. Авторизация через аккаунт Элжур (логин/пароль, хранятся локально).
2. Привязка к Telegram (для получения сообщений и файлов из канала).
3. Расписание: просмотр на день, переключение по дням, отображение замен.
4. Задания: актуальные и архивные, сортировка по предметам, возможность прикреплять файлы и ссылки.
5. Экзамены: список с датами, предметами, местом и принимающим.
6. Сообщения: вкладки Элжур / Telegram, ручное добавление.
7. Настройки: привязка аккаунтов, очистка БД, выбор темы, справка.

**2.3. Нефункциональные требования**

* Кроссплатформенность: Android 8+.
* Конфиденциальность: все данные хранятся только локально (SQLite/Room).
* Локализация: RU.
* Удобство использования: интуитивный интерфейс, минималистичный дизайн.

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

* Клиентская часть: Kotlin (Android)
* База данных: SQLite
* API и интеграции:
  + Telegram API
  + Парсинг данных из Элжура

Обоснование выбора: Kotlin — натив для Android; Room — удобный ORM над SQLite; локальное хранение повышает приватность.

**4. Рекомендуемые инструменты разработки**

* IDE: Android Studio.
* Figma — прототипы / дизайн.
* GitHub — исходники + Actions.
* Контроль версий: Git
* Postman / REST Client — если есть API.

**5. Архитектура и данные**

**5.1. Архитектурная схема (вкратце)**

Приложение (Kotlin, Android) ↔ Room (SQLite) ↔ API (Telegram, Элжур).

**5.2. Структура БД**

* teachers: { teacher\_id, name }
* subjects: { subject\_id, name, teacher\_id }
* schedule: { schedule\_id, date, subject\_id, start\_time, end\_time, replacement\_id }
* tasks: { task\_id, subject\_id, title, description, deadline, status enum('active','archived') }
* task\_attachments: { attachment\_id, task\_id, file\_url, type enum('file','image','link') }
* exams: { exam\_id, subject\_id, exam\_date, location, teacher\_id }
* messages: { message\_id, source enum('eljur','telegram','custom'), content, received\_at }
* settings: { setting\_id, theme enum('light','dark') }
* integrations: { integration\_id, service enum('eljur','telegram'), login, password\_enc, token }

**6. API / Endpoints**

* POST /api/register — регистрация пользователя (создание локального профиля, если есть несколько).
* POST /api/login — авторизация (по сохранённым данным Элжур / Telegram).
* GET /api/schedule — получение расписания (локально + из Элжура).
* POST /api/schedule — добавление/изменение расписания вручную.
* GET /api/replacements — получение замен (локально + из Telegram).
* GET /api/tasks — список заданий.
* POST /api/tasks — создание задания.
* PUT /api/tasks/{id} — редактирование задания.
* DELETE /api/tasks/{id} — удаление задания.
* GET /api/exams — список экзаменов и зачётов.
* POST /api/exams — добавление экзамена.
* GET /api/messages — получение сообщений (Элжур / Telegram / ручные).
* POST /api/messages — добавление заметки вручную.
* GET /api/settings — просмотр настроек.
* PUT /api/settings — изменение настроек (тема, очистка БД, привязки).

**7. UI/UX: основные экраны + требования**

1. Расписание с изменениями
2. Задания (актуальные и архивные).
3. Аттестационные мероприятия
4. Экран управления категориями
5. Сообщения и прочая информация
6. Настройки

Дизайн-концепция: интуитивно понятный минималистичный интерфейс.

**8. Тестирование**

* Модульное тестирование критических компонентов
* Интеграционное тестирование с Telegram API.
* Тестирование UI (переключение вкладок, добавление/удаление задач).
* Пользовательское тестирование на удобство.

**9. Релиз**

* Репозиторий: Git
* Сборка: APK для Android
* Тестирование: внутреннее тестирование
* Release: RuStore.

**10. Безопасность и приватность**

* Пароли хранятся локально.
* Отсутствие передачи данных на сторонние сервера

**11. Роадмап / план по дням с конкретными датами**

**Sprint 0 — Подготовка / Discovery (23 — 28 сентября 2025) — 6 дней**

**Цели:** финальные требования, приоритеты, дизайн-гайд, инфраструктура.

* 23–24.09: Сбор требований, определение MVP vs Stretch, согласование форматов файлов.
* 25.09: Составление ТЗ (этот документ), определение команды и ролей.
* 26–27.09: Быстрые вайрфреймы (главные экраны) + прототип Figma (главный поток).
* 28.09: Настройка среды разработки и репозитория.

**Доставка:** финализированные требования, Figma прототипы главных экранов, рабочий репозиторий.

**Sprint 1 — Core: расписание и задания (29 сентября — 12 октября 2025) — 2 недели**

**Задачи:**

* Реализовать экран расписания (просмотр по дням, добавление вручную).
* Создать экран заданий (актуальные/архив).
* Подключить локальную БД (Room/SQLite).

**Доставка**: приложение с расписанием и системой заданий.

**Sprint 2 — Экзамены и сообщения (13 — 26 октября 2025) — 2 недели**

**Задачи:**

* Экран экзаменов и зачётов (сортировка по датам).
* Экран сообщений (Элжур, Telegram, ручные заметки).
* Возможность прикрепления файлов и ссылок к заданиям.

**Доставка**: экзамены и сообщения работают, есть тестовый ввод данных.

**Sprint 3 — Настройки и интеграция (27 — 9 ноября 2025) — 2 недели**

**Задачи:**

Экран настройки (привязка аккаунтов, очистка БД, смена темы).

Интеграция с Telegram API (получение новых сообщений).

Интеграция с Элжур (получение расписания и дз).

**Доставка**: интеграции и базовые настройки работают.

**Sprint 4 — Интеграция и тестирование (10 — 23 ноября 2025) — 2 недели**

**Задачи:**

* Интеграция всех модулей (расписание + задания + экзамены + сообщения).
* Тестирование работы локальной БД.
* Проверка UX (удобство экранов, переключение вкладок).

**Доставка**: приложение работает целиком, возможны баги.

**Stabilization & UAT fixes (24 ноября — 2 декабря 2025) — 9 дней**

**Задачи:**

* Исправление багов по результатам тестирования.
* Подготовка презентации / демонстрации (скриншоты, видео, инструкции).
* Подготовка пакета сдачи (apk/ipa, инструкции по запуску, тестовые учётные записи).

**Доставка:** финальные билды и документация.

**Final polishing & submission (3 — 6 декабря 2025) — 3 дня (включая саму сдачу)**

**Задачи:**

* Финальные баг-фиксы.
* Подготовка демонстрации/презентации (3–7 минут).
* Отправка/передача приложения и документации преподавателю.

**Финиш:** сдача до **6.12.2025** (включительно).

**12. Уникальные функции проекта (если останется время)**

* Все учебные данные в одном месте (расписание, дз, экзамены, замены, сообщения).
* Локальное хранение (безопасность и приватность).
* Удобная визуальная сортировка (по датам, по предметам).
* Простое добавление вручную (замены, задания, заметки).