AC + NFR
Проект: Telegram-бот рекомендаций книг «AI-Book»

1. Критерии приемки

User Story:

Как читатель, я хочу получать в Telegram-боте «AI-Book» персональные рекомендации книг с кратким объяснением «почему», чтобы быстрее находить интересные книги и сохранять их на свою полку.

Use Case (основной сценарий):

1. Пользователь отправляет команду /recommend в Telegram-бот.

2. Система (API рекомендаций) подбирает 1-5 релевантных книг на основе профиля и истории.

3. Бот отправляет карточки книг: обложка, название, автор(ы), жанры, язык, описание, «почему».

4. Пользователь может нажать: «Лайк», «Скрыть», «Ещё»; действия влияют на следующие выдачи.

Номер кейса: 1

Функциональность: Автоматическая выдача рекомендаций по запросу /recommend

Дано: Пользователь имеет активный чат с ботом; профиль заполнен (жанры/авторы).

Когда: Пользователь отправляет команду /recommend.

Тогда:

- бот получает от API 1-5 книг; каждая содержит: Title, Authors, Genres, Description, CoverURL, Reason;

- карточки показываются с кнопками «Лайк», «Скрыть», «Ещё»;

- в БД сохраняются записи Recommendations (userId, bookId, rank, score, reason, ts);

- время появления первой карточки - не более 5 секунд.

Номер кейса: 2

Функциональность: Онбординг и настройки предпочтений

Дано: Новый пользователь отправил /start; каталог книг доступен.

Когда: Бот запрашивает базовые предпочтения (жанры/авторы).

Тогда:

- выборы сохраняются в профиле пользователя;

- следующие выдачи учитывают новые предпочтения;

- пользователь может изменить настройки в любое время через меню.

Номер кейса: 3

Функциональность: Управление карточками (лайк/скрыть/ещё)

Дано: Показана карточка книги в ответ на /recommend.

Когда: Пользователь нажимает одну из кнопок.

Тогда:

- «Лайк» - книга появляется на полке; веса профиля усиливаются по соответствующим признакам;

- «Скрыть» - книга исключается из будущих подборок для этого пользователя;

- «Ещё» - формируется дополнительная выдача или предлагается перейти к /shelf;

- все действия фиксируются в журнале событий.

Номер кейса: 4

Функциональность: Полка /shelf

Дано: У пользователя есть сохранённые книги.

Когда: Пользователь вызывает /shelf.

Тогда:

- бот показывает список книг с пагинацией (по 10 на страницу);

- можно удалить книгу с полки и открыть её описание;

- если полка пуста — бот подсказывает воспользоваться /recommend.

Номер кейса: 5

Функциональность: Поиск по каталогу

Дано: Каталог доступен (книги, авторы, теги).

Когда: Пользователь отправляет запрос вида: поиск <текст>.

Тогда:

- бот возвращает до 10 карточек;

- сортировка: релевантность - популярность - новизна;

- при отсутствии совпадений - понятное сообщение и совет изменить запрос.

Номер кейса: 6

Функциональность: Исключительные ситуации и деградация

Дано: Система работает в штатном режиме, но внешний сервис может быть недоступен.

Когда: Происходит ошибка LLM или API рекомендаций.

Тогда:

- если LLM недоступен - карточки приходят без поля Reason, пользователь уведомлён;

- если API недоступен - бот показывает понятное сообщение и выполняет до 2 повторов;

- все ошибки фиксируются в журнале и доступны по пользователю.

2. Нефункциональные требования

Требования производительности:

- Время отклика бота на команду (эхо/меню) - < 2 секунд.

- Появление первой карточки после /recommend - ≤ 5 секунд.

- Время ответа LLM (объяснение «почему») - < 2.5 секунд.

- API рекомендаций обрабатывает до 50 одновременных пользователей без деградации p95.

Требования доступности/надёжности:

- Доступность пользовательского API - ≥ 99.5%/месяц (24/7).

- При полном отказе LLM выдача продолжается без Reason, без остановки сервиса.

- Повторы на 5xx/timeout - до 2 попыток с экспоненциальной задержкой.

- Логи действий и ошибок пишутся в реальном времени (не теряются при сбое).

Требования безопасности:

- Доступ к рекомендациям и логам - только авторизованным ролям.

- Секреты (Telegram/LLM) хранить в ENV/хранилище секретов; не логировать.

- Шифрование трафика между компонентами - HTTPS/SSL.

- Логи не содержат текста переписки, только метаданные событий.

Требования к данным и совместимости:

- Схема БД соответствует ERD (Users, Books, Authors, Genres, Tags, Recommendations).

- В Recommendations обязательны: userId, bookId, rank, score; reason - опционально.

- REST-контракты соответствуют Swagger.

- Поддержка актуальной версии Telegram API.

Требования наблюдаемости и эксплуатации:

- Метрики: латентности API и LLM, error-rate, доля пустых выдач.

- Трейсинг с единым traceId: бот - API - LLM - API - бот.

- Алерты: выше цели 5 минут; error-rate > 3% за 10 минут.