**Название проекта**

Mythril — веб-приложение в формате фриланс-биржи.

**Автор проекта**

Степанцев Андрей Станиславович

Описание идеи

Mythril — это платформа для поиска и размещения фриланс-заказов, ориентированная на простоту использования и прозрачность взаимодействия между заказчиком и исполнителем. Проект реализует ключевые функции фриланс-биржи: создание и поиск заданий, отклики с предложениями цены, выбор исполнителя и контроль выполнения заказа.

**Описание реализации**

Приложение написано на Python, через фреймворк Flask, SQLAlchemy для работы с базой данных. Реализована ролевая модель доступа: гость может просматривать задания, пользователь — создавать и откликаться на них, а администратор — управлять контентом на платформе.

Для хранения информации используются четыре основные таблицы: users, jobs, responses, reviews. Пользователи могут оставлять отклики на задания с комментарием и ценой, а заказчик выбирает одного исполнителя. После выбора статус задания меняется на "В работе" и исполнитель получает контакты заказчика для связи, при подтверждении завершения — статус меняется на "Завершён", а заказ появляется в профиле исполнителя, по запросу типа

completed\_jobs = session.query(Jobs).filter(Jobs.executor\_id == user.id, Jobs.status == "Завершён").order\_by(Jobs.created\_date.desc()).limit(2).all().

Отклики можно удалять и отменять. Задания с разными статусами сортируются: открытые выше завершённых, а внутри — по дате или количеству откликов

(jobs = query.options(joinedload(Jobs.author)).order\_by(status\_order, Jobs.created\_date).all()). Работы можно фильтровать, для этого используются запросы query = query.filter(Jobs.category == category)

В проекте есть реализация переключения между светлой и тёмной темами оформления, а также продуманная система сортировки и фильтрации заданий. Темы прописаны в css.

На главной странице можно выбрать валюту для отображения цены, курс будет браться по API запросу к <https://cdn.jsdelivr.net/npm/@fawazahmed0/currency-api@latest/v1/currencies/rub.json>

Запросы будут логгироваться в отдельный файл, через logging

**Используемые технологии и библиотеки:**

* Python 3.x
* Flask
* SQLAlchemy
* Jinja2
* HTML/CSS
* SQLite
* Requests
* Logging
* Дополнительные библиотеки: flask-login, flask-wtf

Файл с требованиями будет в проекте.

**Скриншоты:** 