

**Министерство науки высшего образования Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО**  
**ОБРАЗОВАНИЯ**  
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**  
**(Университет ИТМО)**

Факультет - Инфокоммуникационных Технологий

Образовательная программа - 09.03.02

Направление подготовки (специальность) - Мобильные сетевые технологии

**О Т Ч Е Т**

по курсовой работе

Тема задания: **Реализация системы для проведения городского ориентирования**

Обучающийся: Закоулов Илья Сергеевич, К3340

Руководитель: Говоров А. И.

Подписи членов комиссии:

\_\_\_\_\_  
(подпись)

\_\_\_\_\_  
(подпись)

\_\_\_\_\_  
(подпись)

Дата \_\_\_\_\_

Санкт-Петербург  
2020

# СОДЕРЖАНИЕ

|  |           |
|--|-----------|
| <b>СОДЕРЖАНИЕ</b>  | <b>2</b>  |
| <b>ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ</b>                  | <b>3</b>  |
| <b>ВВЕДЕНИЕ</b>  | <b>3</b>  |
| <b>Глава 1. ГОРОДСКОЕ ОРИЕНТИРОВАНИЕ “НЕИЗВЕСТНЫЙ ПИТЕР”</b> | <b>5</b>  |
| 1.1. Актуальность работы                                     | 5         |
| 1.2. Требование к создаваемой системе                        | 5         |
| <b>Глава 2. FRONTEND ПАНЕЛИ УПРАВЛЕНИЯ</b>                   | <b>7</b>  |
| 2.1. Основные сведения                                       | 7         |
| 2.2. Страницы панели управления                              | 7         |
| <b>Глава 3. BACKEND ПАНЕЛИ УПРАВЛЕНИЯ</b>                    | <b>11</b> |
| 3.1. Основные сведения                                       | 11        |
| 3.2. REST запросы  | 12        |
| <b>Глава 4. АВТОМАТИЗАЦИЯ РАЗВЕРТЫВАНИЯ ПРИЛОЖЕНИЯ</b>       | <b>13</b> |
| 4.1. Dockerfile  | 13        |
| 4.2. Docker-compose  | 13        |
| <b>ЗАКЛЮЧЕНИЕ</b>  | <b>15</b> |
| <b>СПИСОК ЛИТЕРАТУРЫ</b>                                     | <b>16</b> |
| <b>Приложение 1</b>  | <b>17</b> |

## ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ

**Квест** (англ. quest) – отдельно взятое соревнование по городскому ориентированию. Включает в себя задания, на которые необходимо дать ответ.

**Backend** – программно-аппаратная часть сервиса. Это то, что напрямую взаимодействует с базой данных, получает, обрабатывает и отдает информацию.

**Frontend** – клиентская сторона пользовательского интерфейса к backend'у. Обычно представляется в виде веб страниц.

## **ВВЕДЕНИЕ**

Городское ориентирование — одно из новых направлений активного отдыха. Такие мероприятия довольно сложно проводить для большого количества участников без специальной платформы, в которой можно создавать и редактировать квесты, задания и ответы к ним.

В связи с этим было выделено несколько задач: проанализировать текущие существующие решения, выявить их недостатки и реализовать собственную платформу для проведения мероприятий по городскому ориентированию.

# **1. ГОРОДСКОЕ ОРИЕНТИРОВАНИЕ “НЕИЗВЕСТНЫЙ ПИТЕР”**

## **1.1. Актуальность работы**

Городское ориентирование — это возможность по-другому взглянуть на Санкт-Петербург, узнать историю знакомых улиц, ведь чаще всего за обычными фасадами зданий скрыта истинная жизнь места: его прошлое, настоящее и будущее. Принято считать, что все самое красивое и интересное находится в центре, однако, можно доказать, что это неправда. Краеведы и городские активисты находят уникальные, необычные места в каждом районе и придумывают к ним загадки, а участники соревнуются на время в поисках заданных пунктов. Данное соревнование можно назвать квестом — набор заданий, на которые необходимо дать ответ.

Проанализировав несколько сервисов, которые позволяют проводить такие соревнования, было выявлено неудобство их использования и, в конечном счете, было принято решение реализовать собственную систему для проведения городского ориентирования.

## **1.2. Требование к создаваемой системе**

К создаваемой системе было вынесено несколько требований:

### **1. Возможность создания заданий к квесту**

Должна быть возможность создавать задания, которые должны включать в себя: само задание, варианты ответов, две опциональные подсказки и штрафное время за их использование.

## **2. Удобный выбор временных данных**

У большинства конкурентов это главный недостаток: дата или время вводилось вручную, что могло привести к ошибке. Должна быть реализована возможность выбора даты и времени из специальных компонент, в которых невозможно выбрать неверное значение.

## **3. Авторизация**

Пользователи без авторизации не должны иметь возможность сделать изменения на сайте.

## **4. Простой и удобный интерфейс**

Интерфейс должен быть очевиден администратору, который будет создавать задания.

## **2. FRONTEND ПАНЕЛИ УПРАВЛЕНИЯ**

### **2.1. Основные сведения**

Frontend разрабатывался в среде разработки PyCharm от JetBrains при помощи фреймворка Vuetify.js. Этот фреймворк включает в себя довольно большой набор CSS стилей и JS скриптов, которые представлены в виде Vue-компонент.

Стиль компонент по-умолчанию реализован по спецификации Material Design. Эта спецификация дизайна создана компанией Google и используется во всех ее сервисах. Начиная от главной поисковой страницы до ОС Android. Этот набор правил и спецификаций позволяет сгруппировать данные в удобном минималистичном формате. Управление такими данными должно быть интуитивно понятно.

В дизайне панели управления применялось два цвета: основной и акцентирующий. Основным цветом принято выделять все обычные активные кнопки или другие элементы управления страницей. Акцентирующий же цвет применяется для того, чтобы привлечь внимание пользователя.

### **2.2. Страницы панели управления**

Панель управления разделена на 4 основные страницы:

1. Страница авторизации. Стандартная страница для авторизации пользователя.

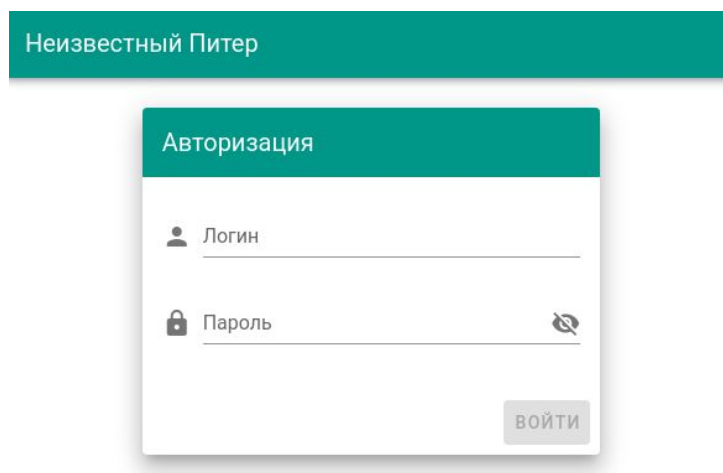


Рисунок 1 – Страница авторизации.

2. Список команд. Здесь можно посмотреть все созданные команды и квест, на который они зарегистрированы. Можно также создать новую команду и поменять пароль у старой команды.

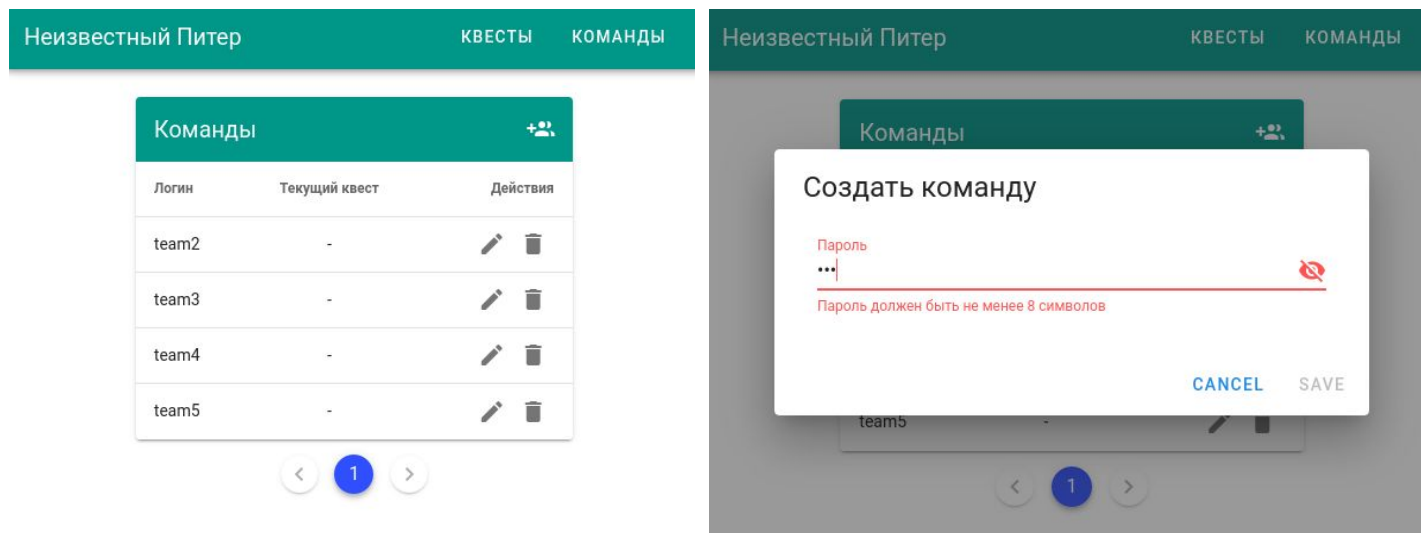


Рисунок 2 – Страница со списком команд (слева) и всплывающее окно для создания команды (справа).



3. Список квестов. На этой странице можно просмотреть все созданные задания, отредактировать их и добавить новые. Есть возможность посмотреть текущую статистику.

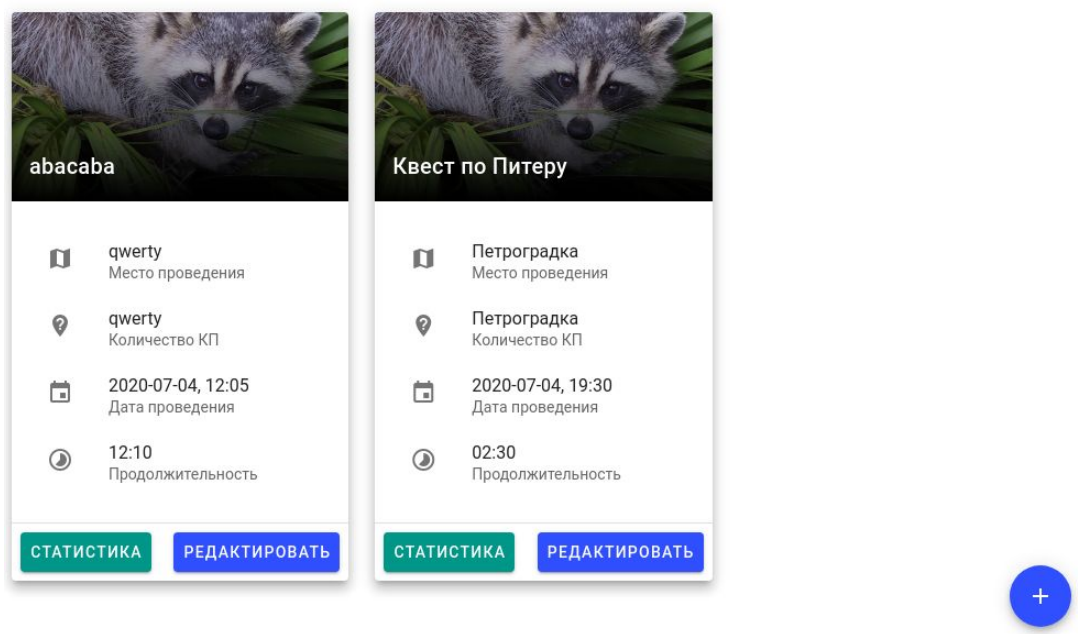



Рисунок 3 – Страница со списком квестов.

4. Редактор квеста. На этой странице можно указать название квеста, место проведения, дату и время старта, продолжительность, приветственный и прощальные тексты, а также штрафные времена для двух подсказок. Страница создания квеста аналогична странице редактирования.



ОПИСАНИЕ

ЗАДАНИЯ

Название

Т Квест по Питеру

Место проведения

Петроградка

Время старта

2020-07-04

Время начала

19:30

Предолжительность

02:30

Приветственный текст

Здравствуй друг! Тебя ждет самый лучший квест в твоей жизни!

Прощальный текст

Встречаемся на Вяземском переулке 5/7 для вручения подарков!

Штрафное время за подсказку №1


00:05

Штрафное время за подсказку №1

00:15

УДАЛИТЬ

СОХРАНИТЬ



ОПИСАНИЕ

ЗАДАНИЯ

1. Реши загадку, получи закладку

Название

Т Реши загадку, получи закладку

Вопрос

Двое ребят копаются в клумбе. Кто это?

Уборщики

Работники ЖЭК

Подсказка №1

Подсказка №2

УДАЛИТЬ ЗАДАНИЕ

+ ДОБАВИТЬ

УДАЛИТЬ

СОХРАНИТЬ

Рисунок 4 – Карточка создания квеста с редактированием описания (слева) и редактированием заданий (справа).

## 3. BACKEND ПАНЕЛИ УПРАВЛЕНИЯ

### 3.1. Основные сведения

Backend разрабатывался в интегрированной среде разработки PyCharm от JetBrains. В качестве веб-фреймворка был выбран Django.

В качестве базы данных использовался PostgreSQL база данных, взаимодействие с которой происходит через стандартную ORM Django.

В базе данных используется 6 сущностей: Квест (Quest), Задание (Task), Ответ (Answer), Статистика задания (TaskStatistic), Статистика команды (TeamStatistic) и Пользователь (User).

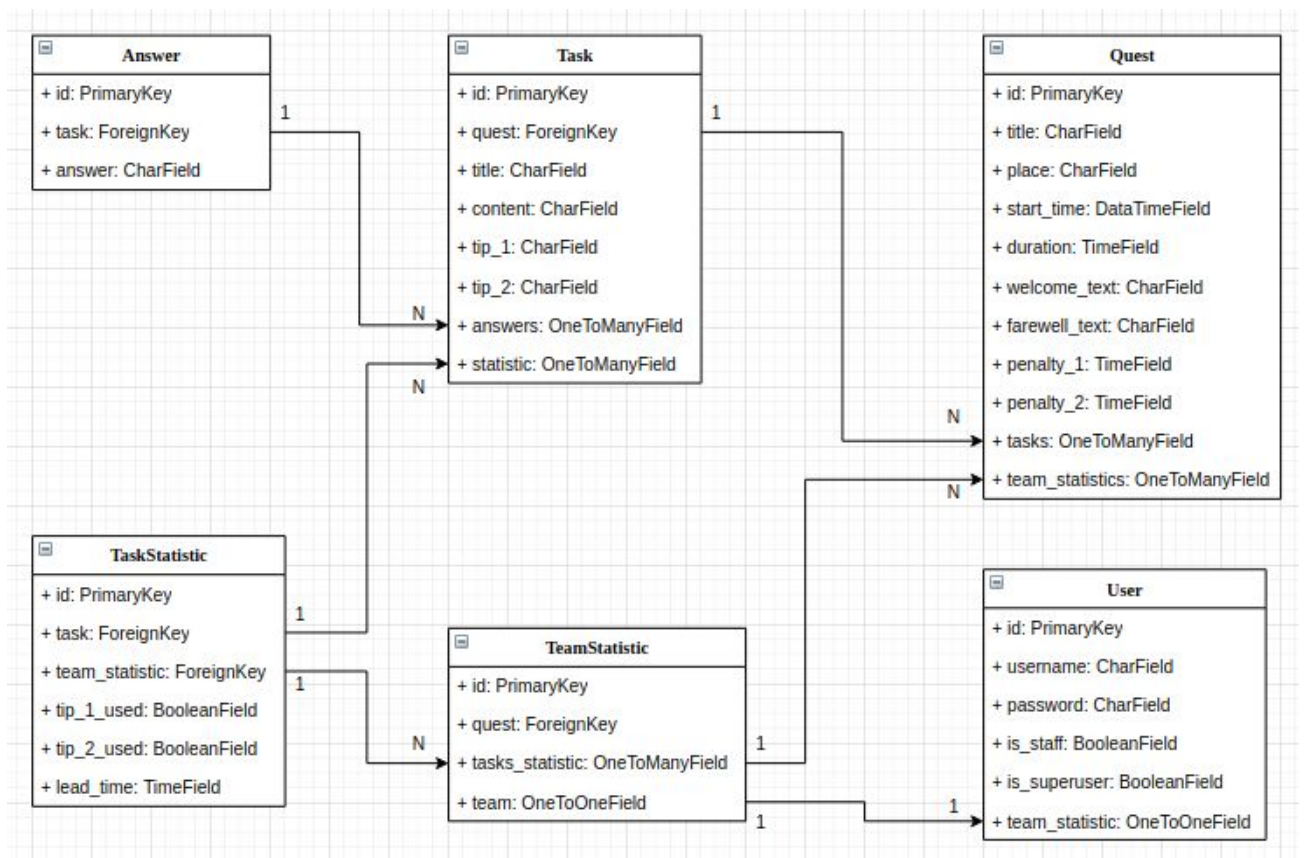


Рисунок 5 – Диаграмма сущность-связь для базы данных.

Серверная часть для админ панели обрабатывает входящие запросы, сохраняет изменения в базе данных и возвращает данные в JSON формате.

### 3.2. REST запросы

Реализовано большое множество REST запросов. Они позволяют мобильному приложению получать данные в формате JSON, и производить изменения в базе данных. Основные REST запросы и их назначение перечислены в таблице 1.

*Таблица 1*

| REST-запрос                 | Описание  | Передаваемые параметры | Возвращаемые параметры               |
|-----------------------------|---|------------------------|--------------------------------------|
| /api/auth                   | JWT-авторизация                                 | login, password        | access, refresh - токены авторизации |
| /api/users/teams            | Создание команды, список команд                 | password               | login, список команд                 |
| /api/users/teams/<id>/      | Редактирование, удаление команды                | password               | -                                    |
| /api/users/questMakers      | Создание администратора, список администраторов | login, password        | login, список администраторов        |
| /api/quests                 | Создание квеста, список квестов                 | Полное описание квеста | Квест, список квестов                |
| /api/quests/<id>/statistics | Статистика квеста                               | -                      | Статистика квеста                    |
| /api/quests/<id>/join       | Принять участие в квесте                        | -                      | -                                    |

## 3. АВТОМАТИЗАЦИЯ РАЗВЕРТЫВАНИЯ ПРИЛОЖЕНИЯ

### 3.1. Dockerfile

Dockerfile – обычный текстовый файл, содержащий инструкции для создания виртуального контейнера. Такой файл, обычно, содержит набор команд для установки необходимых зависимостей и копирования исходных кодов. В рамках реализации приложения было написано два dockerfile: для Frontend и Backend приложения.

### 3.2. Docker-compose

Docker-compose – это Docker утилита, позволяющая создавать, запускать и уничтожать контейнеры. Она настраивается при помощи docker-compose.yaml файла, который содержит информацию о контейнерах, которые необходимо создать, и их сетевой конфигурации. Docker позволяет создавать контейнеры полностью независимыми от системы, что и требует их дополнительной настройки в виде конфигурации сети. Некоторые контейнеры можно объединить между собой, чтобы они могли передавать данные между собой.

Для успешной автоматизации сборки было создано 4 контейнера:

1. Database
2. Frontend
3. Backend
4. Nginx

Причем Database был объединен в одну сеть с Backend, чтобы была возможность обращаться к Postgres из Django. А также Frontend, Backend и

Nginx были объединены в другую сеть, чтобы nginx видел Frontend и Backend приложения.

Также была реализована возможность развертывать веб приложение с различными окружениями, например develop, local и production. Их главное отличие в конфигурации базы данных, debug-режима на Frontend и Backend, SSL сертификатах и порте, на котором открывается доступ к приложению.

## **ЗАКЛЮЧЕНИЕ**

В рамках выполнения задачи по реализации системы проведения городского ориентирования было выявлено, что данная система имеет широкий потенциал для вовлечения населения в активный отдых. Туристы и обычные горожане, приняв участие в таком квесте могут посмотреть на город с другой стороны, узнав много нового.

В будущем можно добавить возможность проходить квест на велосипеде. Рекорды прохождения заданий для таких команд будут учитываться отдельно от пешеходных команд.

## СПИСОК ЛИТЕРАТУРЫ

1. Python Software Foundation: Python 3.7.0 documentation. URL: <https://docs.python.org/3.7/>.
2. Encode: Django REST framework. URL: <https://www.django-rest-framework.org/>.
3. Evan You: Vue CLI. URL: <https://cli.vuejs.org/>.
4. Vuerify, LLC: Vuetify. URL: <https://vuetifyjs.com/en/getting-started/quick-start/>.
5. Docker, Inc: Docker. URL: <https://www.docker.com/>.
6. Nginx, Inc: Nginx. URL: <https://nginx.org/>.



## Приложение 1

С исходным кодом панели управления можно ознакомиться по ссылке: <https://github.com/SprutSDM/cityOrientWeb>.