Пояснительная записка к проекту BookTrade  
1. Введение  
BookTrade — это веб‑приложение для обмена книгами между пользователями. Цель проекта — создать удобную площадку, где любители чтения могут предлагать свои книги на обмен, находить интересные издания и взаимодействовать друг с другом.  
2. Актуальность  
Повышение интереса к устойчивому потреблению: обмен книгами способствует повторному использованию и снижению бумажных отходов.  
Экономическая выгода: пользователи экономят на покупке новых изданий.  
Социальное взаимодействие: проект объединяет книголюбов в онлайн‑сообществе.  
3. Цель и задачи проекта  
Цель: разработать полнофункциональную платформу для обмена книгами.  
Задачи:  
Организация регистрации и безопасного входа пользователей.  
Реализация каталога книг с возможностью добавления, редактирования и удаления записей.  
Поддержка загрузки обложек и ссылок на онлайн‑версии.  
Предоставление REST‑API для интеграции с внешними сервисами.  
Настройка фоновых задач (ежедневная рассылка новинок).  
Обеспечение простого развёртывания (Docker и CLI‑скрипт).  
4. Описание архитектуры и структуры  
Проект имеет модульную структуру:  
app/  
models/ — ORM‑модели (User, Book).  
routes/ — Blueprint‑ы для страниц: [auth.py](https://auth.py/) (регистрация/вход), [main.py](https://main.py/) (основной интерфейс).  
api/ — Flask‑RESTful маршруты.  
forms/ — WTForms‑формы для регистрации, входа и добавления книг.  
templates/ — Jinja2‑шаблоны для страниц.  
static/ — статика и папка uploads для обложек.  
scripts/ — утилиты на argparse: создание админа, импорт книг из JSON, seed данных.  
tasks/ — фоновые задачи на schedule (ежедневная рассылка).  
instance/ — каталог с файлом базы данных SQLite (booktrade.db).  
Dockerfile и docker‑compose.yml — контейнеризация приложения.  
[run.sh](https://run.sh/) — скрипт для локального развёртывания без Docker.  
5. Используемые технологии  
Язык и фреймворк: Python 3, Flask  
База данных: SQLite (Flask‑SQLAlchemy, Flask‑Migrate)  
Авторизация и безопасность: Flask‑Login, Flask‑WTF (CSRF), email-validator  
REST‑API: Flask‑RESTful  
Файловый ввод/вывод: загрузка изображений (Werkzeug), работа с JSON и zip (stdlib)  
Командная строка: argparse  
Фоновые задачи: schedule  
Frontend: Bootstrap 5, Jinja2  
Контейнеризация: Docker, docker‑compose  
6. Особенности реализации  
Регистрация и вход: безопасное хэширование паролей, валидация email.  
Добавление книг: поддержка обложек, URL online‑версии, описание и состояние.  
REST‑API: методы GET /api/books и POST /api/books.  
Фоновые задачи: ежедневное выполнение send\_daily\_email().  
CLI‑скрипты: автоматизация задач — создание администратора, импорт и seed.  
Развёртывание: готовый Docker‑образ, простой скрипт [run.sh](https://run.sh/).  
7. Развёртывание и запуск  
Через Docker:  
docker-compose up --build   
Локально без Docker:  
chmod +x run.sh ./run.sh   
8. Результаты и выводы  
Разработан прототип полноценной платформы обмена книгами.  
Удобный интерфейс и расширяемая архитектура.  
Платформа готова к масштабированию: переход на PostgreSQL, добавление модулей уведомлений.  
9. Перспективы развития  
Интеграция с внешними сервисами (Goodreads, Google Books API).  
Система рейтингов и отзывов пользователей.  
Добавление геолокации и карты для локального обмена.  
Мобильное приложение на React Native или Flutter.  
Пояснительная записка составлена в рамках учебного проекта.