Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего образования
«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»

Институт радиоэлектроники и информационных технологий – РТФ

Школа бакалавриата

ОТЧЕТ

По проекту
«Разработка веб-сервиса для взаимодействия водителей и автомоек»

по дисциплине «Проектный практикум»

Заказчик: Акмалетдинов Д.С.

Куратор: Харисов А.Р.

Студенты команды Коллектив 3826

Андреев С.А.

Калугин М.А.

Нургатин М.М.

Панарин И.Д.

Екатеринбург, 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель нашего проекта заключается в создании веб-сервиса для взаимодействия водителей и автомоек.

Главные задачи проекта – это написание мобильного и веб-приложения, с которыми будут взаимодействовать пользователи нашего продукта, а также написание API и базы данных для обработки и хранения данных автомоек и автовладельцев.

Актуальность проекта заключается в решении проблемы неорганизованного потока клиентов автомоек и отсутствия удобных цифровых сервисов для бронирования. Важность проекта определяется его способностью упростить жизнь автовладельцам, обеспечив быстрый доступ к услугам, и помочь владельцам моек оптимизировать загрузку и увеличить доходы. Сервис способствует цифровизации отрасли, делая взаимодействие между клиентами и автомойками более прозрачным и эффективным.

Продукт будет состоять из двух модулей: для владельцев автомоек (партнеров) и для автовладельцев (клиентов). Клиенты смогут забронировать время и услугу на удобной для них автомойки, а партнеры смогут равномерно распределить нагрузку на автомойку и повысить продажи, путем того, что клиенты могут узнать о их сервисе в нашем сервисе.

Как результат проекта мы планируем MVP нашего сервиса, а именно веб и мобильное приложения с набором базовых функций и API для обслуживания этих приложений.

1 Аналитика

1.1 Распределение по ролям

В нашей команде четыре человека и мы решили поделить роли в данном порядке:

– Калугин Максим – тимлид и DevOps

– Панарин Илья – дизайнер и frontend разработчик

– Нургатин Марат – мобильный разработчик

– Андреев Стас – backend разработчик

На этапе аналитике каждый участник вносил вклад, вне зависимости от роли.

1.2 Обзор аналогов

Первым делом нам нужно было найти сервисы, на которых уже реализован подобный функционал. Выделить плюсы и минусы каждого сервиса и чем наш продукт будет отличаться будет отличаться.

По итогу мы проанализировали три веб-решения, которые от части закрывают сценарии нашего сервиса.

1.2.1 РИМЭКС

Популярный в автосервис, который имеет много точек по всей России. У автосервиса есть сайт, на котором можно записаться на услугу. Плюсы, которые нам удалось выделить у данного сервиса:

– Очень подробный выбор услуги. Нужно указать модель автомобиля и для каждой услуги своею информацию (вид шин, вид колодок и т. п.). Это дает подробную информацию для исполнителя заказа.

– Есть выбор автосервиса на карте

– Удобный и многофункциональный личный кабинет. Можно посмотреть настоящие и прошлые заказы и записи, сохранить данные о нескольких автомобилях.

Но у сервиса так же есть незначительные минусы:

– Обязательная регистрация. Пусть и имеется наличие быстрой авторизации (через Google или Яндекс), но этот аспект заставляет

пользователя тратить лишнее время, учитывая, что данные все равно нужно будет указать в заявке.

Сервис имеет очень функциональный сайт, узконаправленный на шиномонтаж и подобные услуги.

1.2.2 Aquagizer

Небольшая сеть автомоек, на сайте которой есть услуга записи. Как таковых плюсов, отличающих от других сервисов нам, не удалось отметить. Можно посмотреть на карте точку, но лишь в маленьком окне, людям, которые ориентируются в городе, будет тяжело найти точку. Так же можно записаться без регистрации, но и регистрации как таковой нет. Сайт больше похож на лэндинг, сделанный с помощью No Code инструментов.

Из минусов хочется выделить неудобный интерфейс. Было сложно найти на которой странице происходит запись.

1.2.3 Вебмойка

CRM система для владельцев автомоек. Довольно мощный инструмент, который закрывает практически все нужды партнера по учету сырья, ведению клиентской базы и так далее.

Как-то бесплатно попробовать возможности программы нет, поэтому суждения о продукте мы строили на основе предложений на сайте.

Из возможностей предоставляется: администрирование нескольких точек, ведение клиентской базы, учет товаров и расходников, отчеты и аналитика.

Из минусов можно отметить отсутствие какого-либо приложения или сайта для клиентов автомоек. Имеется возможность интегрировать систему оплаты на собственный сайт автомойки, но проработка вызывает вопросы и возможно может покрыть не все требования партнеров.

1.3 Функциональные требования

Проанализировав аналоги и посмотрев практики, которые используют существующие аналоги, мы выделили требования для нашего сервиса.

1.3.1 Требования к приложению для клиентов

– Регистрация и авторизация

– Функционал бронирования (услуга, дата, время, комментарий к брони)

– Личный кабинет: история бронирования со статусами заявок

Так же мы вынесли требования со вторичным приоритетом, которые будут рады видеть пользователи, но они будут необязательны для тестирования гипотезы:

– Отзывы

– Личный кабинет: редактирование профиля

– Регистрация и авторизация при помощи сторонних сервисов

1.3.2 Требования к приложению для партера

– Авторизация

– Инструмент добавления пользователей (новых партнеров)

– Просмотр броней по дате

– Подтверждение и отказ в бронировании

1.4 Создание базы данных

После установки требований к продукту были написаны модели данных

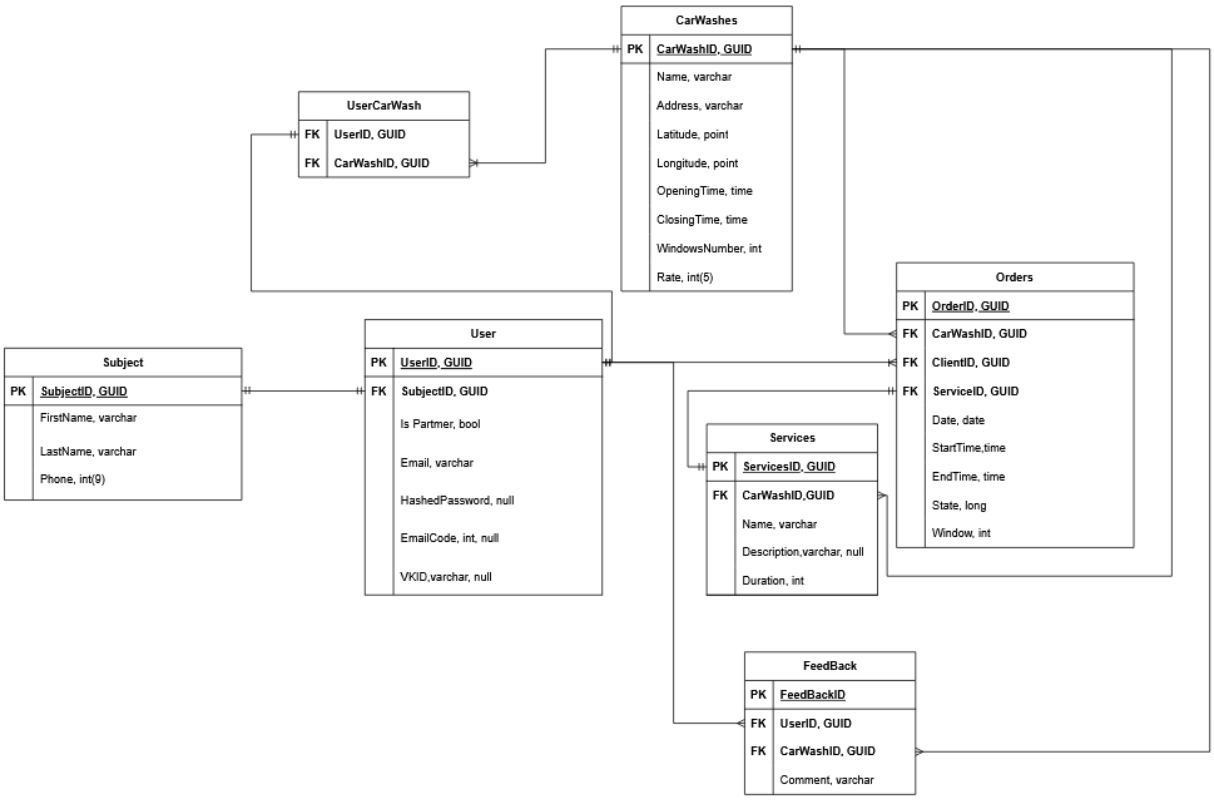

Рисунок 1 – Модель данных

Создали базу данных и развернули ее на хостинге, для удобного обращения во время разработки.

1.5 Пользовательские сценарии

Для документации поведения пользовательского интерфейса мы решили написать пользовательские сценарии (приложение А).

1.6 Поведение системы

Далее мы задокументировали поведение системы на более низком уровне. Для каждого сценария мы описали к каким элементам системы будет обращение и как оно будет происходить (приложение В).

1.7 Макеты приложений

После анализа аналогов и параллельно с документацией требований к системе мы работали над разработкой макетов приложения. В итоге мы сформировали начальную страницу, где у пользователя есть выбор: перейти на страницы с возможностью забронировать автомойку как клиент или зарегистрировать свою собственную автомойку как владелец. Соответственно страницы входа и регистрации, страницы выбора автомойки, услуг и даты, а также личный кабинет с возможностью изменения своих данных и просмотра истории бронирования. А в случае владельца – страница со списком опубликованных слотов для бронирования. (приложение С)

2 Разработка

2.1 Описание стека

Для разработки серверной части был выбран фреймворк FastAPI для Python. Это легковесный фреймворк, который позволяет в функциональном стиле быстро писать API для веб-приложений.

Для веб-приложения мы использовали React, как популярное решений, с которым мы уже имели опыт.

Для разработки мобильного приложения (PWA) применено HTML библиотеку Framework7 и написано на JavaScript.

Для развертывания сервиса был приобретен удаленный сервер. Развертывание будет происходить в Docker-контейнерах, что позволяет минимизировать настройку сервера и окружения разработки.

2.2 Разработка веб-приложения

В процессе разработки веб-приложения были созданы все заявленные страницы, включая интерфейсы как для клиентов, так и для владельцев. На начальном этапе выполнена верстка и настройка роутинга между страницами для обеспечения корректной навигации. После этого реализован базовый функционал, включающий выбор автомойки из списка, услуги, даты, а также ввод номера автомобиля.

Далее проведена доработка функциональных возможностей. Улучшен механизм выбора автомойки, услуги и даты: добавлен блок отображения ранее выбранных параметров для удобства пользователей. Затем выполнена интеграция с серверной частью. Реализованы запросы для получения списка доступных автомоек, услуг и свободных дат. Также разработаны методы поиска конкретной автомойки и услуги из общего списка, что позволило обеспечить более гибкую и удобную работу с системой.

2.3 Разработка мобильного приложения

В связи выбора модели проектирования мобильного приложения PWA (код пишется как веб-приложение, но по итогу получается исполняемый файл,

который является мобильным приложением) процесс разработки мало отличался от разработки веб-приложения.

На данный момент мы успели реализовать верстку основных станиц и подгрузку части данных из бэкенда.

2.4 Разработка API

Основным преимуществом FastAPI является легкость входа в технология и преимущества возможности функционального подхода в Python.

Итерации разработки API заключались в данных шагах:

– Установка нужных пакетов и библиотек

– Создание подключения к базе данных

– Создание модели данных в Python

– И финальная часть – это написание конечных точек, по которым клиенты могут обращаться к API.

2.5 Развертывание

Развертывание мы начали вначале разработки. Такое решение было принято из-за того что из-за разницы окружений могут возникать ошибки и некорректные отображения. Поэтому, чтобы успеть отладить приложения мы развернули их как можно скорее

Так же одной из причин является то что API должно быть доступно для разработчиков приложений всегда.

Как самый удобный и доступный инструмента авто-деплоя мы выбрали GitHub Actions.

Процесс авто-деплоя состоял из нескольких этапов:

– Написания конфигурационных файлов Dockerfile и docker-compose.yml

– Создания SSH соединения между сервером и репозиторием, путем создания приватных и публичных ключей и добавления их в секреты репозитория

– Написания конфигурации для деплоя на GitHub Actions. А именно описание всех действий, которые должны быть сделаны в тот момент, когда был произведен коммит в главную ветку репозитория.

– И как итог отладка и проверка корректного отображения интерфейса для веб приложение и тестирование конечных точек для API.

ЗАКЛЮЧЕНИЕ

На данный момент у нас реализована около 80% заявленных требований по проекту.

Готова вся верстка на мобильном и веб приложениях. Написано около половины обращений к API на получение, добавление и обновление данных.

Ведется работа над системой авторизации.

API предоставляет все нужные данные для реализации клиентских приложений. Ведется работа над конечными точками для партнерского функционала.

Для обоих проектов (API и веб-приложения) настроено автоматическое развертывание через репозитории на GitHub. И на данный момент ведется отладка работы веб приложения в итоговой продакшен среде.

Мы планируем в полной мере закончить все поставленные задачи к проекту до защиты.

ПРИЛОЖЕНИЕ A

Ссылка на описание пользовательских сценариев:

https://docs.google.com/spreadsheets/d/15pK_2JzD8dmp4_vnU-jmOwj_1S25jDjR/edit?usp=sharing&ouid=100321238079208952458&rtpof=true&sd=true

ПРИЛОЖЕНИЕ B

Описание поведения системы в виде UML-диаграмм: https://drive.google.com/drive/folders/1Xvm024ddyOp0gNbYqQ4BzIleqzagBE9m?usp=sharing

ПРИЛОЖЕНИЕ C

Макеты приложений под веб и мобильные устройства: https://www.figma.com/design/hO51kLKefZJpM68ynhfJnE/Untitled?node-id=0-1&t=JbujeQCfP1thOpmb-1

ПРИЛОЖЕНИЕ D

Ссылка на веб-приложение: http://45.153.188.106:5555/

ПРИЛОЖЕНИЕ E

Ссылка на API проекта: http://45.153.188.106:8000/docs#

ПРИЛОЖЕНИЕ F

Ссылка на репозиторий веб-приложения: https://github.com/hil1ch/carwash-booking

ПРИЛОЖЕНИЕ G

Ссылка на репозиторий API: https://github.com/flaup/carwash-api

ПРИЛОЖЕНИЕ H

Ссылка на репозиторий мобильного приложения: https://github.com/aibsg/CarWashes/tree/dev