**Введение (Описание целей проекта)**

В течение полугода я активно развивал навыки работы в области науки о данных. В итоге у меня накопилось достаточно знаний о классическом машинном обучении, но после этого я захотел каким-либо образом предоставить доступ к моим обученным моделям. Наиболее оптимальным вариантом я посчитал создание приложения, используя язык программирования Python и внешние библиотеки. В итоге целью моей курсовой работы является проведение полноценного пайплайна разработки ML-приложения.

**ОСНОВНАЯ ЧАСТЬ**

# ТЕХНОЛОГИЧЕСКАЯ ЗАДАЧА

## Список решаемых задач

Во-первых, нужно найти набор данных, для последующего обучения модели машинного обучения на его основе. Готовые наборы данных можно найти на по адресу <https://www.kaggle.com/datasets>.

Во-вторых, должна быть платформа для разработки непосредственно моделей машинного обучения и последующей загрузки в базу данных.

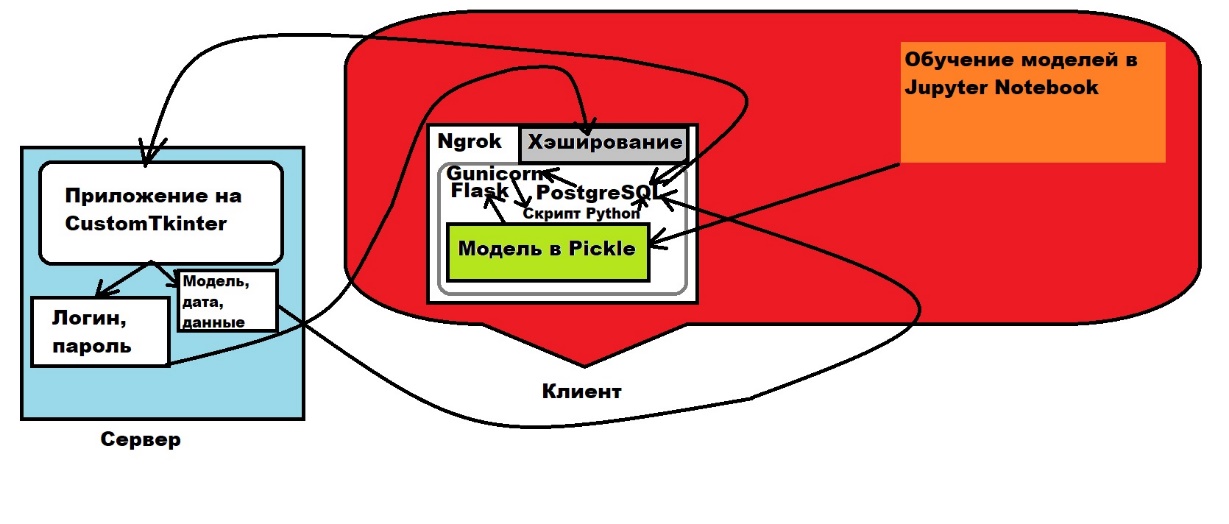
В-третьих, необходимо создать интерактивное приложение, которое позволит пользователю зарегистрироваться/войти по логину и паролю, выбрать нужную ему модель, вписать данные в удобном формате и получить результат с сервера (который загрузит и обработает модель из базы данных), а также просматривать свою историю запросов.

Приложение включает в себя серверную и клиентскую часть.

Серверная часть будет храниться на виртуальной машине в Docker контейнере, а клиентская - устанавливаться на компьютеры пользователей и обращаться к серверу.

### Подключаемые модули, платформы и т. д.

Примерная архитектура проекта приведена на схеме (устаревшая, требует изменений):



В качестве платформы для разработки моделей машинного обучения была выбрана Jupyter Notebook. Jupyter Notebook – это удобное веб-приложение, которое позволяет одновременно писать код, текст и строить графики. Так как в данном приложении код и текст размещены в ячейках, оно является довольно удобным инструментом и пользуется популярностью среди специалистов в сфере науки о данных.

Готовые модели очищаются от лишнего текста, текстовых выводов и комментариев и преобразовываются в двоичные файлы с помощью библиотеки Python под названием Pickle.

Для работы с базой данных используется модуль Python Sqlite3.

Клиентская часть приложения работает на микрофреймворке Flask, а в качестве HTTP-шлюза используется WSGI-сервер Gunicorn, так как Flask предназначен только для разработки и тестового сервера, но не для его постоянного использования с пользователями.

**Про хэширование паролей**

Для того, чтобы клиент мог подключаться к серверу, используется инструмент Pagekite. Он позволяет привязать локальный порт сервера к публичному адресу в сети. Таким образом, клиентская часть приложения будет автоматически обмениваться данными с сервером по открытому адресу.

# РЕАЛИЗУЕМАЯ архитектура

## Модули программы, функции, структуры данных

Описания и диаграммы

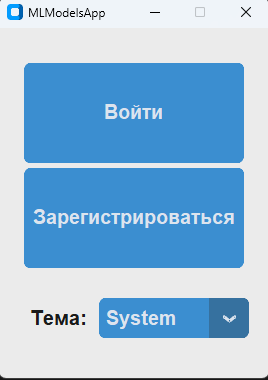
# ОПИСАНИЕ РАБОТЫ

## Экран приложения

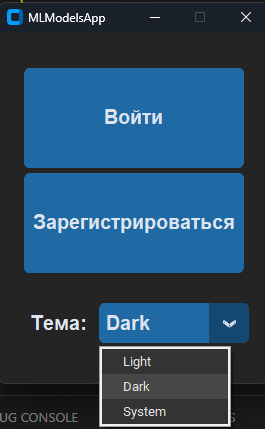
На данном этапе для запуска приложения используется сервер разработки. Он запускается путём исполнения файла ./server/server.py с помощью интерпретатора Python.

Во время работы сервера можно запускать графический интерфейс, расположенный по адресу ./client/main.py.

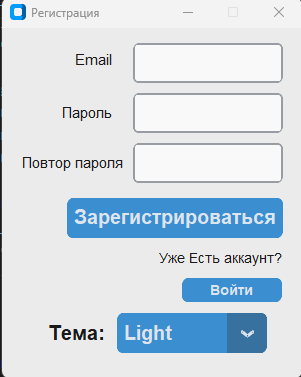
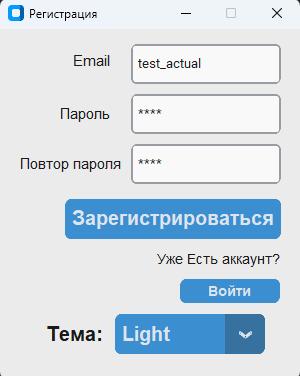
Вот как выглядит приветственная страница:

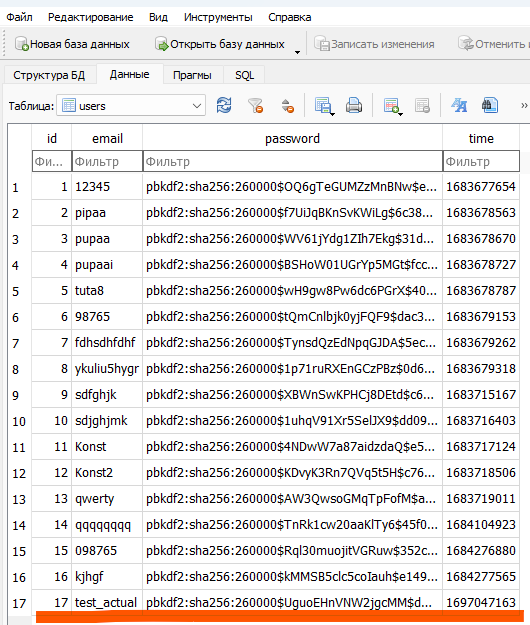


Пользователь может сменить тему, выбрав из списка:



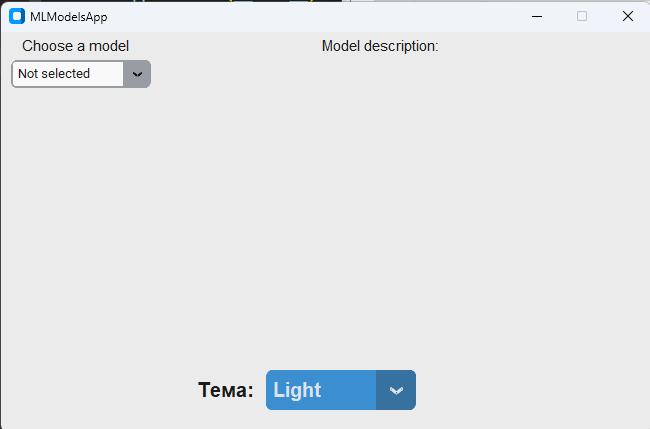
При нажатии на кнопку “Зарегистрироваться” происходит переход в другую часть приложения, отвечающую за запись новых пользователей в базу данных:



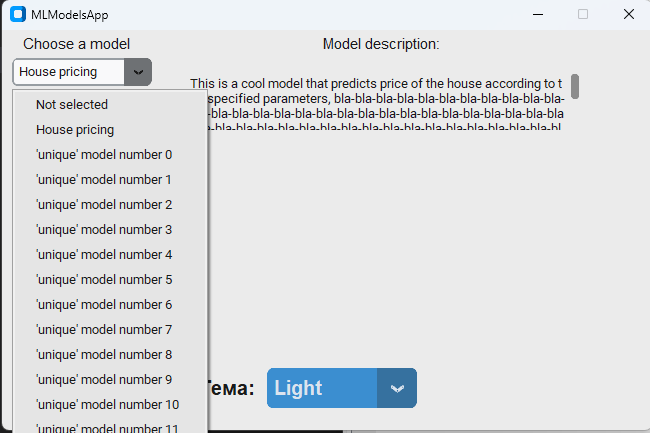


Как видно из таблицы выше, произошла запись введённых данных в таблицу ./server/flsite.db.

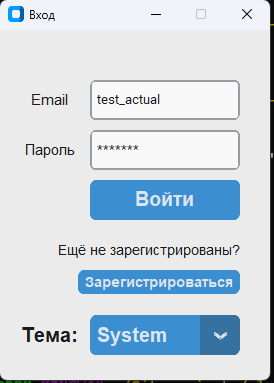
После регистрации пользователь мгновенно оказывается в основном окне приложения:



Под надписью “Choose a model” расположен список имеющихся на сервере моделей. При выборе каждой из них отображается описание, относящееся к ней:



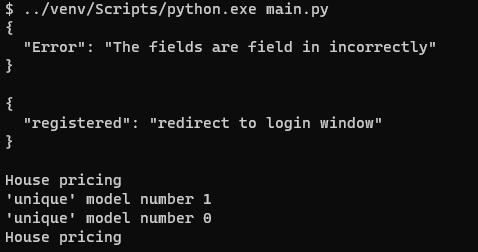
Также имеется возможность войти в приложение по данным пользователя в окне входа:

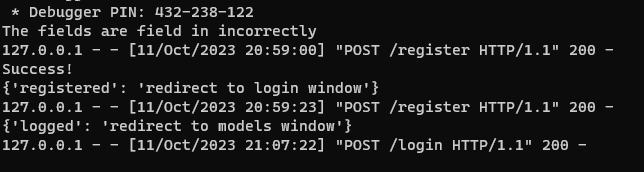


После нажатия кнопки “Войти” пользователь также попадает на основную страницу.

Как в окне регистрации, так и авторизации можно увидеть ещё по одной кнопке: они переадресуют пользователя на окно авторизации и регистрации соответственно, в зависимости от того, на какой странице пользователь находится сейчас.

При различных действиях пользователя на экран консоли и сервера и приложения выводятся логи (совершённые действия), но на данном этапе разработки они скорее являются отладочной информацией и не сохраняются в отдельный файл:





# ВЫВОДЫ

# БИБЛИОГРАФИЯ