**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**на** **создание**

**мобильного приложения QUADROPLE**

**СПИСОК СОКРАЩЕНИЙ**

API - программный интерфейс для интеграции с внутренними и внешними системами

Backend - основная программно-аппаратная часть системы

GUI - графический пользовательский интерфейс — визуальное представление системы

REST API сервис - веб-сервис, который предоставляет доступ к программному интерфейсу системы по протоколу HTTPS

Администратор - пользователь с правами объект-менеджера, картографа, модератора и возможностью редактировать роли и функционал других пользователей

База данных системы - внутренняя база данных системы

Интерфейс администратора - веб-представление административной панели

Интерфейс служебного пользователя - представление административной панели, ограниченного правами пользователя, функционала.

Пользователь - человек, взаимодействующий с системой в качестве потребителя услуг и информации

1. ЗАДАНИЕ НА РАЗРАБОТКУ

Создание приложения осуществляется путем разработки серверной и клиентской части, а также API для их взаимодействия.

Технологический стек:

На сервере используются:

* операционная система Ubuntu 20.04;
* СУБД Postgres 13.2;
* стабильный релиз Python 3.9;
* программно-аппаратный элемент Firewall на iptables и fail2ban.

ОБЩИЕ ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

Задачи приложения:

* представить возможность познакомиться с добавленными городами туристам;
* дать возможность жителям добавленных городов лучше узнать его, проходя по созданным маршрутам.

Приложение должно:

* предлагать пользователю эксклюзивные маршруты для прогулок по городу;
* предлагать возможность пользователям создавать свои собственные маршруты и оценивать маршруты друг-друга;
* заинтересовать пользователя системой фото-отметок в точках маршрута;

1. ОПИСАНИЕ СИСТЕМЫ
   1. Пользовательские роли

Система должна поддерживать следующую ролевую модель, а также возможность управления ролями и их правами.

Основные ролевые группы и их состав:

3.1.1. Пользователи приложения:

1. Незарегистрированный пользователь;
2. Зарегистрированный пользователь.

3.1.2. Пользователи админки:

1. Администратор системы.
   * 1. Пользователи приложения

Эта ролевая группа включает в себя пользователей приложения, обладающих первичным ограниченным доступом.

1. **Незарегистрированный Пользователь** – это пользователь, который не авторизовался в системе.

Пользователь получает актуальную запрашиваемую информацию в режиме реального времени.

Пользователь может:

* просматривать карту и маршруты на ней.

1. **Зарегистрированный пользователь** – это пользователь, который имеет учетную запись и успешно произвёл авторизацию в системе.

Зарегистрированный пользователь наследует права и возможности незарегистрированного.

Зарегистрированный пользователь может:

* проходить существующие маршруты;
* оставлять в качестве метки фотографии на маршрутах;
* создавать собственные маршруты;
* оценивать маршруты.
  + 1. Администратор

Администратор может:

* изменять и удалять данные пользователей;
* изменять и удалять маршруты и метки на них;
* строить новые маршруты на карте.
  1. Компоненты системы

Предварительно считается, что система будет состоять из следующих компонентов:

* 1. Backend

Backend – это та серверная часть системы, внутри которой происходит вся бизнес логика и процессы.

* 1. REST API Сервис

Взаимосвязь и отношения с Backend происходят посредством обращения клиентских приложений к REST API сервису. Интеграция со сторонними системами так же производится посредствам API сервиса.

1. Пользователи

Регистрация и авторизация пользователя осуществляется напрямую и через API ВКонтакте.

Личный кабинет пользователя содержит следующие поля:

* ID - Числовой тип
* Логин - Строковый тип
* Фамилия - Строковый тип
* Имя - Строковый тип
* Выбранный город - Строковый тип
* Чекбокс для согласия с правилами пользования и обработкой персональных данных.

Интерфейс управления пользователями для администраторов должен позволять:

* Просматривать список зарегистрированных пользователей;
* Осуществлять поиск и фильтрацию среди пользователей по фамилии и имени, логину;
* Позволять редактировать информацию о пользователе как в индивидуальном, так и массовом режиме;
* Иметь механизмы автоматизированной выгрузки участников определенных маршрутов формате (Excel или CSV).

1. **Маршруты**

Приложение позволяет пользователю создать и пройти определённый маршрут.

Пользователь может выбрать точки своего маршрута, указав их в интерактивном режиме на карте, добавить для каждой точки описание.

Функционал маршрутов должен позволять:

* Осуществлять поиск маршрутов с применением поиска/фильтрации по рейтингу;
* Просматривать список пройденных и созданных пользователем маршрутов;

1. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К ПРИЛОЖЕНИЮ
2. 1. Управление личными данными пользователей

**Система управления пользователями должна:**

* позволять заполнять и редактировать личные данные;
* позволять пользователю изменять принадлежащие ему маршруты.

4.2. Система управления маршрутами

**Система управления маршрутами должна:**

* позволять пользователям создавать и редактировать маршруты;
* позволять пользователям оставлять комментарии и оценки маршрутам;
* позволять администраторам изменять и удалять маршруты;
* позволять администраторам изменять и удалять комментарии.

1. Интерфейсы

Приложение

В данном разделе описаны основные требования к приложению.

*Общие требования к приложению*

Приложение должно отвечать общим функциональным требованиям, представленным в разделе «Общие функциональные требования».

*Административная часть*

Доступ к административной части системы осуществляется посредством авторизации пользователя с соответствующей ролью.

Пользователь административной части системы должен иметь возможность управлять настройкой и/или производить администрирование системы без дополнительных технических навыков посредством GUI приложения.

Административная часть системы должна быть простой и удобной в использовании с сопровождением подсказок и справок, составленных разработчиком ПО.

**В административной части имеются отдельные разделы для управления:**

* Маршрутами.
* Пользователями, группами пользователей и правами доступа.
* Контентом.
* Сбором данных (отчёты и статистика).
* Просмотром статистики (составление отчётов и графиков).

1. Сбор персональных данных

Система должна собирать персональные пользовательские данные с разрешения пользователя, такие как:

* поисковые запросы;
* частота использования приложения;
* сбор геопозиционных данных при их доступности.

1. ТЕХНИЧЕСКИЕ ТРЕБОВАНИЯ
   1. Требования к платформе

**Backend** системы должен:

* быть написан на языке последнего стабильного релиза, оптимизированном под большие нагрузки;
* быть написан по всем канонам современной разработки с использованием современных библиотек, с приоритетом на «открытый код»;
* быть компонентным, расширяемым.
  1. Требования к GUI мобильного приложения

GUI приложения должен учитывать рекомендации к проектированию интерфейса от производителя операционной системы.

GUI должен отвечать требованиям фирменного брендбука и должен быть согласован с заказчиком.

* 1. Требования к API приложения

Приложение подразумевает под собой набор конечных точек API, с помощью которых возможно использование её функционала. Работа должна быть устроена так, чтобы была возможность добавлять новые интерфейсы (клиентской части) без вмешательств в программный код.

**API приложения должен**:

* отвечать всем канонам современный разработки (HTTPS, WebSockets, RESTful, oAuth и т.д.);
* быть максимально полным и покрывать весь функционал пользовательской и административной части. Работа всех клиентских частей должна производиться посредством использования этого API;
* сопровождаться всеобъемлющей документацией с примерами кода, в закрытом доступе;
* предоставлять систему авторизации сторонних приложений (например, token). Выдача доступа осуществляется администрацией.
  1. Требования к информационной безопасности
* Используемая платформа и стороннее ПО должно соответствовать стандарту информационной безопасности ISO/IEC 27001 и гарантировать безопасность общих и пользовательских данных.
* Уровень защиты персональных данных должен отвечать требованиям ФЗ-№152.

1. ВНЕШНИЕ СЕРВИСЫ

Система должна поддерживать внешние сервисы, такие как:

* сервис почтовых рассылок;
* провайдер авторизации VK.com.