Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ВТ
_____ О.В. Непомнящий
подпись инициалы, фамилия
« ___ » _____ 2020 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Веб-приложение для просмотра сведений о гражданских аэропортах
тема

Руководитель	_____	доцент, к.т.н.	С.Н. Титовский
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		Д.И. Моргачев
	подпись, дата		инициалы, фамилия
Нормоконтролер	_____	доцент, к.т.н.	С.Н. Титовский
	подпись, дата	должность, ученая степень	инициалы, фамилия

Красноярск 2020

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ВТ
_____ О.В. Непомнящий
подпись инициалы, фамилия
« ___ » _____ 2020 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работ

Студенту Моргачеву Дмитрию Игоревичу
фамилия, имя, отчество

Группа 16-08Б Направление (специальность) 09.03.01
номер код

Информатика и вычислительная техника
наименование

Тема выпускной квалификационной работы Веб-приложение для просмотра сведений о гражданских аэропортах

Утверждена приказом по университету № 6623/с от 26 мая 2020 г.

Руководитель ВКР С.Н. Титовский, доцент кафедры «Информатика и вычислительная техника» ИКИТ СФУ, кандидат технических наук
инициалы, фамилия, должность, ученое звание и место работы

Исходные данные для ВКР Анализ существующих аналогов, учебные пособия

Перечень разделов ВКР Сравнительный анализ, используемые технологии, разработка веб-приложения

Перечень графического материала Презентация в формате Power Point

Руководитель ВКР _____ С.Н. Титовский
подпись инициалы, фамилия

Задание принял к исполнению _____ Д.И. Моргачев
подпись инициалы, фамилия

« ___ » _____ 2020

РЕФЕРАТ

Выпускная квалификационная работа по теме «Веб-приложение для просмотра сведений о гражданских аэропортах» содержит 40 страниц текстового документа, 26 иллюстраций, 3 таблицы и 8 использованных источников.

АЭРОПОРТЫ, ВЕБ-ПРИЛОЖЕНИЕ, PYTHON, DJANGO, POSTGRESQL, MVT, DOCKER, JAVASCRIPT.

Целью работы является проектирование и разработка веб-приложения для просмотра сведений о гражданских аэропортах.

Основные задачи:

1. провести анализ предметной области и существующих решений;
2. определить технологический стек для разработки;
3. спроектировать архитектуру разрабатываемого веб-приложения;
4. спроектировать базу данных;
5. спроектировать серверную часть веб-приложения;
6. разработать бизнес-логику;
7. разработать клиентскую часть приложения.

Результатом работы является разработанное указанное выше веб-приложение с подробным описанием структуры и алгоритмов работы.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной выпускной квалификационной работы (ВКР) является создание программного продукта, осуществляющего поиск по базе данных аэропортов всего мира.

Актуальность разработки данного программного продукта обусловлена тем, что большинство схожих функционально веб-приложений (ВП) платны или недостаточно функциональны, в работе будет проведен сравнительный анализ существующих приложений.

Для достижения поставленной цели необходимо выполнить следующие задачи:

- провести анализ предметной области и существующих решений;
- определить технологический стек для разработки;
- спроектировать архитектуру разрабатываемого веб-приложения;
- спроектировать базу данных;
- спроектировать серверную часть веб-приложения;
- разработать бизнес-логику;
- разработать клиентскую часть приложения.

1 Анализ предметной области

1.1 Сравнительный анализ

1.1.1 World-airport-codes.com

Сайт имеет современный интуитивный не перегруженный интерфейс. Также есть в верхнее меню, в котором пользователь может ознакомиться с лучшими аэропортами мира.

Присутствует форма поиска, есть возможность оставлять отзывы об аэропортах, также есть отображение местонахождения аэропортов на карте. Информация о текущей погоде в выбранном аэропорту доступна только после оформления платной подписки. Нет загрузки фотографий аэропорта и сайт представлен только на английском языке. Главная страница сайта представлена на рисунке 1.

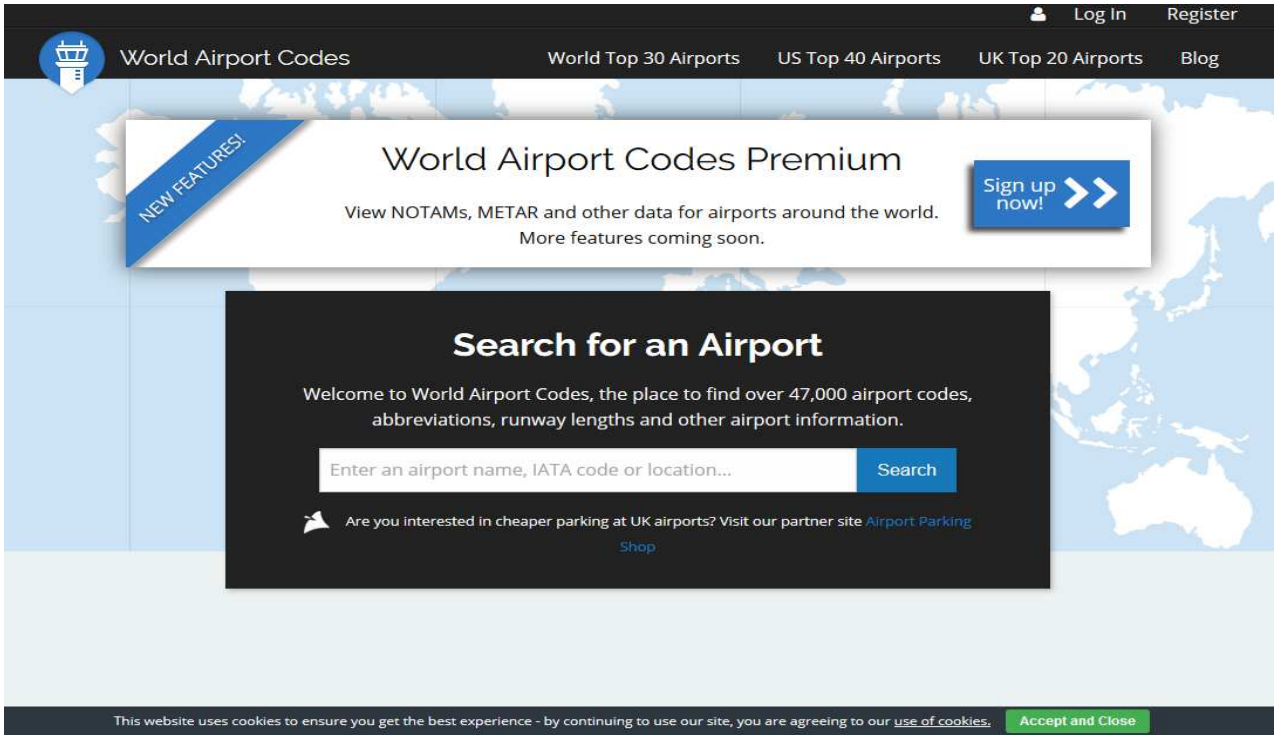


Рисунок 1 – Главная страница сайта world-airport-codes.com

1.1.2 Skyvector.com

Сайт Skyvector.com имеет слегка устаревший, но вполне понятный интерфейс.

Главная страница – это карта с отметками аэропортов. На сайте есть возможность оставлять отзывы об аэропортах, также есть отображение местонахождения аэропортов на карте, есть возможность просмотреть фотографии, отсутствует информация о текущей погоде в аэропорту. Главная страница сайта представлена на рисунке 2.

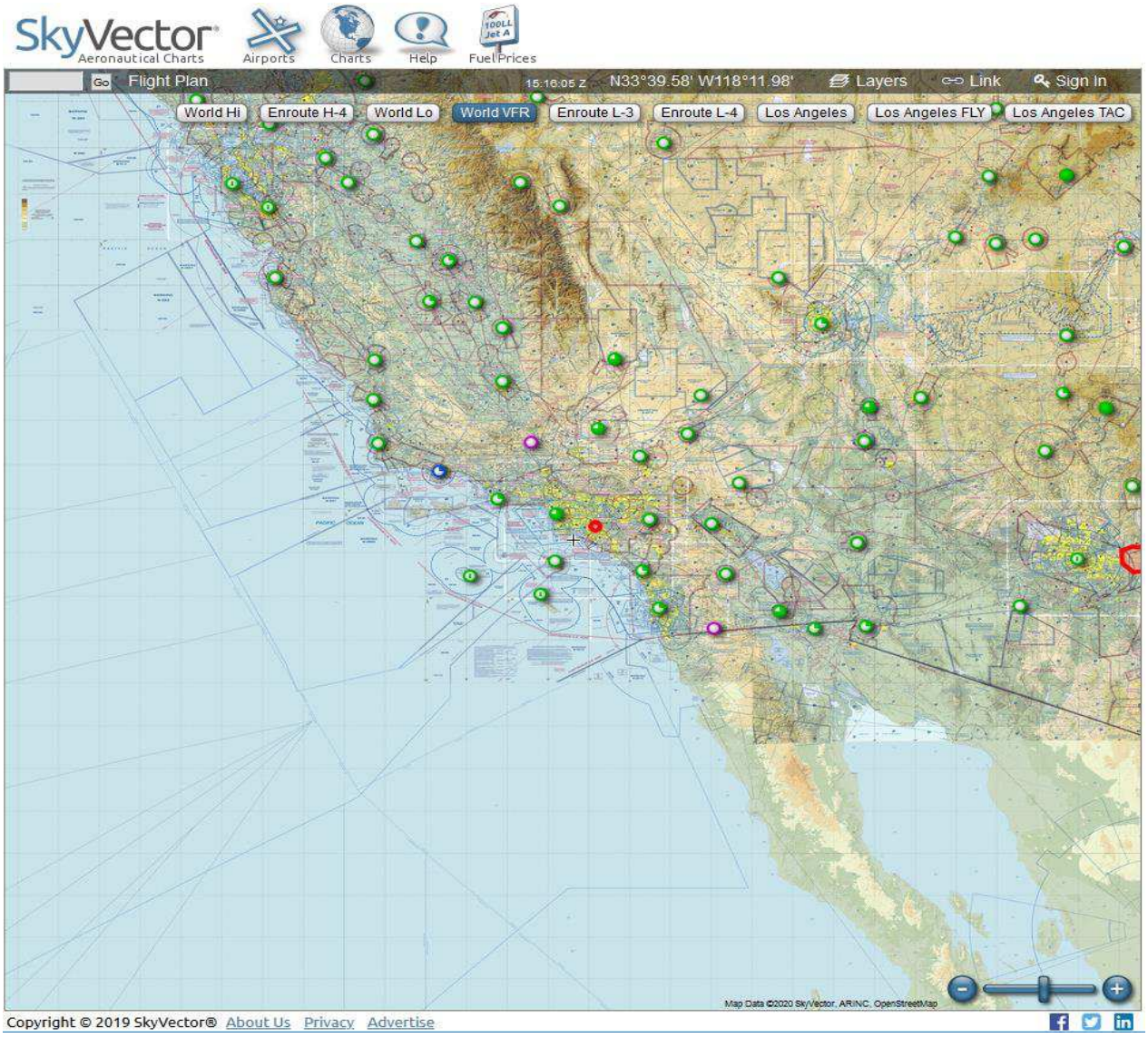


Рисунок 2 – Главная страница сайта skyvector.com

1.1.3 Apinfo.ru

Сайт имеет неудобный и устаревший интерфейс. Серьезным недостатком является отсутствие общего поиска по основным критериям, поиск возможен только по конкретному параметру: по коду IATA, по коду ICAO, по названию, по стране, по городу. Главная страница содержит новостную ленту с соответствующей информации.

На сайте есть возможность оставлять отзывы об аэропортах, также есть отображение местонахождения аэропортов на карте. Нет возможности просматривать информацию о текущей погоде в аэропорту, и нет загрузки фотографий аэропорта. Есть возможность экспорта информации о аэропортах в CSV, XLS или XML форматах. Главная страница представлена на рисунке 3.

Рисунок 3 – Главная страница сайта Apinfo.ru

1.2 Результаты сравнительного анализа

В таблице 1 приведены результаты сравнения четырех веб-сервисов для поиска технической и общей информации об аэропортах мира по итогам их обзорного исследования. Критериями сравнения являются наличие у веб-сервиса функций, наличие которых требуется для удобного взаимодействия с сервисом и быстрого доступа к нужной информации.

Таблица 1 – Сравнительная таблица сервисов-аналогов

Критерий сравнения	World-airport-codes.com	Skyvector.com	Apinfo.ru
Поиск аэропортов по базе данных	реализована	реализована	реализована
Возможность оставлять отзывы об аэропортах	реализована	реализована	реализована
Местоположение аэропорта	реализована	реализована	реализована
Текущая погода в аэропорту	реализована (платно)	не реализована	не реализована
Загрузка фотографий	не реализована	реализована	не реализована
Интерфейс	Современный, интуитивно-понятный	Устаревший, но в целом понятный	Неудобный поиск, устаревший
Наличие русского языка	Нет	Нет	Да

На всех рассмотренных веб-сервисах реализован основной функционал - поиск по базе данных аэропортов и предоставление найденной информации пользователю, но все сайты варьируются в удобстве и качестве представляемой информации.

Из всех рассмотренных сервисов-аналогов наиболее полноценным является сервис world-airport-codes.com, однако некоторый функционал сайта является платным и отсутствует поддержка русского языка.

2 Используемые технологии

2.1 Python

Python – это универсальный современный язык программирования (ЯП) высокого уровня, к преимуществам которого относят высокую производительность программных решений и структурированный, хорошо читаемый код. Широкий перечень встроенных библиотек позволяет применять внушительный набор полезных функций и возможностей. ЯП может использоваться для написания прикладных приложений, а также разработки веб-приложений [1].

К основным особенностям можно отнести строгую динамическую типизацию и автоматическое управление памятью. Немаловажно, что Python имеет одно из самых развитых сообществ и огромное количество информации в открытых источниках, поэтому не составит большого труда найти ответ на любой возникший вопрос.

Для Python существует огромное количество сервисов, сред разработки и фреймворков, в том числе для написания веб-приложений – это позволяет легко найти подходящий продукт для выполнения поставленной задачи.

Исходя из преимуществ и особенностей, Python полностью подходит для решения поставленной задачи и выбран в качестве основного для разработки веб-приложения в рамках данной ВКР.

2.2 Django

Django является одним из самых популярных фреймворков среди питон-программистов, что неудивительно, поскольку фреймворк позволяет очень быстро создавать безопасные и поддерживаемые веб-сайты. Фреймворк заботится о многих аспектах веб-разработки, поэтому разработчик может сосредоточиться на написании своего приложения.

Django включает в себя концепцию Model-View-Template (MVT),

позволяющую разделить общую архитектуру на отдельные части. При этом управляющая логика разделена на три отдельных компонента так, что модификация одного из них оказывает минимальное воздействие на другие части. К таким компонентам относят разделяемые данные, логику и слои визуализации. В общем случае такая концепция позволяет разделить разработку информационного наполнения на уровне базы данных и разработку web-страниц [2].

Некоторые возможности и особенности Django:

– архитектура MVT;

– включает в себя легковесный и автономный веб-сервер для разработки и тестирования;

– мощный шаблонизатор, основанный на специальных тегах, с возможностью наследования шаблонов;

– динамическая административная панель;

– по умолчанию обеспечивает защиту от многих уязвимостей, включая SQL-инъекцию, межсайтовый скриптинг, подделку межсайтовых запросов и кликджекинг;

– унифицированные средства для работы с базами данных любых поддерживаемых форматов: SQLite, MySQL, PostgreSQL, Firebird и др.

2.3 PostgreSQL

PostgreSQL — это объектно-реляционная система управления базами данных (ОРСУБД), основанная на POSTGRES, Version 4.2 — программе, разработанной на факультете компьютерных наук Калифорнийского университета в Беркли. В POSTGRES появилось множество новшеств, которые были реализованы в некоторых коммерческих СУБД гораздо позднее [3].

Фундаментальная характеристика объектно-реляционной базы данных — это поддержка пользовательских объектов и их поведения, включая типы данных, функции, операции, домены и индексы. Это делает PostgreSQL

невероятно гибким и надежным. Среди прочего, он умеет создавать, хранить и извлекать сложные структуры данных.

Существует обширный список типов данных, которые поддерживает PostgreSQL. Кроме числовых, с плавающей точкой, текстовых, булевых и других ожидаемых типов данных (а также множества их вариаций), PostgreSQL может похвастаться поддержкой uuid, денежного, перечисляемого, геометрического, бинарного типов, сетевых адресов, битовых строк, текстового поиска, xml, json, массивов, композитных типов и диапазонов, а также некоторых внутренних типов для идентификации объектов и местоположения логов. Справедливости ради стоит сказать, что MySQL, MariaDB и Firebird тоже имеют некоторые из этих типов данных, но только PostgreSQL поддерживает их все.

Благодаря свободной лицензии, PostgreSQL разрешается бесплатно использовать, изменять и распространять всем и для любых целей — личных, коммерческих или учебных.

2.4 HTML5 и CSS

Для подготовки гипертекстовых документов в данной работе используется HTML5 (Hyper Text Markup Language), предоставляющий широкие возможности по форматированию и структурной разметке документов, организации связей между различными документами, средства включения графической и мультимедийной информации.

В пятой версии HTML разработчики постарались объединить все инструменты, необходимые для создания профессиональных, современных и динамичных сайтов, использующих наиболее распространенные технологии и соответствующих современным стандартам.

CSS (Cascading Style Sheets) означает каскадные таблицы стилей и представляет собой формальный язык описания внешнего вида документа, написанного с использованием языка разметки. Таким образом, CSS ничего не представляем сам по себе, если не связан с HTML-документом.

Назначение CSS – отделять то, что задает внешний вид страницы, от ее содержания. Если документ создан только с использованием HTML, то в нем определяется не только каждый элемент, но и способ его отображения (цвет, шрифт, положение блока и т.д.). Если же подключены каскадные таблицы стилей, то HTML описывает только очередность объектов. А за все их свойства отвечает CSS. В HTML достаточно прописывать класс, не перечисляя все стили каждый раз.

Следует отметить, что Django позволяет динамически генерировать HTML и предоставляет бэкенд для собственной системы шаблонов, которая называется - язык шаблонов Django (Django template language, DTL). Самый распространенный подход - использование шаблонов. Шаблоны содержат статический HTML и динамические данные, рендеринг которых описан специальным синтаксисом [4].

2.5 JavaScript

JavaScript – это инструмент, предназначенный для придания динамичности HTML-страницам. Некоторые из тех задач, которые можно решать с помощью JavaScript:

– добавление взаимодействий с действиями пользователя на веб-страницах;

– добавление логики – веб-страница будет вести по-особому, в зависимости от того, как ведет себя пользователь;

– математические вычисления на страницах;

– добавление каких-то эффектов на веб-страницы, которые требуют вычислительных операций. Движущиеся элементы, всплывание и скрытие элементов и т.д.

В работе JavaScript используется для реализации динамической поисковой выдачи по базе аэропортов и для создания интерактивных элементов управления.

2.6 Docker

Docker – это программная платформа для быстрой разработки, тестирования и развертывания приложений. Docker упаковывает ПО в стандартизованные блоки, которые называются контейнерами. Каждый контейнер включает все необходимое для работы приложения: библиотеки, системные инструменты, код и среду исполнения. Благодаря Docker можно быстро развертывать и масштабировать приложения в любой среде и сохранять уверенность в том, что код будет работать [5].

Преимущества использования:

– ускоренный процесс разработки, так как нет необходимости устанавливать вспомогательные инструменты вроде PostgreSQL, Redis, Celery: их можно запускать в контейнерах;

– минимальное потребление ресурсов;

– скоростное развертывание;

– простое масштабирование;

– удобная инкапсуляция приложений.

Для управления несколькими контейнерами, из которых состоит проект, используют пакетный менеджер – Docker Compose. Он применяется при проектировании сложных программных продуктов, включающих в себя множество процессов и сервисов.

В данной ВКР Docker Compose будет использоваться для запуска двух контейнеров: самого веб-приложения и базы данных PostgreSQL.

2.7 PyCharm

Последнее, на что стоит обратить внимание, при разработке сайта в данной работе использовалась среда разработки (IDE) для языка Python – PyCharm.

PyCharm – это кросс-платформенная интерактивная IDE для языка программирования Python, обеспечивающая простоту использования.

PyCharm содержит следующий функционал:

– редактор с подсветкой синтаксиса Python;

– динамическая интроспекция кода — автодополнение, переход к определению объекта по клику мыши;

– нахождение ошибок;

– поддержка одновременного использования множества консолей Python;

– просмотр и редактирование переменных с помощью GUI;

– встроенные средства доступа к документации;

– гибко настраиваемый интерфейс;

– встроенный графический отладчик.

3 Разработка веб-приложения

3.1 Докеризация приложения

Для реализации среды была выбрана мультиконтейнерная сборка на основе Docker Compose. Для этого необходимо объявить два сервиса: server, внутри которого будет находиться само веб-приложение, и db, для запуска PostgreSQL. Конфигурация Docker Compose представлена на рисунке 4.

```yaml
version: '3'
services:
  server:
    build:
      context: ./
      dockerfile: ./server/Dockerfile
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - ./server:/server
    ports:
      - "8000:8000"
    depends_on:
      - db
    environment:
      DEBUG: 'True'
      DATABASE_URL: 'postgres://postgres:@db:5432/postgres'

  db:
    image: postgres:11.2
    environment:
      POSTGRES_DB: postgres
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: postgres
    ports:
      - 3406:5432
```

Рисунок 4 – Конфигурация docker-compose.yml

Запуск контейнеров производится через команду docker-compose up -d.

3.2 Исходные данные

Для наполнения веб-приложения содержимым, в сети интернет была найдена база данных аэропортов, состоящая из двух таблиц формата xlsx (Microsoft Excel), где каждая колонка представляет техническую или информационную характеристику аэропорта. Описание колонок представлено в таблицах 2 и 3 соответственно.

Таблица 2 – Описание колонок первого файла

Колонка	Значение колонки
id	Идентификатор аэропорта
name	Название аэропорта
latitude_deg	Широта
longitude_deg	Долгота
elevation_ft	Высота над уровнем моря
type	Тип аэропорта
municipality	Муниципалитет
iata_code	Уникальный идентификатор аэропорта (ИАТА)
home_link	Ссылка на веб-сайт аэропорта
ident	ИКАО идентификатор аэропорта

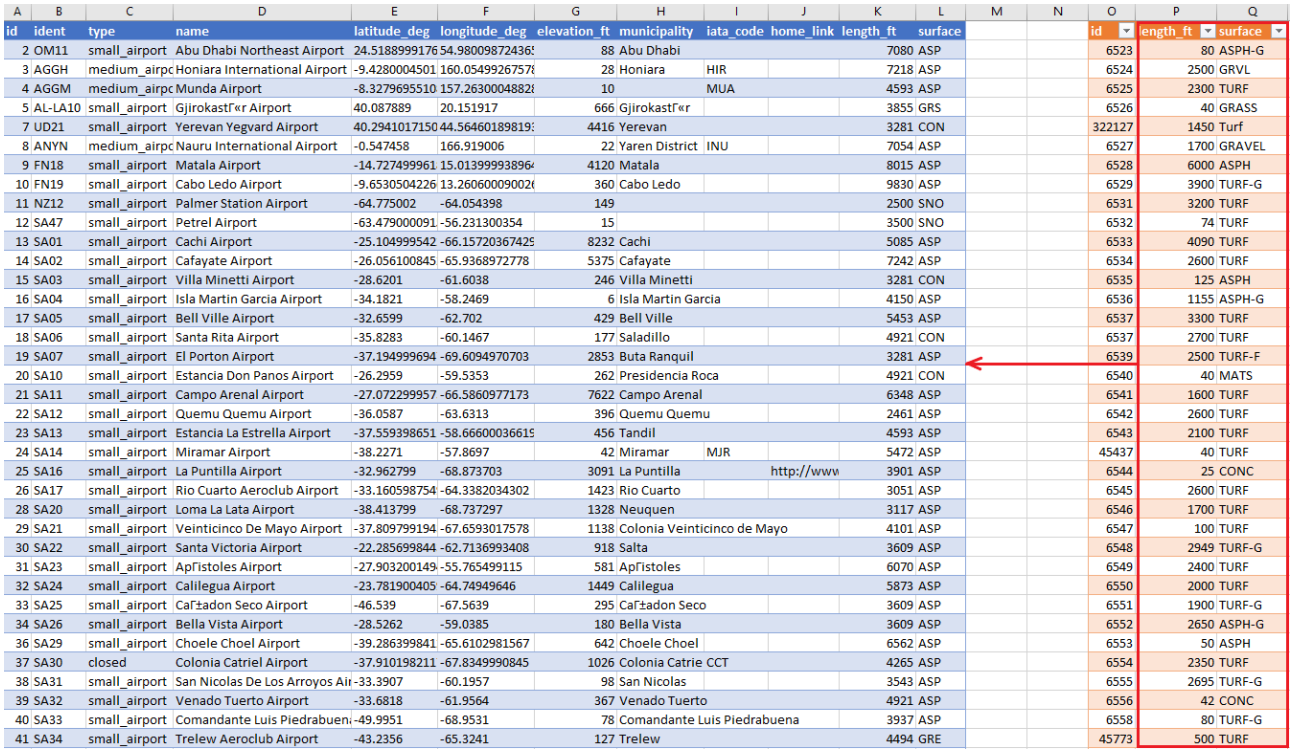
Таблица 3 – Описание колонок второго файла

Колонка	Значение колонки
id	Идентификатор аэропорта
surface	Поверхность взлётно-посадочной полосы
length_ft	Длина взлётно-посадочной полосы

3.2.1 Объединение исходных данных

Так как исходные данные представлены в виде двух отдельных таблиц, будет целесообразно объединить их в одну таблицу. Обе таблицы содержат общий столбец с уникальными идентификаторами аэропортов id, на основе

которого можно произвести объединение таблиц используя функцию ВПР в Microsoft Excel. На рисунке 5 приведены исходные таблицы, где красным прямоугольником указаны столбцы, которые будут присоединены после применения функции ВПР. На рисунке 6 приведена итоговая таблица.



id	ident	type	name	latitude_deg	longitude_deg	elevation_ft	municipality	iata_code	home_link	length_ft	surface
2	OM11	small_airport	Abu Dhabi Northeast Airport	24.5188999176	54.98009872436	88	Abu Dhabi			7080	ASP
3	AGGH	medium_airpc	Honiara International Airport	-9.4280004501	160.0549926757	28	Honiara	HIR		7218	ASP
4	AGGM	medium_airpc	Munda Airport	-8.3279695510	157.2630004882	10		MUA		4593	ASP
5	AL-LA10	small_airport	GjirokastГ«r Airport	40.087889	20.151917	666	GjirokastГ«r			3855	GRS
7	UD21	small_airport	Yerevan Yegvard Airport	40.2941017150	44.5646018981	4416	Yerevan			3281	CON
8	ANYN	medium_airpc	Nauru International Airport	-0.547458	166.919006	22	Yaren District	INU		7054	ASP
9	FN18	small_airport	Matala Airport	-14.727499961	15.01399993896	4120	Matala			8015	ASP
10	FN19	small_airport	Cabo Ledo Airport	-9.6530504226	13.2606009002	360	Cabo Ledo			9830	ASP
11	NZ12	small_airport	Palmer Station Airport	-64.775002	-64.054398	149				2500	SNO
12	SA47	small_airport	Petrel Airport	-63.479000091	-56.231300354	15				3500	SNO
13	SA01	small_airport	Cachi Airport	-25.104999542	-66.1572036742	8232	Cachi			5085	ASP
14	SA02	small_airport	Cafayate Airport	-26.056100845	-65.9368972778	5375	Cafayate			7242	ASP
15	SA03	small_airport	Villa Minetti Airport	-28.6201	-61.6038	246	Villa Minetti			3281	CON
16	SA04	small_airport	Isla Martin Garcia Airport	-34.1821	-58.2469	6	Isla Martin Garcia			4150	ASP
17	SA05	small_airport	Bell Ville Airport	-32.6599	-62.702	429	Bell Ville			5453	ASP
18	SA06	small_airport	Santa Rita Airport	-35.8283	-60.1467	177	Saladillo			4921	CON
19	SA07	small_airport	El Porton Airport	-37.194999694	-69.6094970703	2853	Buta Ranquil			3281	ASP
20	SA10	small_airport	Estancia Don Panos Airport	-26.2959	-59.5353	262	Presidencia Roca			4921	CON
21	SA11	small_airport	Campo Arenal Airport	-27.072299957	-66.5860977173	7622	Campo Arenal			6348	ASP
22	SA12	small_airport	Quemu Quemu Airport	-36.0587	-63.6313	396	Quemu Quemu			2461	ASP
23	SA13	small_airport	Estancia La Estrella Airport	-37.559398651	-58.6660003661	456	Tandil			4593	ASP
24	SA14	small_airport	Miramar Airport	-38.2271	-57.8697	42	Miramar	MJR		5472	ASP
25	SA16	small_airport	La Puntilla Airport	-32.962799	-68.873703	3091	La Puntilla		http://www	3901	ASP
26	SA17	small_airport	Rio Cuarto Aeroclub Airport	-33.160598754	-64.3382034302	1423	Rio Cuarto			3051	ASP
28	SA20	small_airport	Loma La Lata Airport	-38.413799	-68.737297	1328	Neuquen			3117	ASP
29	SA21	small_airport	Veinticinco De Mayo Airport	-37.809799194	-67.6593017578	1138	Colonia Veinticinco de Mayo			4101	ASP
30	SA22	small_airport	Santa Victoria Airport	-22.285699844	-62.7136993408	918	Salta			3609	ASP
31	SA23	small_airport	ApГistoles Airport	-27.903200149	-55.765499115	581	ApГistoles			6070	ASP
32	SA24	small_airport	Calilegua Airport	-23.781900405	-64.74949646	1449	Calilegua			5873	ASP
33	SA25	small_airport	CaГ±adon Seco Airport	-46.539	-67.5639	295	CaГ±adon Seco			3609	ASP
34	SA26	small_airport	Bella Vista Airport	-28.5262	-59.0385	180	Bella Vista			3609	ASP
36	SA29	small_airport	Choele Choel Airport	-39.286399841	-65.6102981567	642	Choele Choel			6562	ASP
37	SA30	closed	Colonia Catriel Airport	-37.910198211	-67.8349990845	1026	Colonia Catrie	CCT		4265	ASP
38	SA31	small_airport	San Nicolas De Los Arroyos Ai	-33.3907	-60.1957	98	San Nicolas			3543	ASP
39	SA32	small_airport	Venado Tuerto Airport	-33.6818	-61.9564	367	Venado Tuerto			4921	ASP
40	SA33	small_airport	Comandante Luis Piedrabueni	-49.9951	-68.9531	78	Comandante Luis Piedrabuena			3937	ASP
41	SA34	small_airport	Trelew Aeroclub Airport	-43.2356	-65.3241	127	Trelew			4494	GRE

id	length_ft	surface
6523	80	ASPH-G
6524	2500	GRVL
6525	2300	TURF
6526	40	GRASS
322127	1450	Turf
6527	1700	GRAVEL
6528	6000	ASPH
6529	3900	TURF-G
6531	3200	TURF
6532	74	TURF
6533	4090	TURF
6534	2600	TURF
6535	125	ASPH
6536	1155	ASPH-G
6537	3300	TURF
6537	2700	TURF
6539	2500	TURF-F
6540	40	MATS
6541	1600	TURF
6542	2600	TURF
6543	2100	TURF
45437	40	TURF
6544	25	CONC
6545	2600	TURF
6546	1700	TURF
6547	100	TURF
6548	2949	TURF-G
6549	2400	TURF
6550	2000	TURF
6551	1900	TURF-G
6552	2650	ASPH-G
6553	50	ASPH
6554	2350	TURF
6555	2695	TURF-G
6556	42	CONC
6558	80	TURF-G
45773	500	TURF

Рисунок 5 – Исходные таблицы

A	B	C	D	E	F	G	H	I	J	K	L
id	ident	type	name	latitude_deg	longitude_deg	elevation_ft	municipality	iata_code	home_link	length_ft	surface
2	OM11	small_airp	Abu Dhabi Northeast Airport	24.51889991760	54.9800987243652	88	Abu Dhabi			7080	ASP
3	AGGH	medium_	Honiara International Airport	-9.428000450134	160.05499267578	28	Honiara	HIR		7218	ASP
4	AGGM	medium_	Munda Airport	-8.327969551086	157.263000488281	10		MUA		4593	ASP
5	AL-LA10	small_airp	GjirokastГ«r Airport	40.087889	20.151917	666	GjirokastГ«r			3855	GRS
7	UD21	small_airp	Yerevan Yegvard Airport	40.29410171508	44.5646018981933	4416	Yerevan			3281	CON
8	ANYN	medium_	Nauru International Airport	-0.547458	166.919006	22	Yaren District	INU		7054	ASP
9	FN18	small_airp	Matala Airport	-14.72749996185	15.0139999389648	4120	Matala			8015	ASP
10	FN19	small_airp	Cabo Ledo Airport	-9.653050422668	13.2606000900268	360	Cabo Ledo			9830	ASP
11	NZ12	small_airp	Palmer Station Airport	-64.775002	-64.054398	149				2500	SNO
12	SA47	small_airp	Petrel Airport	-63.47900009159	-56.231300354	15				3500	SNO
13	SA01	small_airp	Cachi Airport	-25.1049995422	-66.157203674299	8232	Cachi			5085	ASP
14	SA02	small_airp	Cafayate Airport	-26.05610084529	-65.9368972778	5375	Cafayate			7242	ASP
15	SA03	small_airp	Villa Minetti Airport	-28.6201	-61.6038	246	Villa Minetti			3281	CON
16	SA04	small_airp	Isla Martin Garcia Airport	-34.1821	-58.2469	6	Isla Martin Garcia			4150	ASP
17	SA05	small_airp	Bell Ville Airport	-32.6599	-62.702	429	Bell Ville			5453	ASP
18	SA06	small_airp	Santa Rita Airport	-35.8283	-60.1467	177	Saladillo			4921	CON
19	SA07	small_airp	El Porton Airport	-37.19499969479	-69.6094970703	2853	Buta Ranquil			3281	ASP
20	SA10	small_airp	Estancia Don Panos Airport	-26.2959	-59.5353	262	Presidencia Roca			4921	CON
21	SA11	small_airp	Campo Arenal Airport	-27.0722999573	-66.5860977173	7622	Campo Arenal			6348	ASP
22	SA12	small_airp	Quemu Quemu Airport	-36.0587	-63.6313	396	Quemu Quemu			2461	ASP
23	SA13	small_airp	Estancia La Estrella Airport	-37.5593986511	-58.666000366199	456	Tandil			4593	ASP
24	SA14	small_airp	Miramar Airport	-38.2271	-57.8697	42	Miramar	MJR		5472	ASP
25	SA16	small_airp	La Puntilla Airport	-32.962799	-68.873703	3091	La Puntilla		http://www.a	3901	ASP
26	SA17	small_airp	Rio Cuarto Aeroclub Airport	-33.1605987549	-64.3382034302	1423	Rio Cuarto			3051	ASP
28	SA20	small_airp	Loma La Lata Airport	-38.413799	-68.737297	1328	Neuquen			3117	ASP
29	SA21	small_airp	Veinticinco De Mayo Airport	-37.8097991943	-67.6593017578	1138	Colonia Veinticinco de Mayo			4101	ASP
30	SA22	small_airp	Santa Victoria Airport	-22.28569984439	-62.7136993408	918	Salta			3609	ASP
31	SA23	small_airp	ApГistoles Airport	-27.90320014949	-55.765499115	581	ApГistoles			6070	ASP
32	SA24	small_airp	Calilegua Airport	-23.7819004059	-64.74949646	1449	Calilegua			5873	ASP
33	SA25	small_airp	CaГ±adon Seco Airport	-46.539	-67.5639	295	CaГ±adon Seco			3609	ASP
34	SA26	small_airp	Bella Vista Airport	-28.5262	-59.0385	180	Bella Vista			3609	ASP
36	SA29	small_airp	Choele Choel Airport	-39.28639984129	-65.6102981567	642	Choele Choel			6562	ASP
37	SA30	closed	Colonia Catriel Airport	-37.9101982117	-67.8349990845	1026	Colonia Catriel	CCT		4265	ASP
38	SA31	small_airp	San Nicolas De Los Arroyos Air	-33.3907	-60.1957	98	San Nicolas			3543	ASP
39	SA32	small_airp	Venado Tuerto Airport	-33.6818	-61.9564	367	Venado Tuerto			4921	ASP
40	SA33	small_airp	Comandante Luis Piedrabuena	-49.9951	-68.9531	78	Comandante Luis Piedrabuena			3937	ASP
41	SA34	small_airp	Trelew Aeroclub Airport	-43.2356	-65.3241	127	Trelew			4494	GRE

Рисунок 6 – Итоговая таблица после объединения

3.3 База данных

В реализуемом веб-приложении используется реляционная база данных PostgreSQL. Ключевой особенностью проектирования таблиц в Django является механизм объектно-ориентированного отображения (англ. Object-Relational Mapping, рус. объектно-реляционное отображение) [6], благодаря которому таблицы описываются программным кодом в виде классов в модуле models.py, где каждый класс является отдельной таблицей, а каждый атрибут – колонкой таблицы в БД.

В предыдущей главе была подготовлена таблица с исходными данными в формате xlsx (Microsoft Excel), содержащая данные о 53760 аэропортах. На основе полученной таблицы была написан класс в приложении Django, представленный на рисунках 7 и 8.

```python
class AirportGeneral(models.Model):
    id = models.AutoField(
        primary_key=True
    )
    name = models.CharField(
        verbose_name='Название аэропорта',
        max_length=500,
        default=''
    )
    latitude_deg = models.FloatField(
        verbose_name='Широта'
    )
    longitude_deg = models.FloatField(
        verbose_name='Долгота'
    )
    home_link = models.URLField(
        verbose_name='Ссылка на сайт',
        default=''
    )
    ident = models.CharField(
        verbose_name='ICAO',
        max_length=10,
        default=''
    )
    iata_code = models.CharField(
        verbose_name='IATA',
        max_length=10,
        default=''
    )
    elevation_ft = models.IntegerField(
        verbose_name='Высота НУМ',
        default=0,
        blank=True,
        null=True
    )
    municipality = models.CharField(
        verbose_name='Местоположение аэропорта',
        max_length=200,
        default=''
    )
    type = models.CharField(
        verbose_name='Тип',
        max_length=50,
        default=''
    )
    surface = models.CharField(
        verbose_name='Покрытие ВПП',
        max_length=100,
        default='',
        blank=True
    )
```

Рисунок 7 – Модель AirportGeneral

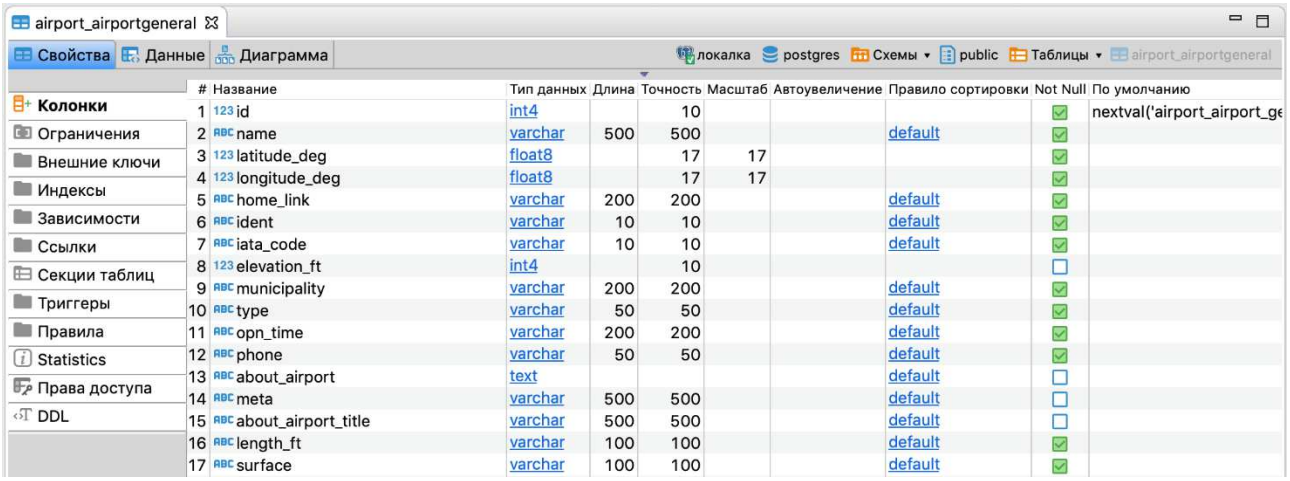
```
    length_ft = models.CharField(
        verbose_name='Длина ВПП',
        max_length=100,
        default='',
        blank=True,
    )
    opn_time = models.CharField(
        verbose_name='Время работы',
        max_length=200,
        default='',
        blank=True,
    )
    phone = models.CharField(
        verbose_name='Номер телефона',
        max_length=50,
        default='',
        blank=True,
    )
    about_airport_title = models.CharField(
        verbose_name=u'Заголовок текстового блока',
        help_text=u'Если поле пустое, то подставляется название аэропорта',
        max_length=500,
        null=True,
        blank=True
    )
    about_airport = tinymce_models.HTMLField(
        verbose_name=u'Блок информации о аэропорте',
        null=True,
        blank=True,
    )
    meta = models.CharField(
        verbose_name=u'Метатеги',
        help_text="Если поле пустое, то метатеги страницы генерируются в виде: "
                  "ICAO - Местоположение аэропорта / Название аэропорта",
        max_length=500,
        null=True,
        blank=True
    )

    class Meta:
        verbose_name = 'Аэропорт'
        verbose_name_plural = 'Аэропорты'

    def __str__(self):
        return self.name
```

Рисунок 8 – Модель AirportGeneral

Этот класс был преобразован фреймворком в таблицу «airport_airportgeneral», представленную на рисунке 9.



#	Название	Тип данных	Длина	Точность	Масштаб	Автоувеличение	Правило сортировки	Not Null	По умолчанию
1	id	int4		10				☑	nextval('airport_airport_ge
2	name	varchar	500	500			default	☑	
3	latitude_deg	float8		17	17			☑	
4	longitude_deg	float8		17	17			☑	
5	home_link	varchar	200	200			default	☑	
6	ident	varchar	10	10			default	☑	
7	iata_code	varchar	10	10			default	☑	
8	elevation_ft	int4		10				☐	
9	municipality	varchar	200	200			default	☑	
10	type	varchar	50	50			default	☑	
11	opn_time	varchar	200	200			default	☑	
12	phone	varchar	50	50			default	☑	
13	about_airport	text					default	☐	
14	meta	varchar	500	500			default	☐	
15	about_airport_title	varchar	500	500			default	☐	
16	length_ft	varchar	100	100			default	☑	
17	surface	varchar	100	100			default	☑	

Рисунок 9 – Колонки таблицы «airport_airportgeneral»

Была создана модель Bubbles (рисунок 10), необходимая для удобного оформления всплывающих подсказок на пользовательской части.

```
13  class Bubbles(models.Model):
14      block = models.CharField(
15          verbose_name=u'Карточка',
16          choices=TECH_CHOICES,
17          help_text=u'К какой технической характеристике будет относиться карточка',
18          max_length=255,
19      )
20      title = models.CharField(
21          verbose_name=u'Заголовок карточки',
22          max_length=50
23      )
24      text = models.TextField(
25          verbose_name=u'Текст карточки',
26          max_length=255
27      )
28
29      class Meta:
30          verbose_name = u'Карточка',
31          verbose_name_plural = u'Карточки'
32
33      def __str__(self):
34          return self.get_block_display()
```

Рисунок 10 – Модель Bubbles

Эти же поля воссозданы фреймворком в таблице «bubbles_bubbles» (рисунок 11).

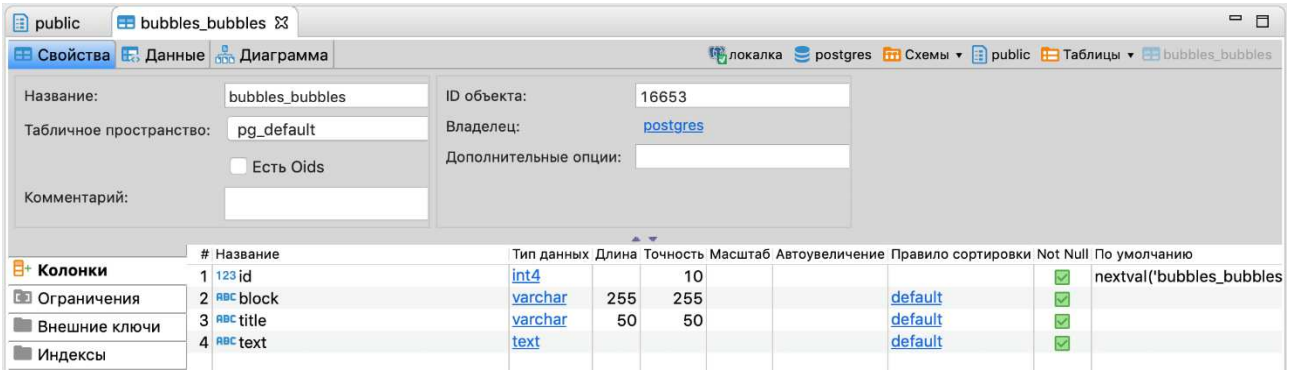


Рисунок 11 – Колонки таблицы Bubbles

Так же были созданы модели About (рисунок 12) и Terms (рисунок 13), необходимые для удобного заполнения текстом страниц “О проекте” и “Пользовательское соглашение” соответственно.

```
6   class About(models.Model):
7       text = tinymce_models.HTMLField(
8           verbose_name=u'Текст',
9           null=True,
10          blank=True,
11      )
12
13      class Meta:
14          verbose_name = 'Страница "О проекте"'
15          verbose_name_plural = 'Страница "О проекте"'
16
17      def __str__(self):
18          return str('Страница "О проекте"')
19
```

Рисунок 12 – Модель About

```
6   class Terms(models.Model):
7       text = tinymce_models.HTMLField(
8           verbose_name=u'Текст',
9           null=True,
10          blank=True,
11      )
12
13      class Meta:
14          verbose_name = 'Страница "Пользовательское соглашение"'
15          verbose_name_plural = 'Страница "Пользовательское соглашение"'
16
17      def __str__(self):
18          return str('Страница "Пользовательское соглашение"')
```

Рисунок 13 – Модель Terms

На рисунке 14 представлена итоговая схема БД, включающая в себя созданные выше модели, а также служебные таблицы Django.

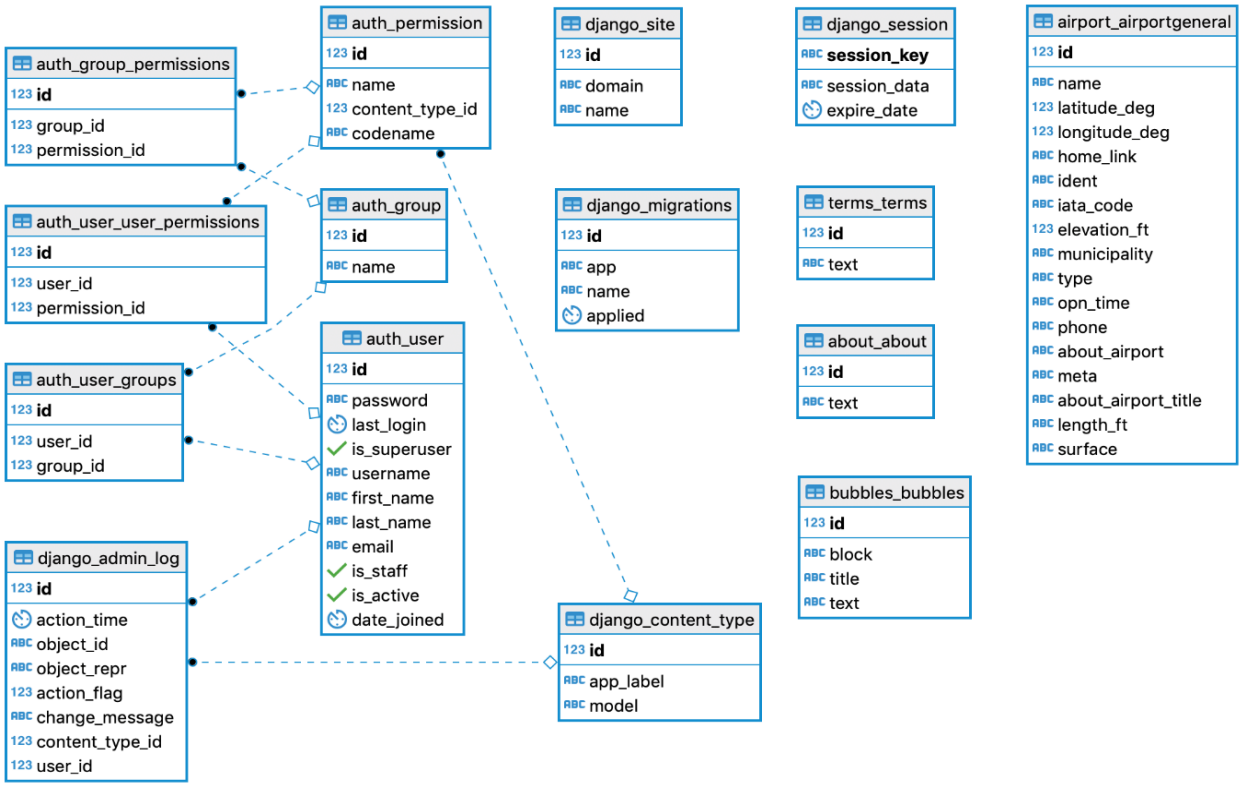


Рисунок 14 – Схема БД

3.4 Административная панель

После определения моделей проекта, приложение Django Admin может автоматически создать интерфейс администратора, позволяющий создавать, просматривать, обновлять и удалять записи зарегистрированных моделей. Регистрацию модели можно произвести с помощью декоратора @admin.register(Model). Регистрация модели AirportGeneral представлена на рисунке 15.

Рисунок 15 – ModelAdmin для модели AirportGeneral

В кортеже list_display перечислены колонки таблицы, которые будут отображаться при просмотре объектов модели AirportGeneral (рисунок 15), а в массиве search_fields перечислены колонки по которым будет осуществляться поиск по таблице.

Для того, чтобы войти в администраторскую панель, необходимо иметь учетную запись суперпользователя (англ. superuser), которая дает полный доступ к сайту и все необходимые разрешения.

Для создания суперпользователя необходимо исполнить в командной строке python manage.py createsuperuser, и ввести имя пользователя, адрес электронной почты и надежный пароль [7]. На рисунках 16 и 17 представлен интерфейс администраторской панели для модели AirportGeneral.

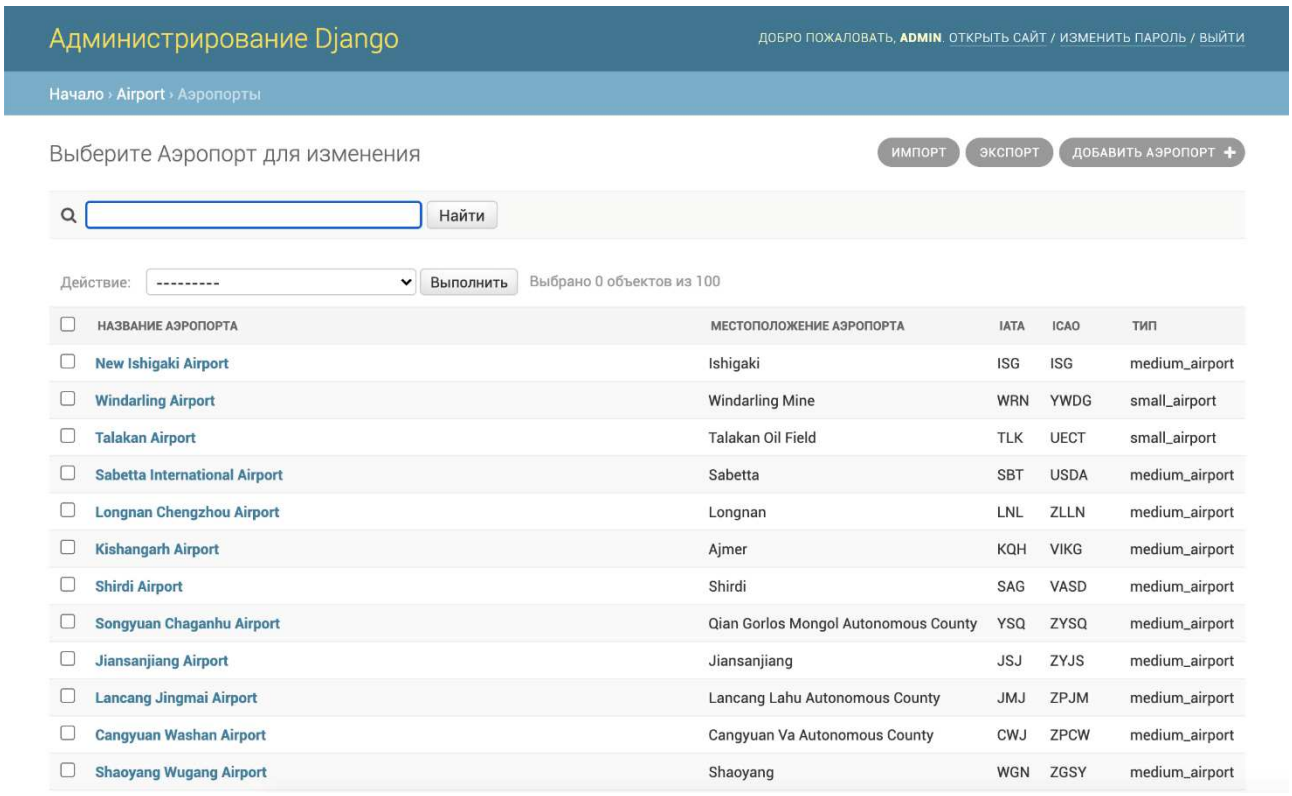


Рисунок 16 – Объекты модели AirportGeneral

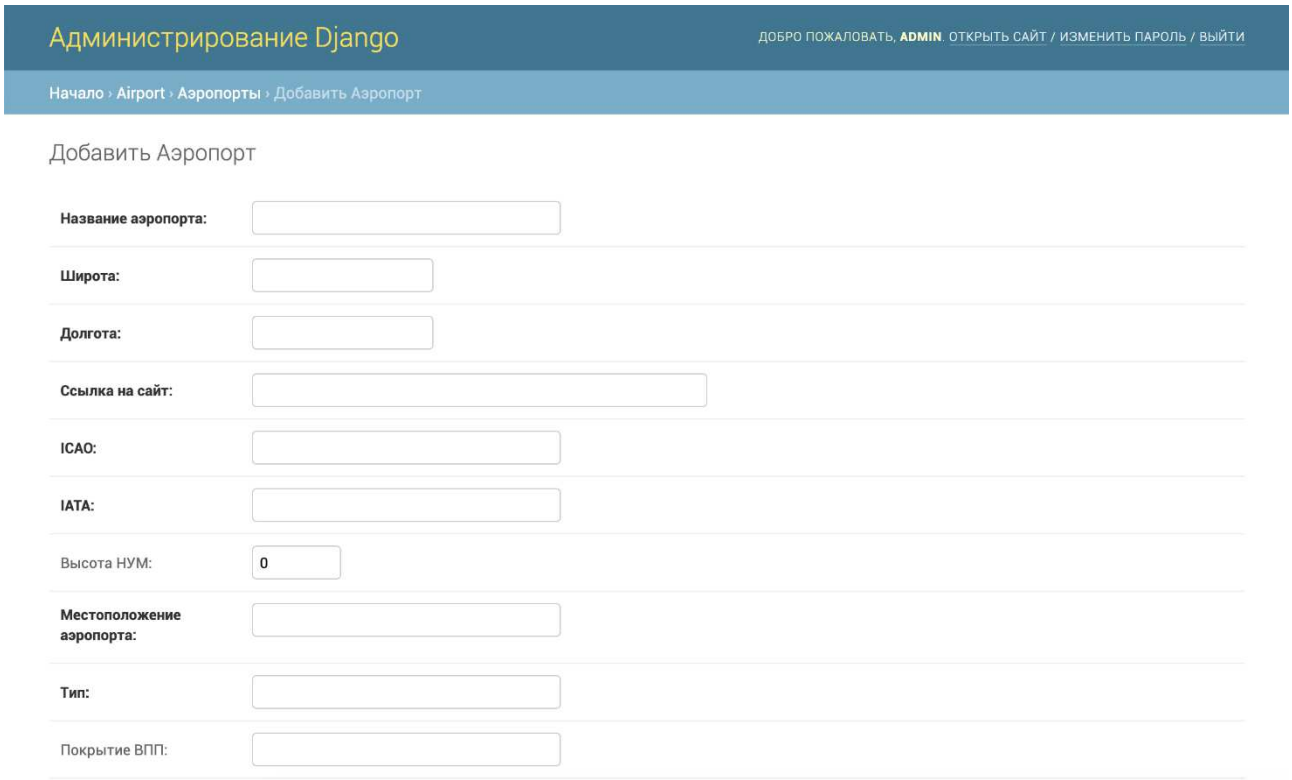


Рисунок 17 – Фрагмент страницы создания объекта

3.5 Импорт исходных данных в БД

В главе 3.2.1 были получены требуемые данные, которые необходимо преобразовать в объекты таблицы «airport_airportgeneral», создание которой было описано в главе 3.2. Так как исходные данные представлены в формате таблицы Microsoft Excel, а PostgreSQL не предоставляет функционал импорта данных из этого формата, было принято решение использовать сторонний модуль django-import-export для фреймоворка Django, благодаря которому импорт данных будет возможен прямо из администраторской панели.

Установка сторонних модулей в Django обычно производится в несколько простых шагов:

1. Установить требуемый модуль через менеджер пакетов для Python: pip install django-import-export;
2. Добавить название модуля в кортеж INSTALLED_APPS в settings.py.

Теперь модуль готов к использованию в проекте.

Для реализации задуманного функционала был написан класс AirtportGeneralResource (рисунок 18), в котором перечислены поля для импортирования из загружаемого файла и указаны настройки импорта.

```
class AirportGeneralResource(resources.ModelResource):
    def for_delete(self, row, instance):
        if row['iata_code']:
            return False
        return True

    class Meta:
        model = AirportGeneral
        skip_unchanged = True
        report_skipped = False
        import_id_fields = ('id',)
        fields = [
            'id', 'name', 'latitude_deg', 'longitude_deg', 'home_link', 'surface',
            'iata_code', 'elevation_ft', 'municipality', 'ident', 'type', 'length_ft'
        ]
```

Рисунок 18 – Класс AirportGeneralResource

Теперь в административной панели доступен импорт данных в таблицу «airport_airportgeneral» из файлов различных форматов. Интерфейс представлен на рисунке 19.

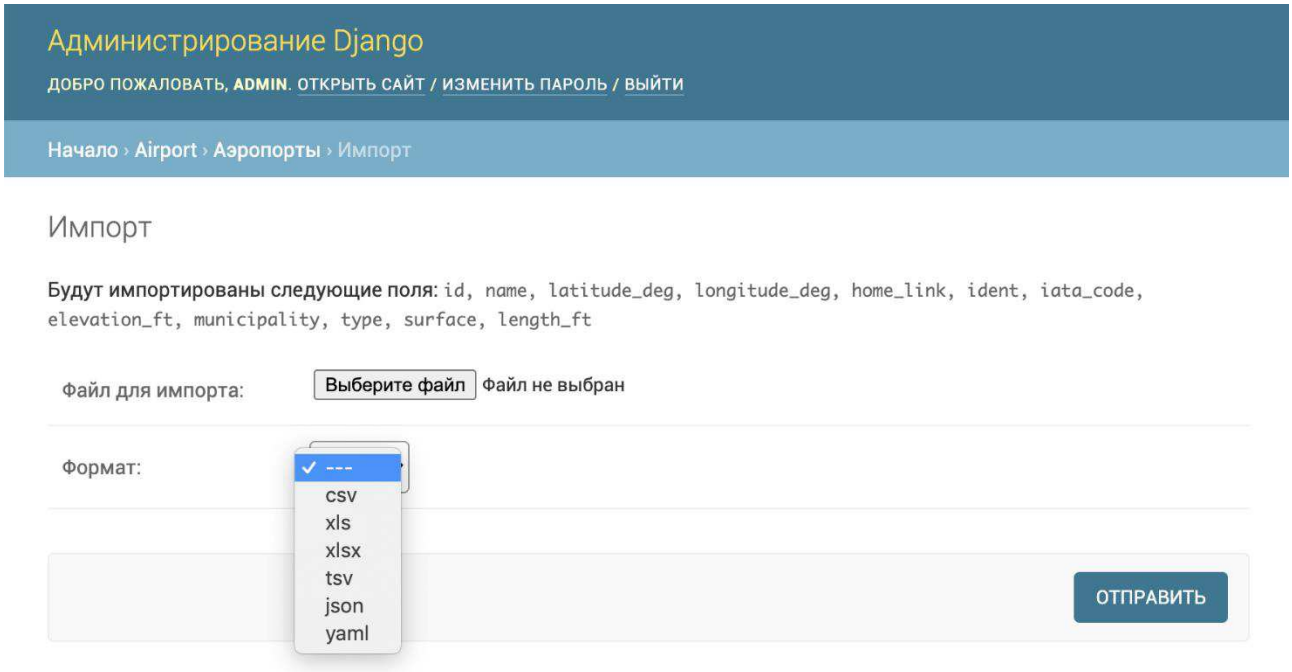


Рисунок 19 – Импорт в административной панели

3.6 Представления

Функция представления, или коротко представление – это функция Python, которая принимает Web-запрос и возвращает Web-ответ. Ответом может быть HTML-содержимое страницы, или перенаправление, или 404 ошибка, или XML-документ, или изображение, что угодно. Представление содержит всю необходимую логику для создания ответа. Этот код может находиться где угодно, главное, чтобы он находился в PYTHON_PATH. Никаких других требований нет – никакой “магии”. Несмотря на возможность расположить код представлений где угодно, принято держать его в файле views.py, который находится в каталоге проекта или приложения [8].

В данной ВКР будут использоваться представления, основанные на классах, представляющие собой альтернативный путь реализации представлений.

Основные отличия:

– организация обработки специфичных для HTTP методов (GET, POST, и т.д.) разнесена по соответствующим методам, вместо того чтобы писать кучу условий;

– объектно-ориентированные технологии, такие как миксины (примеси, множественное наследование) позволяют выделять код в компоненты, которые могут повторно использоваться.

Для реализации задуманного функционала потребуется три класса-представления:

1. MainView – для главной страницы сайта;
2. AiportView – для детальной страницы аэропорта;
3. GetAirportView – для реализации поиска на клиентской части;
4. AboutView – для страницы «О проекте»;
5. TermsView – для страницы «Пользовательское соглашение».

На рисунке 20 представлена View для детальный страницы аэропорта.

```
15  class AirportView(DetailView):
16      model = AirportGeneral
17      context_object_name = 'airport'
18      template_name = 'airport/airport_page.html'
19
20      def get_object(self, queryset=None):
21          iata_code = self.kwargs.get('iata_code')
22          return get_object_or_404(AirportGeneral, iata_code__iexact=iata_code)
23
24      def get_context_data(self, **kwargs):
25          context = super().get_context_data(**kwargs)
26          context['bubbles'] = Bubbles.objects.all()
27          return context
```

Рисунок 20 – Представления MainView и AirportView

Также для реализации поиска на клиентской части необходимо представление, которое будет получать GET-запрос, содержащий поисковую фразу, и возвращать объекты аэропортов, если были найдены пересечения.

Поиск пересечений осуществляется по четырем полям таблицы AirportGeneral:

– названию аэропорта;
– муниципалитету;
– iata_code;
– ident.

Найденные пересечения возвращаются на клиентскую часть в формате JSON и преобразовываются в HTML с помощью скрипта, написанного на языке JavaScript. Получившаяся View представлена на рисунке 21.

```
30  class GetAirportView(View):
31      def get(self, *args, **kwargs):
32          term = self.request.GET.get('term')
33          status = 200
34
35          airports = AirportGeneral.objects.filter(
36              Q(name__icontains=term) |
37              Q(ident__icontains=term) |
38              Q(iata_code__icontains=term) |
39              Q(municipality__icontains=term)
40          )[:3]
41
42          data = [({
43              'name': airport.name,
44              'url': airport.iata_code.lower(),
45              'icao': airport.ident,
46              'city': airport.municipality,
47          }) for airport in airports]
48
49          return JsonResponse(data, status=status, safe=False)
```

Рисунок 21 – Представление GetAirportView

3.7 Фронтенд

После того как были определены модели БД, импортированы исходные данные и написаны представления, пришло время написать код, который будет показывать информацию пользователям. Для этой задачи использовалась собственная система шаблонов Django, которая позволяет динамически генерировать HTML.

Одной из полезных особенностей Django является наследование шаблонов. Наследование позволяет использовать один и тот же HTML в разных шаблонах, что значительно сокращает количество повторяющихся участков кода. На рисунке 22 представлен базовый шаблон сайта, от которого наследуются остальные страницы.

```
{% load static %}
{% load leaflet_tags %}

<!DOCTYPE HTML>
<html lang="ru">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.3/css/all.css" in
    <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700" rel="stylesheet">
    <script src="https://code.jquery.com/jquery-3.3.1.js"    integrity="sha256-2Kok7MbOyxpgUV
    <script src="http://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" int
    <script src="https://unpkg.com/popper.js@1/dist/umd/popper.min.js"></script>
    <script src="https://unpkg.com/tippy.js@4"></script>
    {% leaflet_js %}
    {% leaflet_css %}
    {% block head %}{% endblock %}
    {% block styles %}{% endblock %}
  </head>
  <body>
    {% block content %}
    {% endblock %}

    {% block javascript %}
    {% endblock %}
  </body>
</html>
```

Рисунок 22 – Базовый шаблон

Для реализации поиска на клиентской части веб-приложения был написан скрипт на языке JavaScript (рисунок 23), который получает вводимые значения из поисковой строки и отправляет GET-запрос на серверную часть. В главе 3.6 было описано представление, которое получает GET-запросы с клиентской части, осуществляет поиск по БД и возвращает обратно на клиентскую часть найденные объекты аэропортов в формате JSON.

```
{% block javascript %}
  <script type="text/javascript">
      $("#airportInput").autocomplete({
                source: "/search/",
                minLength: 1,
                appendTo: ".search-form",
                messages: {
                    noResults: '',
                    results: function() {}
                },
                focus: function( event, ui ) {
                    $("#airportInput").val(ui.item.name);
                    return false;
                },
                select: function( event, ui ) {
                    $("#airportInput").val(ui.item.label);
                    return false;
                },
                close: function( event, ui ) {
                    $(".triangle").addClass('hidden');
                }
            })
            .data("ui-autocomplete")._renderItem = function(ul, item) {
                $(".triangle").removeClass('hidden');
                return $("<li></li>")
                    .data("item.autocomplete", item)
                    .append("<a href="+ item.url +">" + item.name + "</a>")
                    .append("<div class='municipality'>" +
                                "<img src='{% static 'img/icons/geo.png' %}'>" +
                                    "<span>" + item.city + "</span>" +
                            "</div>")
                    .appendTo(ul);
            };
  </script>
{% endblock %}
```

Рисунок 23 – Скрипт поиска

На рисунке 24 представлена главная страница с осуществленным поиском.

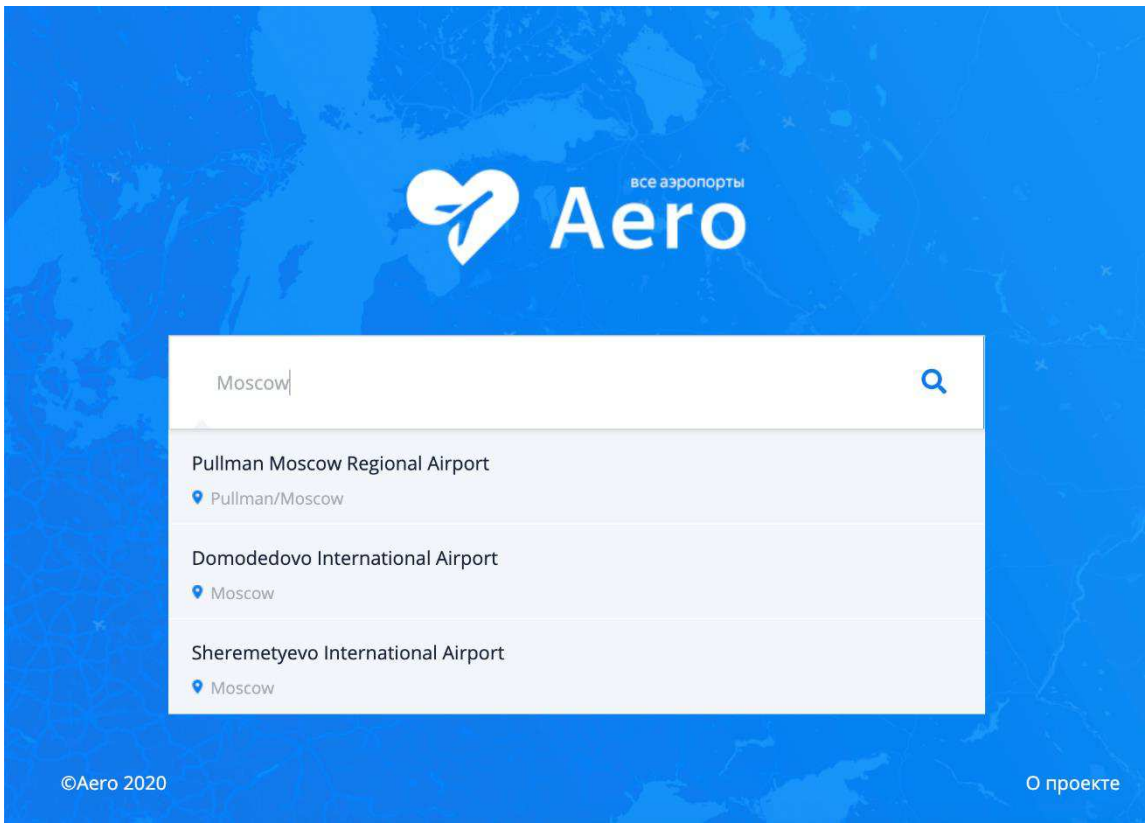


Рисунок 24 – Главная страница

Для отображения местоположения аэропорта на карте была использована библиотека leaflet для JavaScript, которая позволяет отображать интерактивную карту с заданными координатами на основе OpenStreetMap. На рисунке 25 представлен скрипт инициализации карты на основе широты и долготы – эти и остальные данные передаются в контексте, при генерации страницы.

```
22      function map_init_basic (map, options) {
23          map.setView([{{ airport.latitude_deg}}, {{ airport.longitude_deg }}], 13);
24          map.removeControl(map.zoomControl);
25      }
```

Рисунок 25 – Скрипт поиска

На рисунке 26 представлена детальная страница аэропорта Емельяново.

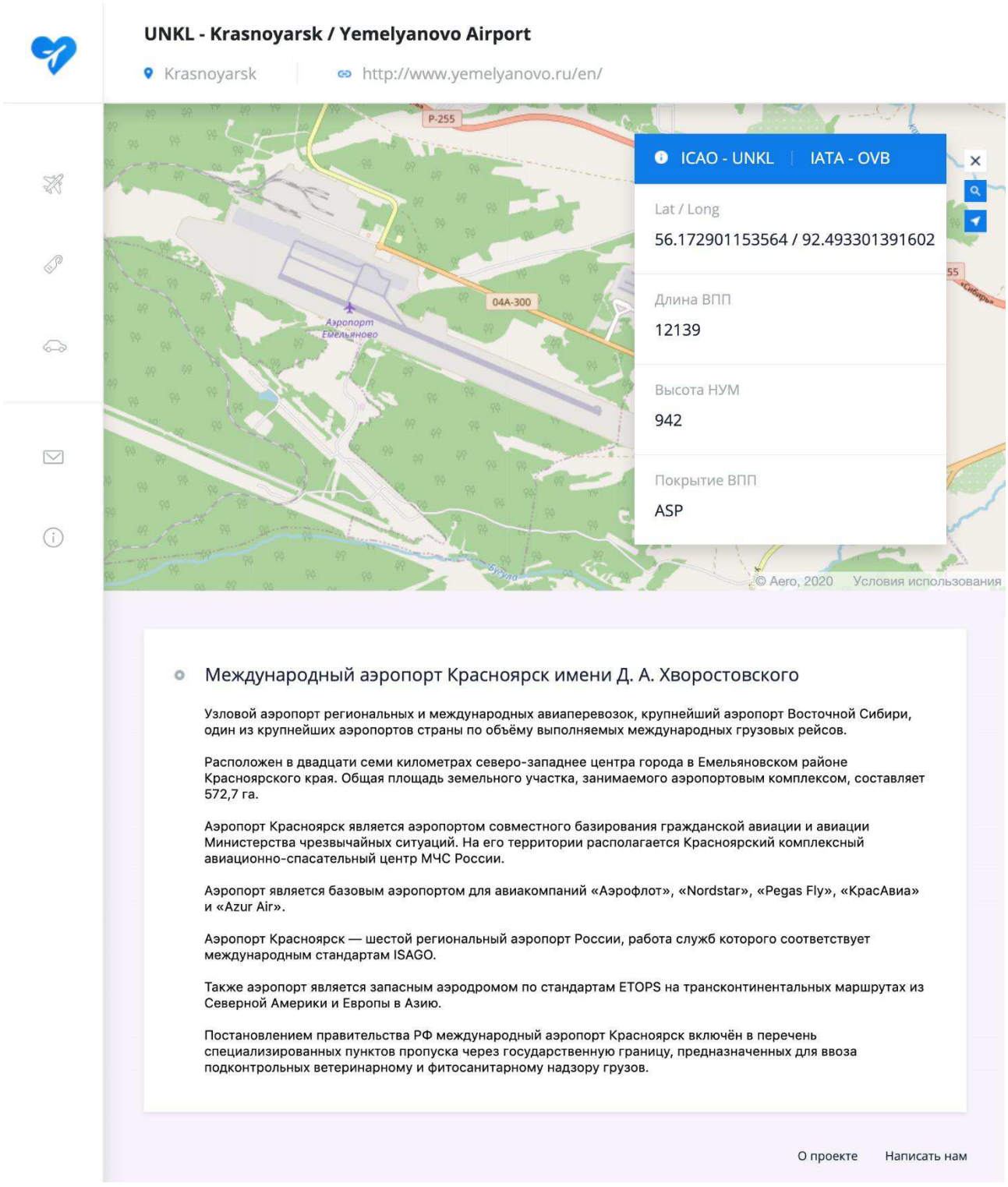


Рисунок 26 – Детальная страница аэропорта

ЗАКЛЮЧЕНИЕ

В результате проделанной работы был спроектировано и реализовано веб-приложения для просмотра сведений о гражданских аэропортах. В процессе проектирования и разработки были решены все задачи и достигнуты все поставленные цели.

Разработанное веб-приложение в дальнейшем позволит осуществлять поиск по базе данных на клиентской части и удобно просматривать подробную информацию об аэропортах. Реализован функционал позволяющий импортировать данные об аэропортах в базу данных веб-приложения. Также предусмотрена административная панель для создания, просмотра, редактирования и удаления данных. Среда веб-приложения реализована на мультиконтейнерной сборке на основе Docker Compose, что позволяет легко разворачивать рабочее окружение проекта.

В ходе выполнения выпускной квалификационной работы были полностью выполнены следующие задачи:

- проведен анализ существующих аналогов;
- определен технологический стек для разработки;
- спроектирована архитектура веб-приложения;
- спроектирована база данных;
- спроектирована серверная часть веб-приложения;
- разработана бизнес-логика;
- разработана клиентская часть веб-приложения.

СПИСОК СОКРАЩЕНИЙ

ВКР – выпускная квалификационная работа

ВП – веб-приложение

ЯП – язык программирования

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Python – краткий обзор языка [Электронный ресурс] — 2019 — Режим доступа: https://techrocks.ru/2019/01/21/about-python-briefly
2. Использование Django [Электронный ресурс] — 2010 — Режим доступа: http://www.ibm.com/developerworks/ru/library/l_django
3. PostgreSQL : Документация [Электронный ресурс] — 2017 — Режим доступа: https://postgrespro.ru/docs/postgresql/9.6/intro-whatis
4. Шаблоны – Документация Django [Электронный ресурс] — 2015 — Режим доступа: https://djbook.ru/rel1.8/topics/templates.html
5. Что такое Docker? [Электронный ресурс] — 2020 — Режим доступа: https://aws.amazon.com/ru/docker/
6. Реляционная база данных [Электронный ресурс] — 2017 — Режим доступа: https://ru.bmstu.wiki/Реляционная_база_данных
7. Руководство Django [Электронный ресурс] — 2019 — Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Admin_site
8. Введение в представления-классы [Электронный ресурс] — 2016 — Режим доступа: https://djbook.ru/rel1.9/topics/class-based-views/intro.html

ПРИЛОЖЕНИЕ А

Листинг программы парсинга и обработки данных

```
class AirportGeneralResource(resources.ModelResource):
    def for_delete(self, row, instance):
        if row['iata_code']:
            return False
        return True

    class Meta:
        model = AirportGeneral
        skip_unchanged = True
        report_skipped = False
        import_id_fields = ('id',)
        fields = [
            'id', 'name', 'latitude_deg', 'longitude_deg', 'home_link', 'surface',
            'iata_code', 'elevation_ft', 'municipality', 'ident', 'type', 'length_ft'
        ]
```

ПРИЛОЖЕНИЕ Б

Листинг бизнес-логики для главной страницы, детальной страницы и поиска

```
class MainView(TemplateView):
    template_name = 'main/index.html'


class AirportView(DetailView):
    model = AirportGeneral
    context_object_name = 'airport'
    template_name = 'airport/airport_page.html'

    def get_object(self, queryset=None):
        iata_code = self.kwargs.get('iata_code')

        return get_object_or_404(
            AirportGeneral,
            iata_code__iexact=iata_code
        )

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['bubbles'] = Bubbles.objects.all()
        return context


class GetAirportView(View):
    def get(self, *args, **kwargs):
        term = self.request.GET.get('term')
        status = 200

        airports = AirportGeneral.objects.filter(
            Q(name__icontains=term) |
            Q(ident__icontains=term) |
            Q(iata_code__icontains=term) |
            Q(municipality__icontains=term)
        )[:3]

        data = [({
            'name': airport.name,
            'url': airport.iata_code.lower(),
            'icao': airport.ident,
            'city': airport.municipality,
        }) for airport in airports]

        return JsonResponse(data, status=status, safe=False)
```

ПРИЛОЖЕНИЕ В

Листинг программы инициализации карты и поиска для детальной страницы аэропорта

```
<script type="text/javascript">
  // map
  function map_init_basic (map, options) {
     map.setView([{{ airport.latitude_deg}}, {{ airport.longitude_deg }}], 13);
     map.removeControl(map.zoomControl);
  }

  // search
  function truncate(string){
    if (string.length > 24)
       return string.substring(0,24)+'...';
    else
       return string;
  };
  $("#searchInput").autocomplete({
         source: "/search/",
         minLength: 1,
         appendTo: ".search-block",
         messages: {
            noResults: '',
            results: function() {}
         },
         focus: function(event, ui) {
            $("#searchInput").val(ui.item.name);
            return false;
         },
         select: function(event, ui) {
            $("#searchInput").val(ui.item.label);
            return false;
         }
     })

     .data("ui-autocomplete")._renderItem = function(ul, item) {
         return $("<div></div>")
            .data("item.autocomplete", item)
            .append("<a href=/" + item.url + ">"  + item.icao + " - " + truncate(item.name) + "</a>")
            .append("<div class='municipality'><img src='{% static 'img/icons/geo.png' %}'><span>"
+ item.city + "</span></div>")
            .appendTo(ul);
     };
</script>
```

ПРИЛОЖЕНИЕ Г

Листинг программы для разворачивания рабочего окружения веб-приложения

```
version: '3'
services:
   server:
     build:
       context: ./
       dockerfile: ./server/Dockerfile
     command: python manage.py runserver 0.0.0.0:8000
     volumes:
       - ./server:/server
     ports:
       - "8000:8000"
     depends_on:
       - db
     environment:
        DEBUG: 'True'
        DATABASE_URL: 'postgres://postgres:@db:5432/postgres'

   db:
      image: postgres:11.2
      environment:
         POSTGRES_DB: postgres
         POSTGRES_USER: postgres
         POSTGRES_PASSWORD: postgres
      ports:
         - 3406:5432
```

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой ВТ
_____ О.В. Непомнящий
подпись инициалы, фамилия
« ___ » _____ 2020 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Веб-приложение для просмотра сведений о гражданских аэропортах
тема

Руководитель _____ доцент, к.т.н. С.Н. Титовский
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник 26.06.2020 Д.И. Моргачев
подпись, дата инициалы, фамилия

Нормоконтролер _____ доцент, к.т.н. С.Н. Титовский
подпись, дата должность, ученая степень инициалы, фамилия

Красноярск 2020