**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**ВЫСШЕГО ОБРАЗОВАНИЯ**

**“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”**

Факультет *компьютерных наук*

Кафедра информационных систем

*Система покупки электронных авиабилетов*

*Курсовой проект*

*по дисциплине*

*Технологии программирования*

*09.03.02. Информационные системы и технологии*

*Информационные системы и сетевые технологии*

Преподаватель *\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.С. Тарасов, ассистент* \_\_.\_\_.20\_\_

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_ *А.П. Ашурков, О.И. Перфильева, 3 курс, д/о*

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ *И.Ю. Иванов, преподаватель*

Воронеж2019

Оглавление

[Введение 3](#_Toc10515984)

[1. Постановка задачи 4](#_Toc10515985)

[1.1. Цель 4](#_Toc10515986)

[1.2. Сфера использования: 4](#_Toc10515987)

[1.3. Требования: 4](#_Toc10515988)

[1.4. Задачи: 4](#_Toc10515989)

[2. Обоснование выбора средств разработки 5](#_Toc10515990)

[3. Модульная схема 8](#_Toc10515991)

[4. Анализ предметной области 9](#_Toc10515992)

[4.1. Область применения 9](#_Toc10515993)

[4.2. Общие положения 9](#_Toc10515994)

[5. Диаграммы, используемые в данном проекте 10](#_Toc10515995)

[5.1. Диаграмма классов 10](#_Toc10515996)

[5.2. Use Case 11](#_Toc10515997)

[5.3. State Machine 12](#_Toc10515998)

[5.4. ER-диаграмма 12](#_Toc10515999)

[6. План тестирования 13](#_Toc10516000)

[ЗАКЛЮЧЕНИЕ 15](#_Toc10516001)

# Введение

В современном мире оптимальнее совершать покупки через Интернет. В данном проекте поднимается вопрос актуальности покупки авиабилетов. Чтобы приобрести билет достаточно иметь доступ в Интернет и быть клиентом одной из поддерживаемых платёжных систем. Покупка электронного билета через сайт существенно экономит Ваше время и средства - купить его можно заранее и без комиссии. Покупать авиабилеты на сайте намного выгоднее и удобнее, чем в кассе. Поэтому целью нашей работы является создание такого сервиса, который позволит пользователю оптимизировать процесс покупки авиабилетов.

# 1. Постановка задачи

## 

## 1.1. Цель

разработать систему покупки электронных авиабилетов.

## 

## 1.2. Сфера использования

путешествия, работа.

## 

## 1.3. Требования

1. Регистрация и авторизация пользователей.

2. Для обычного пользователя – покупка и возврат билетов, поиск и просмотр информации о рейсах, просмотр списка купленных им билетов.

3. Для администратора – поиск и просмотр информации о рейсах, а также Возможность добавления и удаления рейсов и всей информации о них.

4. Возможность поиска и просмотра информации о рейсах неавторизованным пользователем.

## 

## 1.4. Задачи

1. Провести анализ требований к разрабатываемой системе.

2. Спроектировать приложение.

3. Реализовать приложение, удовлетворяющее указанным требованиям, и описать процесс разработки и итоговый результат.

# 2. Обоснование выбора средств разработки

В основе всего проекта лежит «клиент-серверная» технология, которая интересна и актуальна главным образом потому, что обеспечивает простое и относительно дешевое решение проблемы коллективного доступа к базам данных. Архитектура «клиент-сервер» определяет общие принципы организации взаимодействия в сети, где имеются серверы, узлы-поставщики некоторых специфичных функций (сервисов) и клиенты, потребители этих функций.

В данной работе мы будем использовать HTML, CSS, Python и СУБД SQLight.

Применение вычислительной техники невозможно без рациональной организации информационной базы и обеспечение эффективного доступа к ней пользователя. Для этой цели и служат базы данных. Широкое использование баз данных различными категориями пользователей привело, с одной стороны, к созданию интерфейсов, требующих минимум времени на освоение средств управления системой, , а с другой - к построению мощных, гибких СУБД, имеющих, в том числе, развитые средства защиты данных от случайного или преднамеренного разрушения. В нашем проекте будем использовать SQLite - это БД которая хранит базу в одном файле и не требует отдельного процесса для запуска, при этом использует не стандартный вариант языка SQL. Такой подход позволяет встроить sqlite прямо в программу, без необходимости установки сервера БД. SQLite, на практике, нередко оказывается в 2-3 раза (и даже больше) быстрее MySQL. Такое возможно благодаря высоко упорядоченной внутренней архитектуре и устранению необходимости в соединениях типа «сервер-клиент» и «клиент-сервер».

Всё это, собранное в один пакет, лишь немногим больше по размеру клиентской части библиотеки MySQL, является впечатляющим достижением для полноценной базы данных. Используя высокоэффективную инфраструктуру, SQLite может работать в крошечном объеме выделяемой для неё памяти, гораздо меньшем, чем в любых других системах БД.

Преимущества:

1. Файловая структура - вся база данных состоит из одного файла, поэтому её очень легко переносить на разные машины
2. Отсутствие необходимости настройки сервера СУБД
3. Полностью свободная лицензия
4. Кроссплатформенность
5. Высокая скорость простых операций выборки данных
6. Поддержка транзакций, триггеров, представлений (views), вложенных запросов
7. Безопасность. БД хранится в одном файле, права доступа к которому можно контролировать стандартными средствами ОС
8. Очень экономичная, в плане ресурсов, архитектура

SQLite прекрасно подойдет для проектов у которых мало операций записей, не нужна система прав доступа к БД и ограничены ресурсы сервера.

* HTML (HyperText Markup Language) - язык разметки гипертекста – предназначен для создания Web-страниц. Под гипертекстом в этом случае понимается текст, связанный с другими текстами указателями-ссылками. HTML представляет собой достаточно простой набор кодов, которые описывают структуру документа. HTML позволяет выделить в тексте отдельные логические части (заголовки, абзацы, списки и т.д.), поместить на Web-страницу подготовленную фотографию или картинку, организовать на странице ссылки для связи с другими документами. HTML не задает конкретные и точные атрибуты форматирования документа. Конкретный вид документа окончательно определяет только программа-браузер на компьютере пользователя Интернета.

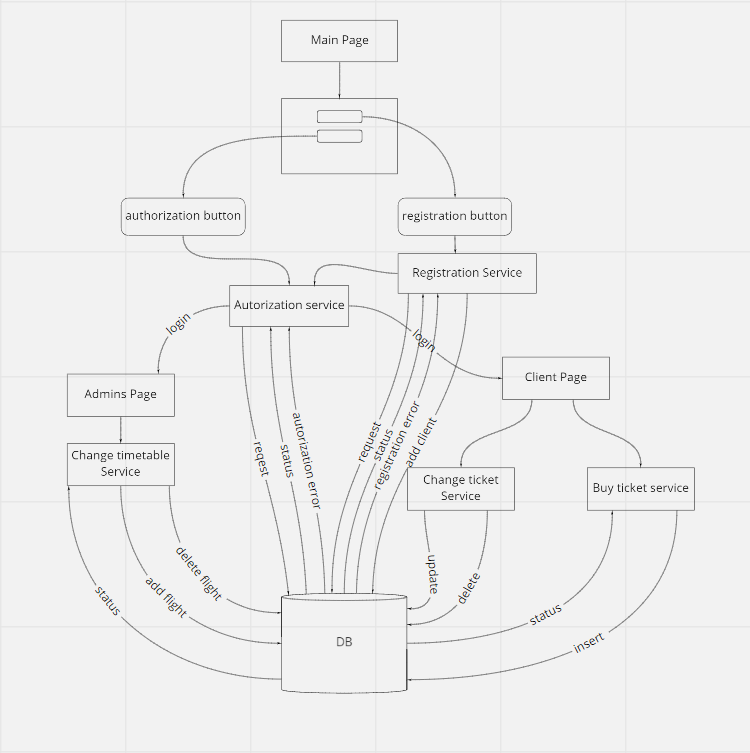
HTML применяется для того, чтобы определить как должен отображаться ваш контент: в виде абзаца, списка, заголовка, ссылки, изображения, мультимедийного проигрывателя, формы или же в виде одного из множества других доступных элементов, а также возможного нового элемента. У него нет аналогов.

* CSS (англ. Cascading Style Sheets - каскадные таблицы стилей) - формальный язык описания внешнего вида документа, написанного с использованием языка разметки. CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля. Для включения стилей в html-документ применяются - каскадные таблицы стилей CSS (Cascading Style Sheet), с их возможностями значительно упрощается разработка web-страниц и управление полученным сайтом.
* В данном приложении мы будем использовать язык программирования Python - высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Мы будем использовать Python-фреймворки, т.к. фреймворки позволяют легко и быстро создать базовую логику бэкенда. Она включает в себя сопоставление разных URL-адресов с частями Python-кода, работу с базами данных, создание HTML-представлений для отображения на устройствах пользователя.

В данном приложении мы будем использовать web-фреймворк Flask, т.к. он обеспечивает простоту, гибкость и полный контроль над проектом. Он позволяет пользователю самостоятельно решать, как реализовывать те или иные вещи.

# 3. Модульная схема



# 4. Анализ предметной области

В настоящее время на просторах Интернета можно обнаружить огромное количество сайтов, связанных с продажей авиабилетов. Электронный авиабилет является цифровым аналогом обычного билета и имеет множество положительных качеств. Такой билет невозможно потерять, ведь это не кусочек бумаги, а электронная запись. Когда вы покупаете такой билет, информация о нем вносится в базу данных.

## 4.1. Область применения

Данное приложение будут использовать люди, жизнь которых так или иначе связана с перелетами.

## 4.2. Общие положения

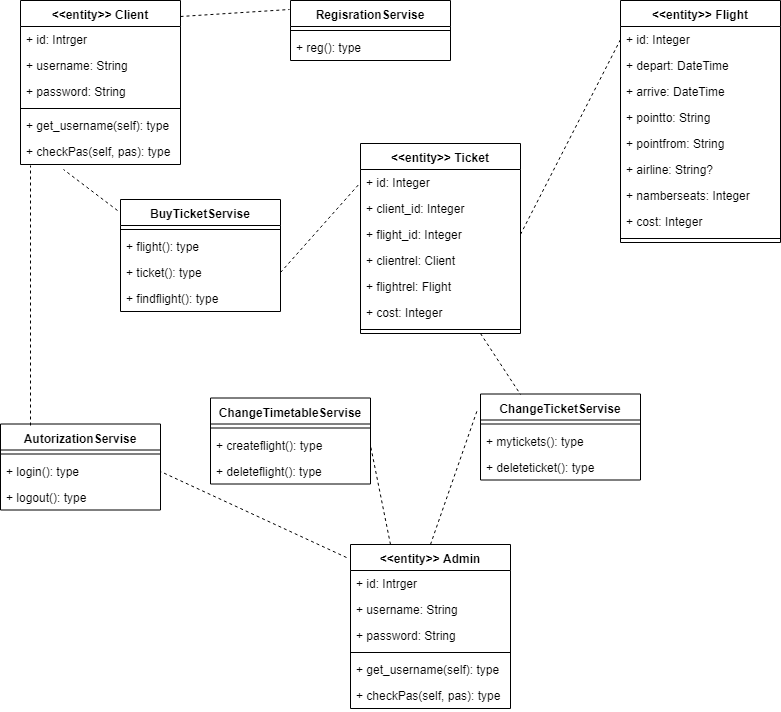
В нашем приложении на начальной странице Вам предоставляется возможность поиска и просмотра необходимой информации о всех рейсах.

Для покупки билетов необходимо пройти процесс регистрации, затем авторизации, если пользователь ранее не пользовался нашим приложением, иначе - только процесс авторизации доступные по ссылкам на начальной странице.

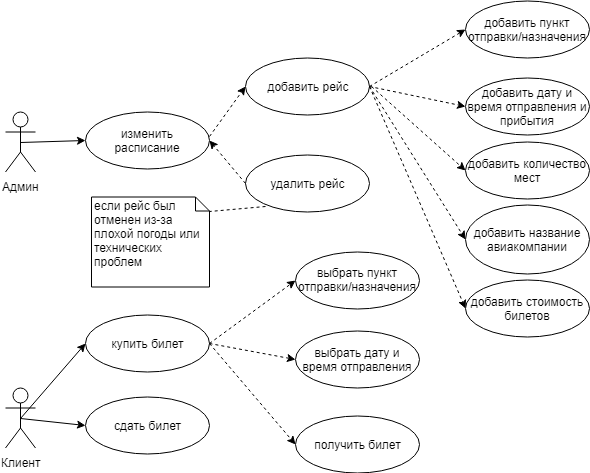
Затем Вы сможете приобрести билет на выбранный вами рейс, а также получаете доступ к просмотру списка купленных вами билетов и возможность их сдать.

# 5. Диаграммы, используемые в данном проекте

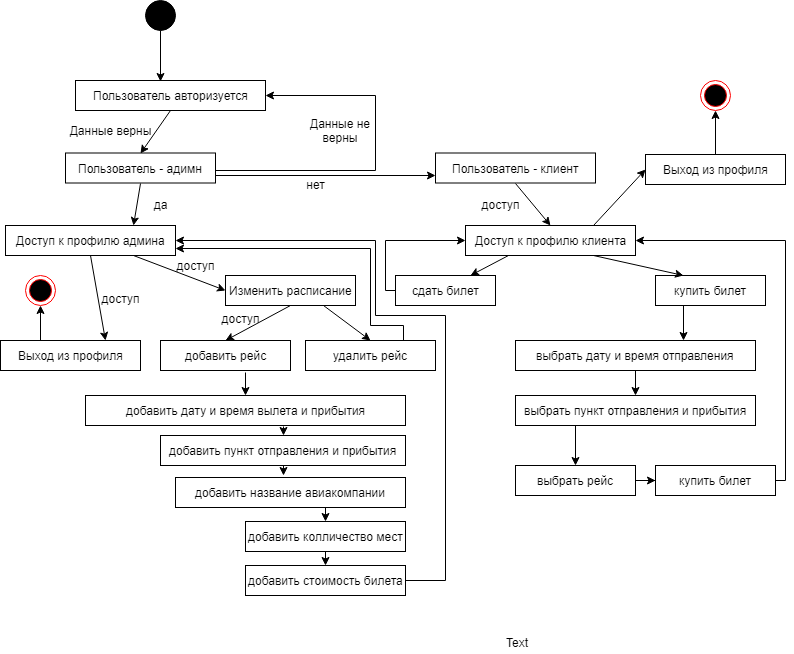
## 5.1. Диаграмма классов



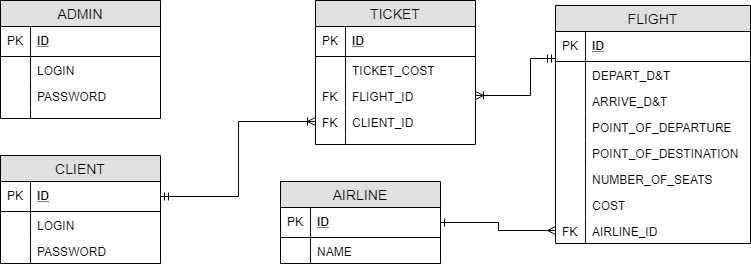
## 5.2. Use Case

****

## 5.3. State Machine

****

## 5.4. ER-диаграмма

****

# 6. План тестирования

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **название** | **предпосылки** | **Шаг** | **Ожидаемый результат** | **Отметка о прохождении** | **Коментарий** |
| **1** | **стартовая страница** | | | | | |
|  | регистрация |  | нажатие кнопки "зарегистрироваться" на странице регистрации/авторизации, заполнение полей Логин, Пароль, ФИО | добавление в бд новой уникальной записи пользователя |  |  |
|  | аутентификация |  | нажатие кнопки "авторизироваться" на странице регистрации/авторизации, заполнение полей логин и пароль | авторизация и переход в профиль админа или клиента |  |  |
| **2** | **профиль администратора** | | | | | |
| 2.1 | добавление рейса |  | нажатие кнопки "добавить рейс" на странице администратора | открытие страницы, на которой возможно создание новой уникальной записи бд с обязательными для заполнения полями |  |  |
| 2.1.1 | добавить пункт отправки/назначения | добавление рейса |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 2.1.2 | добавить дату отправления/прибытия | добавление рейса |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 2.1.3 | добавить стоимость билета | добавление рейса |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 2.1.4 | добавить количество мест | добавление рейса |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 2.1.5 | добавить название авиакомпании | добавление рейса |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 2.1.6 | добавить рейс | добавление рейса | нажатие кнопки "Добавить" | производится проверка введенных в поля записи данных, и при соответствии записи условию уникальности, она добавляется в бд, затем возврат на страницу админа. Иначе выявл ошибка в диалог окн. с предупреждением о неверных данных. |  |  |
| 2.2 | удаление рейса | изменение расписания | нажатие кнопки "удалить рейс" рядом с соответствующей записью таблицы рейсрв на страницеа дминистратора | удаление записи из бд, обновление страницы |  |  |
| **3** | **профиль пользователя** | | | | | |
| 3.1 | покупка билета |  | нажатие кнопки "Купить билет" на странице пользоватля | открытие страницы, на которой возможно выбрать необходимые параметры удля покупки билета |  |  |
| 3.1.1 | выбрать пункт отправки/назначения | покупка билета |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 3.1.2 | выбрать дату вылета | покупка билета |  | заполнение соответствующего поля, проверка введенных данных на правильность |  |  |
| 3.1.3 | купить билеты | покупка билета | нажатие кнопки "Купить" | появление окна c информацией об успешной покупке |  |  |
| 3.2 | сдача билета |  | нажатие кнопки "Сдать билет" на странице пользоватля |  |  |  |

# 

# ЗАКЛЮЧЕНИЕ

Мы разработали web-приложение, которое позволяет приобрести авиабилет не выходя из комнаты, тем самым, экономя время и деньги. А чтобы приобрести билет достаточно иметь доступ в Интернет и быть клиентом одной из поддерживаемых платёжных систем.