



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

КУРСОВАЯ РАБОТА

по дисциплине: Разработка клиент-серверных приложений
по профилю: Разработка программных продуктов и проектирование информационных систем
направления профессиональной подготовки: 09.03.04 «Программная инженерия»

Тема: «Разработка клиент-серверного фуллстек-приложения «Каршеринг»»

Студент: Смирнов Егор Олегович

Группа: ИКБО-13-19

Работа представлена к защите _____ (дата) _____ / _____ /
(подпись и ф.и.о. студента)

Руководитель: к.т.н., доцент Куликов Александр Анатольевич

Работа допущена к защите _____ (дата) _____ / _____ /
(подпись и ф.и.о. рук-ля)

Оценка по итогам защиты: _____

_____/_____/_____
_____/_____/_____

(подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших
защиту)

2020г.

Заведующему кафедрой
инструментального и прикладного
программного обеспечения (ИиППО)
Института информационных технологий (ИТ)

Болбакову Роману Геннадьевичу
От студента Смирнова Егора Олеговича

ФИО
ИКБО-13-19
группа
3 курс
курс

Контакт: +7 (965) 350-67-39

Заявление

Прошу утвердить мне тему *курсовой работы/курсового проекта* по дисциплине «Разработка клиент-серверных приложений» образовательной программы бакалавриата 09.03.04 (Программная инженерия) Тема: Разработка клиент-серверного фулстек-приложения «Каршеринг».

Приложение: лист задания на КР/КП в 2-ух экземплярах на двухстороннем листе (проект)

Подпись студента

 / Смирнов Е.О
подпись ФИО

Дата

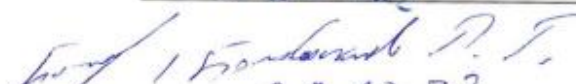
28.02.2022

Подпись руководителя

 /к.т.н, доцент Куликов А.А
подпись Должность, ФИО

Дата

28.02.2022

 / Болбаков Р.Г.
28.02.22



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА - Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ЗАДАНИЕ
на выполнение курсовой работы

по дисциплине: Разработка клиент-серверных приложений
по профилю: Разработка программных продуктов и проектирование информационных систем
направления профессиональной подготовки: Программная инженерия (09.03.04)
Студент: Смирнов Егор Олегович
Группа: ИКБО-13-19
Срок представления к защите: 18.05.2022
Руководитель: к.т.н., доцент Куликов Александр Анатольевич

Тема: «Разработка клиент-серверного фуллстек-приложения "Каршеринг"»

Исходные данные: PostgreSQL, React, Node.js, Java Spring, Git, GitHub, Heroku.

Перечень вопросов, подлежащих разработке, и обязательного графического материала: 1. Провести анализ предметной области для выбранной темы с обоснованием выбора клиент-серверной архитектуры для разрабатываемого приложения; 2. Выбрать, ориентируясь на мировой опыт и стандарты в данной области, а также на выбранную тему, программный стек для реализации фуллстек разработки; 3. Дать описание архитектуры разрабатываемого клиент-серверного приложения, с помощью UML нотаций и с обоснованием выбора вида клиента(браузер, мобильное приложение, кроссплатформенный клиент и т.д.) ; 4.Провести реализацию фронтенд и бекенд части клиент-серверного приложения, обеспечив версионный контроль процесса разработки с помощью Git. В разработанном приложении должна обеспечиваться авторизация и аутентификация пользователя; 5. Разместить проект клиент-серверного приложения в репозитории GitHub с приложением в тексте отчёта ссылки на данный репозиторий; 6. Интегрировать проект на GitHub с Heroku, с целью развёртывания разработанного клиент-серверного приложения в облаке. Ссылку на приложение в облаке Heroku привести в тексте пояснительной записки. 7. Провести проверку функционирования минимально жизнеспособного продукта с использованием сгенерированных тестовых данных. 8. Разработать презентацию с графическими материалами.

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

Зав. кафедрой ИиППО: Болбаков Р. Г. / Болбаков Р. Г. /, « 28 » 02 2022 г.

Задание на КР выдал: Куликов А.А. / Куликов А.А. /, « 28 » 02 2022 г.

Задание на КР получил: Смирнов Е.О. / Смирнов Е.О. /, « 28 » февраля 2022 г.

ОГЛАВЛЕНИЕ

1. ОБЩИЕ СВЕДЕНИЯ.....	6
1.1. Обозначение и наименование программы.....	7
1.2. Программное обеспечение, необходимое для функционирования	7
1.3. Языки программирования, на которых написана программа.....	7
2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ	8
3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ	9
3.1. Анализ предметной области	9
3.1.1. Описание предметной области.....	9
3.1.2. Анализ аналогов разрабатываемого приложения	9
3.2. Методология разработки сервиса.....	11
3.2.1. Выбор языка разработки	11
3.2.2. Выбор инструментов для разработки интерфейса приложения	12
3.2.3. Выбор среды для разработки приложения и БД	14
3.3. Реализация сервиса	16
3.3.1. Стадии и этапы разработки.....	16
3.3.2. Структурная схема сервиса	16
3.3.3. UML-диаграмма структуры БД.....	18
3.3.4. UML-диаграмма деятельности	18
3.3.5. Разработка серверной части	21
3.3.6. UML-диаграммы классов экранов	22
3.3.7. Разработка интерфейса приложения.....	23
3.3.8. Выводы к разделу	24

ВВЕДЕНИЕ

Данная курсовая работа была выполнена в рамках предмета «Разработка клиент-серверных приложений».

Актуальность выбранной темы обусловлена тем, что в современных реалиях бизнесу, предоставляющему физические (материальные) услуги, необходимо иметь Интернет-ресурс, с помощью которого возможно будет производить взаимодействие с клиентами онлайн. Сейчас, когда люди проводят все больше и больше времени в интернете, можно с уверенностью сказать, что собственный сайт непременно принесет пользу его владельцу. Более того, значение интернет-технологий в жизни людей возросло настолько, что создание сайта является необходимым условием для развития бизнеса. Несомненно, компания, имеющая корпоративный сайт, с полным основанием может быть причислена к лидерам, которые используют для развития своего бизнеса современные технологии, и, соответственно, может предложить потенциальным клиентам товары или услуги самого высокого качества. Веб-сайт, представляющий бизнес в Интернете, поможет создать и укрепить репутацию надежной, стабильной организации, идущей в ногу со временем, привлечь новых клиентов, найти инвесторов или деловых партнеров с помощью собственного сайта будет намного проще.

Основной целью курсовой работы является формирование и закрепление компетенций путем практического использования знаний, умений и навыков, полученных в рамках теоретического обучения, а также выработка самостоятельного творческого подхода к решению конкретных профессиональных задач на примере разработки и создания клиент-серверного приложения.

Курсовая работа содержит подробное обоснование выбранной темы, описание процесса разработки, включающее в себя анализ предметной области разрабатываемого сервиса, методологию разработки и описание реализации сервиса.

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Обозначение и наименование программы

Наименование программы: Веб-приложение «Каршеринг»;

Обозначение: CityDrive;

Полное наименование: клиент-серверное фуллстек веб-приложение «Каршеринг».

1.2. Программное обеспечение, необходимое для функционирования программы

ОС Windows, веб-браузер.

1.3. Языки программирования, на которых написана программа

Веб-приложение было написано с помощью языков программирования и разметки Java, HTML, CSS, JavaScript, SQL.

2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

С помощью разработанного веб-приложения клиенты данного каршеринга могут ознакомиться с предоставляемыми услугами, зарегистрироваться и авторизоваться, что позволит бронировать автомобили и в дальнейшем отслеживать статус текущей и видеть историю предыдущих поездок в личном кабинете. Для администраторов ресурса доступна страница со всеми заказами всех пользователей с возможностью поиска по ключевым словам.

3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1. Анализ предметной области

3.1.1. Описание предметной области

Предметной областью для данной курсовой работы является исследование веб-сервисов по теме «Каршеринг».

Каршеринг представляет собой поминутную аренду авто. Данная услуга пользуется большой популярностью во многих крупных городах, так как является выгодным вариантом в сравнении с использованием такси или другого общественного транспорта.

Владельцы личного автомобиля постоянно сталкиваются с такими проблемами, как прохождение техосмотра, мойка, заправка транспорта. Каршеринг избавляет водителей от всех этих действий. В стоимость аренды уже включена мойка и химчистка салона, топливо, страховка и другие услуги. К плюсам каршеринга можно отнести следующее:

- Отсутствие затрат на содержание автомобиля.
- Выбор модели транспортного средства на любой вкус.
- Возможность бесплатного использования парковок и некоторых участков платных трасс.
- Доступная стоимость услуги.

Возможность свободно передвигаться по городу

3.1.2. Анализ аналогов разрабатываемого приложения

Как видно из рисунков 3.1.2.1 – 3.1.2.2, на большинстве сайтов-аналогов не реализована система личного кабинета и, следовательно, функционал данных ресурса сильно ограничен.

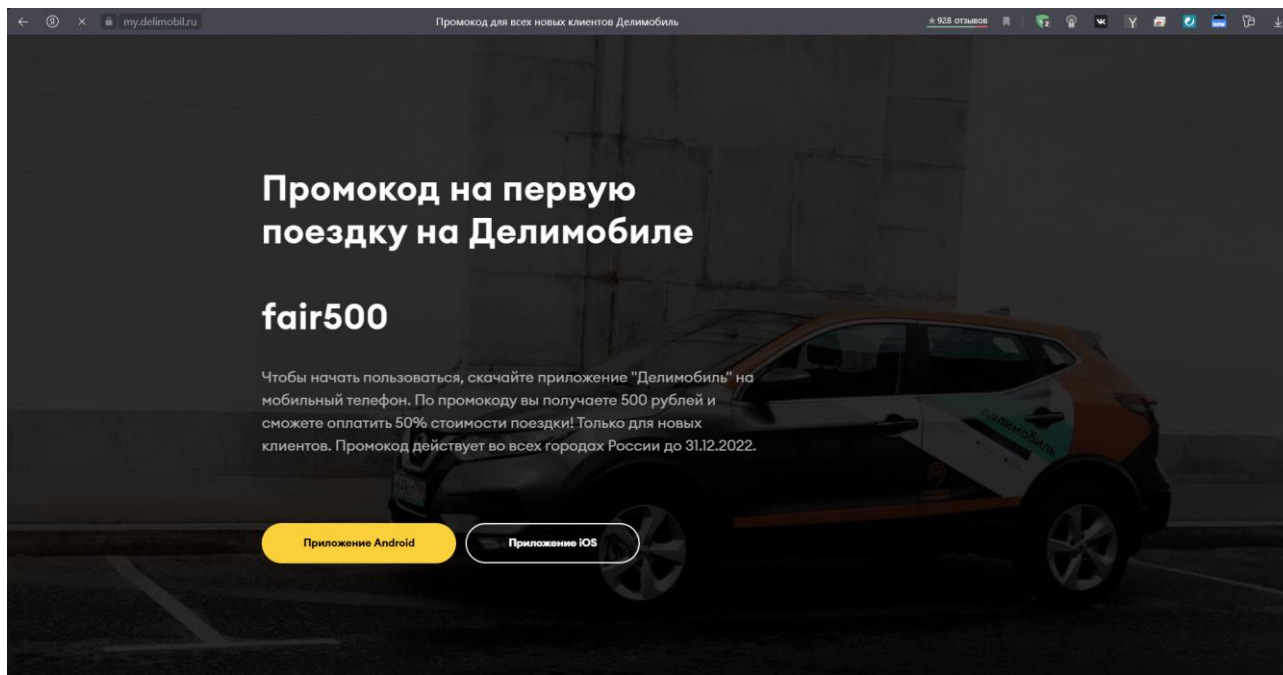


Рисунок 3.1.2.1 – Интерфейс онлайн-ресурса «Делимобиль»

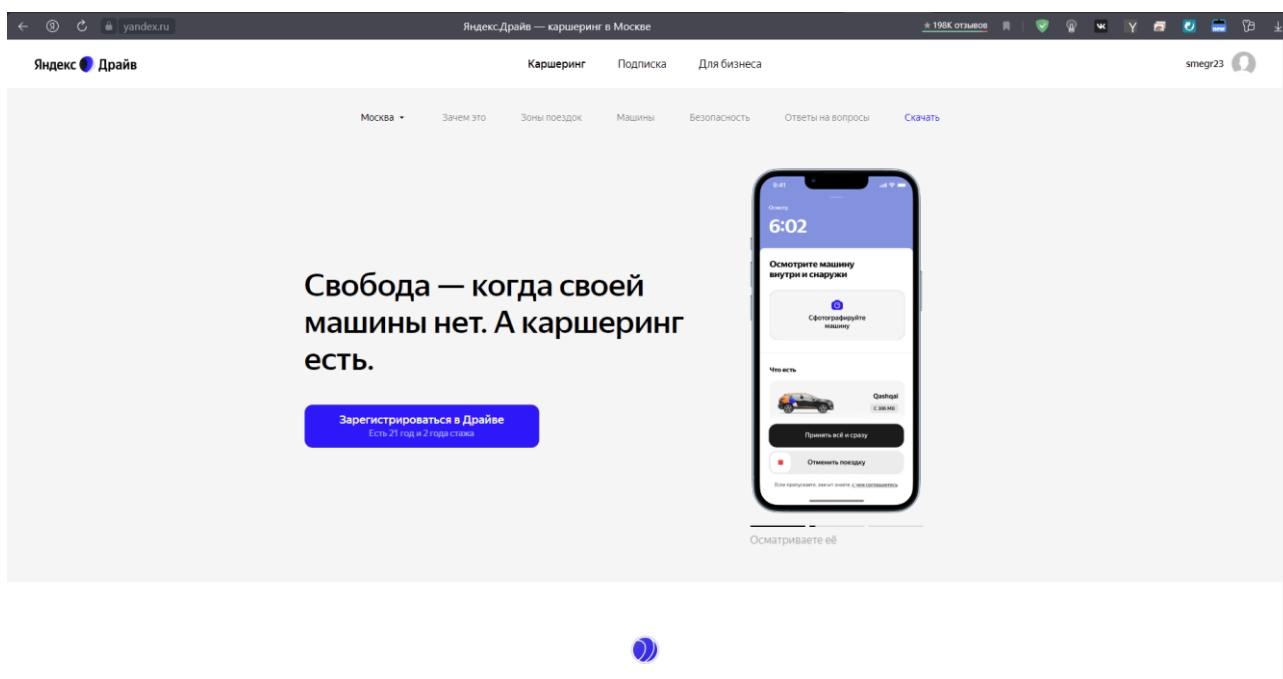


Рисунок 3.1.2.2 – Интерфейс онлайн-ресурса «Яндекс Драйв»

3.2. Методология разработки сервиса

3.2.1. Выбор языка разработки

Основным языком для разработки серверной части приложения был выбран Java, ведь на сегодняшний момент язык Java является одним из самых распространенных и популярных языков программирования. Первая версия языка появилась еще в 1996 году в недрах компании Sun Microsystems, впоследствии поглощенной компанией Oracle. Java задумывался как универсальный язык программирования, который можно применять для различного рода задач. И к настоящему времени язык Java проделал большой путь, было издано множество различных версий. Текущей версией является Java 16, которая вышла в марте 2021 года. А Java превратилась из просто универсального языка в целую платформу и экосистему, которая объединяет различные технологии, используемые для целого ряда задач: от создания десктопных приложений до написания крупных веб-порталов и сервисов. Кроме того, язык Java активно применяется для создания программного обеспечения для множества устройств: обычных ПК, планшетов, смартфонов и мобильных телефонов и даже бытовой техники. Достаточно вспомнить популярность мобильной ОС Android, большинство программ для которой пишутся именно на Java.

Также был использован фреймворк Spring Framework.

Spring Framework – это платформа Java с открытым исходным кодом. Первоначально он был написан Родом Джонсоном и впервые выпущен под лицензией Apache 2.0 в июне 2003 года. Spring является самой популярной платформой разработки приложений для корпоративной Java. Миллионы разработчиков по всему миру используют Spring Framework для создания высокопроизводительного, легко тестируемого и многоразового кода.

Spring очень легковесный, когда дело доходит до размера и прозрачности. (размер базовой версии Spring framework составляет около 2 МБ). [1]

Apache Maven — фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM, являющемся подмножеством XML [2]. Проект Maven издаётся сообществом Apache Software Foundation, где формально является частью Jakarta Project.

Название системы является словом из языка идиш, смысл которого можно примерно выразить как «собиратель знания».

Maven обеспечивает декларативную, а не императивную (в отличие от средства автоматизации сборки Apache Ant) сборку проекта. В файлах описания проекта содержится его спецификация, а не отдельные команды выполнения. Все задачи по обработке файлов, описанные в спецификации, Maven выполняет посредством их обработки последовательностью встроенных и внешних плагинов.

Maven используется для построения и управления проектами, написанными на Java, C#, Ruby, Scala, и других языках.

3.2.2. Выбор инструментов для разработки интерфейса приложения

Для разработки интерфейса приложения были выбраны языки разметки HTML, CSS и JavaScript-библиотека React.

React — JavaScript-библиотека с открытым исходным кодом для разработки пользовательских интерфейсов.

React может использоваться для разработки одностраничных и мобильных приложений. Его цель — предоставить высокую скорость, простоту и масштабируемость. В качестве библиотеки для разработки пользовательских интерфейсов React часто используется с другими библиотеками, такими как MobX, Redux и GraphQL [3].

HTML — стандартизированный язык разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее

интерпретируют код в интерфейс, который будет отображаться на экране монитора.

Элементы HTML являются строительными блоками HTML страниц. С помощью HTML разные конструкции, изображения и другие объекты, такие как интерактивная веб-форма, могут быть встроены в отображаемую страницу. HTML предоставляет средства для создания заголовков, абзацев, списков, ссылок, цитат и других элементов. Элементы HTML выделяются тегами, записанными с использованием угловых скобок. Такие теги, как `` и `<input />`, напрямую вводят контент на страницу. Другие теги, такие как `<p>`, окружают и оформляют текст внутри себя и могут включать другие теги в качестве подэлементов. Браузеры не отображают HTML-теги, но используют их для интерпретации содержимого страницы.

Язык XHTML является более строгим вариантом HTML, он следует синтаксису XML и является приложением языка XML в области разметки гипертекста.

В HTML можно встроить программный код на языке программирования JavaScript, для управления поведением и содержанием веб-страниц. Также включение CSS в HTML описывает внешний вид и макет страницы [4].

CSS — формальный язык описания внешнего вида документа (веб-страницы), написанного с использованием языка разметки (чаще всего HTML или XHTML). Также может применяться к любым XML-документам, например, к SVG или XUL.

CSS используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS является ограждение и отделение 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 [5].

3.2.3. Выбор среды для разработки приложения и БД

Средой разработки серверной части приложения была выбрана среда разработки IntelliJ IDEA. IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains.

Первая версия появилась в январе 2001 года и быстро приобрела популярность как первая среда для Java с широким набором интегрированных инструментов для рефакторинга, которые позволяли программистам быстро реорганизовывать исходные тексты программ. Дизайн среды ориентирован на продуктивность работы программистов, позволяя сконцентрироваться на функциональных задачах, в то время как IntelliJ IDEA берёт на себя выполнение рутинных операций.

Начиная с шестой версии продукта IntelliJ IDEA предоставляет интегрированный инструментарий для разработки графического пользовательского интерфейса. Среди прочих возможностей, среда хорошо совместима со многими популярными свободными инструментами разработчиков, такими как CVS, Subversion, Apache Ant, Maven и JUnit.

Средой разработки клиентской части приложения была выбрана среда разработки WebStorm - интегрированная среда разработки на JavaScript, CSS & HTML от компании JetBrains, разработанная на основе платформы IntelliJ IDEA.

WebStorm обеспечивает авто дополнение, анализ кода на лету, навигацию по коду, рефакторинг, отладку, и интеграцию с системами управления версиями. Важным преимуществом интегрированной среды разработки WebStorm является работа с проектами (в том числе, рефакторинг кода JavaScript, находящегося в разных файлах и папках проекта, а также вложенного в HTML). Поддерживается

множественная вложенность (когда в документ на HTML вложен скрипт на Javascript, в который вложен другой код HTML, внутри которого вложен Javascript) — то есть в таких конструкциях поддерживается корректный рефакторинг.

Системой управления базами данных была использована PostgreSQL.

PostgreSQL — это объектно-реляционная система управления базами данных (ОРСУБД, ORDBMS), 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.

3.3. Реализация сервиса

3.3.1. Стадии и этапы разработки

Процесс реализации данного сервиса можно разделить на две основные стадии:

- Этап анализа и проектирования

На данном этапе были разработаны пользовательские сценарии, спроектирована архитектура и логика работы системы, а также определена общая стилистика проекта и был подготовлен контент.

- Этап разработки

Далее, основываясь на результатах первого этапа, была произведена разработка серверной и клиентских частей.

3.3.2. Структурная схема сервиса

При разработке сервиса был выбран шаблон проектирования MVC, так как он удовлетворяет всем требованиям проекта [6].

MVC — это паттерн проектирования веб-приложений, который включает в себя несколько более мелких шаблонов. При использовании MVC на три отдельных компонента разделены модель данных приложения, пользовательский интерфейс и логика взаимодействия пользователя с системой, благодаря чему модификация одного из этих компонентов оказывает минимальное воздействие на остальные или не оказывает его вовсе.

Концепция MVC разделяет данные, представление и обработку действий пользователя на компоненты:

Модель / Model — предоставляет собой объектную модель некой предметной области, включает в себя данные и методы работы с этими данными, реагирует на запросы из контроллера, возвращая данные и/или изменяя своё состояние. При этом модель не содержит в себе информации о способах визуализации данных или форматах их представления, а также не взаимодействует с пользователем напрямую.

Представление / View — отвечает за отображение информации (визуализацию). Одни и те же данные могут представляться различными способами и в различных форматах. Например, коллекцию объектов при помощи разных представлений можно представить на уровне пользовательского интерфейса как в табличном виде, так и списком; на уровне API можно экспортировать данные как в JSON, так в XML или XSLX.

Контроллер / Controller — обеспечивает связь между пользователем и системой, использует модель и представление для реализации необходимой реакции на действия пользователя. Как правило, на уровне контроллера осуществляется фильтрация полученных данных и авторизация — проверяются права пользователя на выполнение действий или получение информации.

Структура сервиса состоит из клиентской части, которая представлена интерфейсом приложения, серверной части и базой данных. Между клиентской и серверной частью данные передаются с помощью HTTP-запросов, обрабатываются и сохраняются в базу данных. Впоследствии сохраненные данные можно отобразить на клиентской части.

Структурная схема сервиса представлена на рисунке 3.3.2.1.

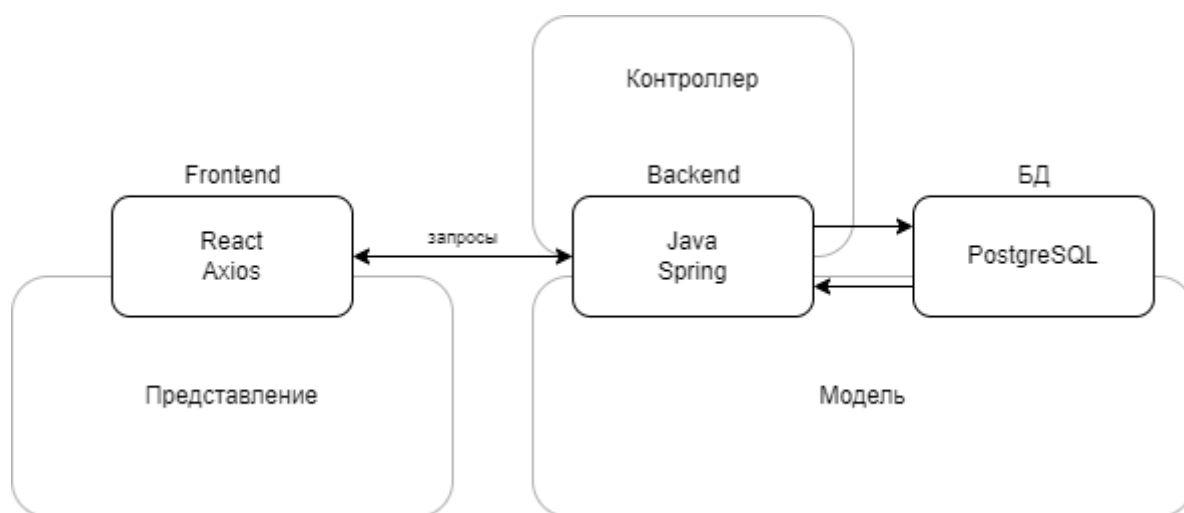


Рисунок 3.3.2.1 – Структурная схема сервиса

3.3.3. UML-диаграмма структуры БД

Модель базы данных приложения представлена на рисунке 3.3.3.1.

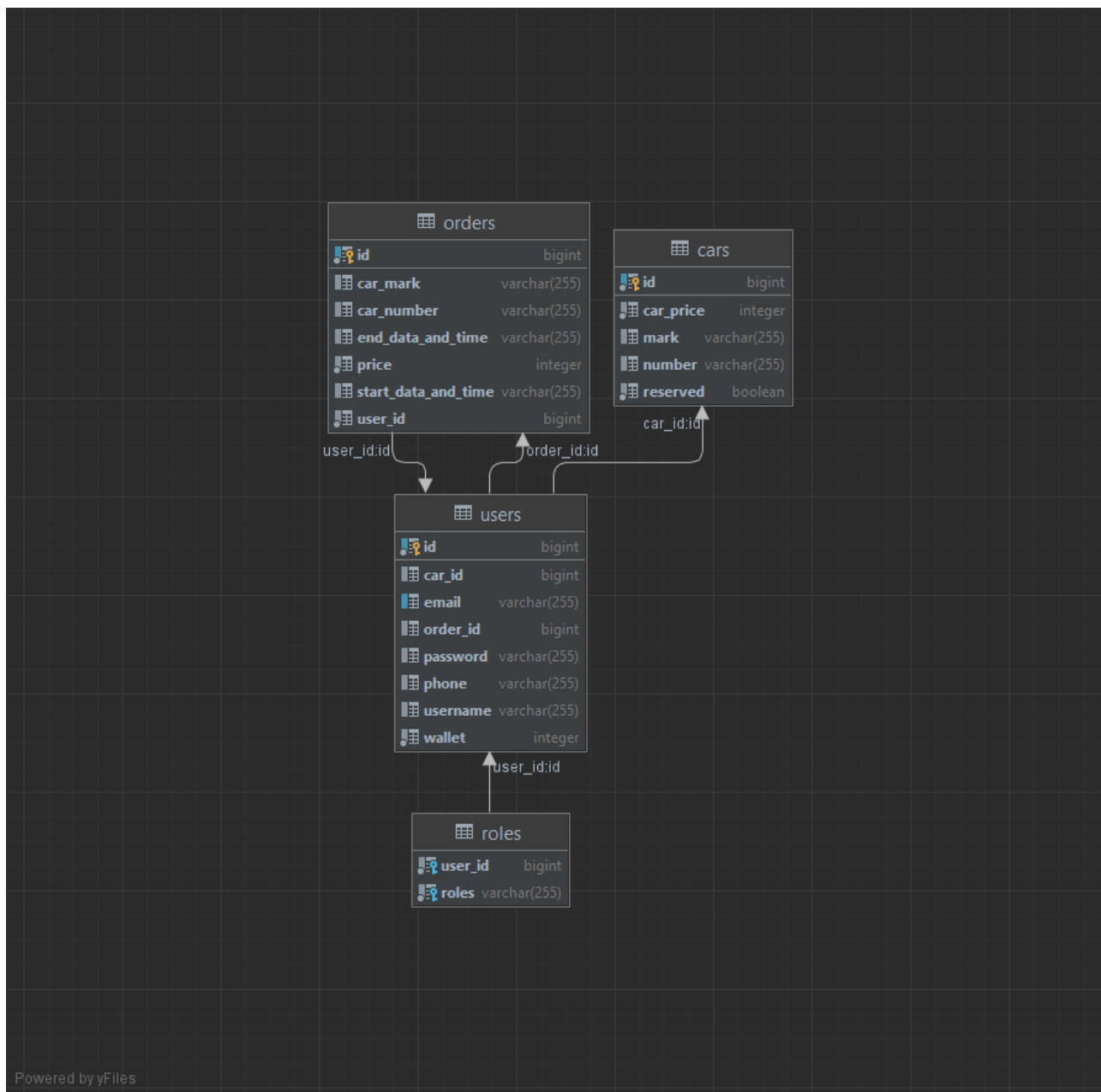


Рисунок 3.3.3.1 – UML-диаграмма структуры БД

3.3.4. UML-диаграмма деятельности

Были составлены диаграммы деятельности для процесса регистрации пользователя и процесса создания пользователем заявки, представленные на рисунках 3.3.4.1 и 3.3.4.2.

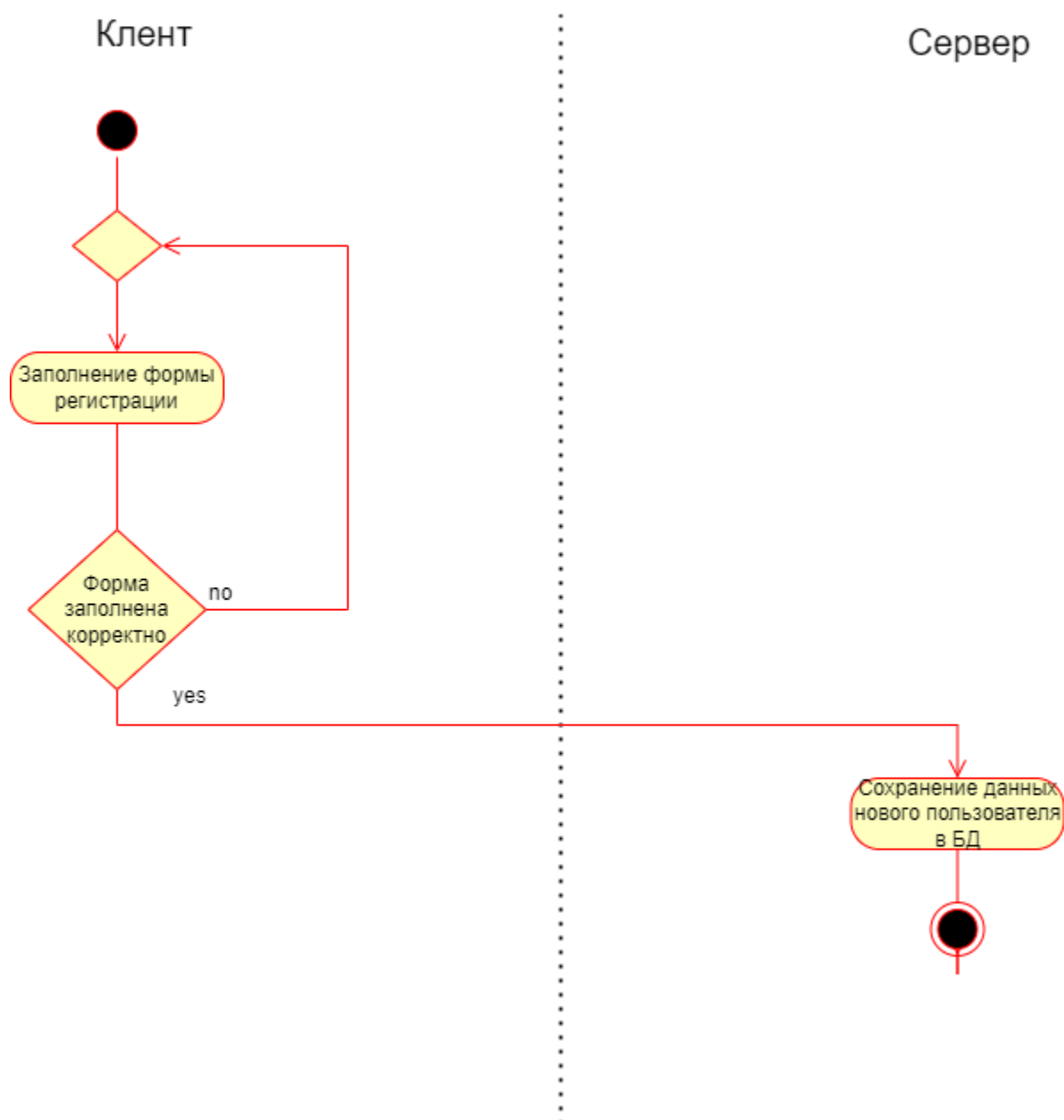


Рисунок 3.3.4.1 – Диаграмма деятельности регистрации пользователя

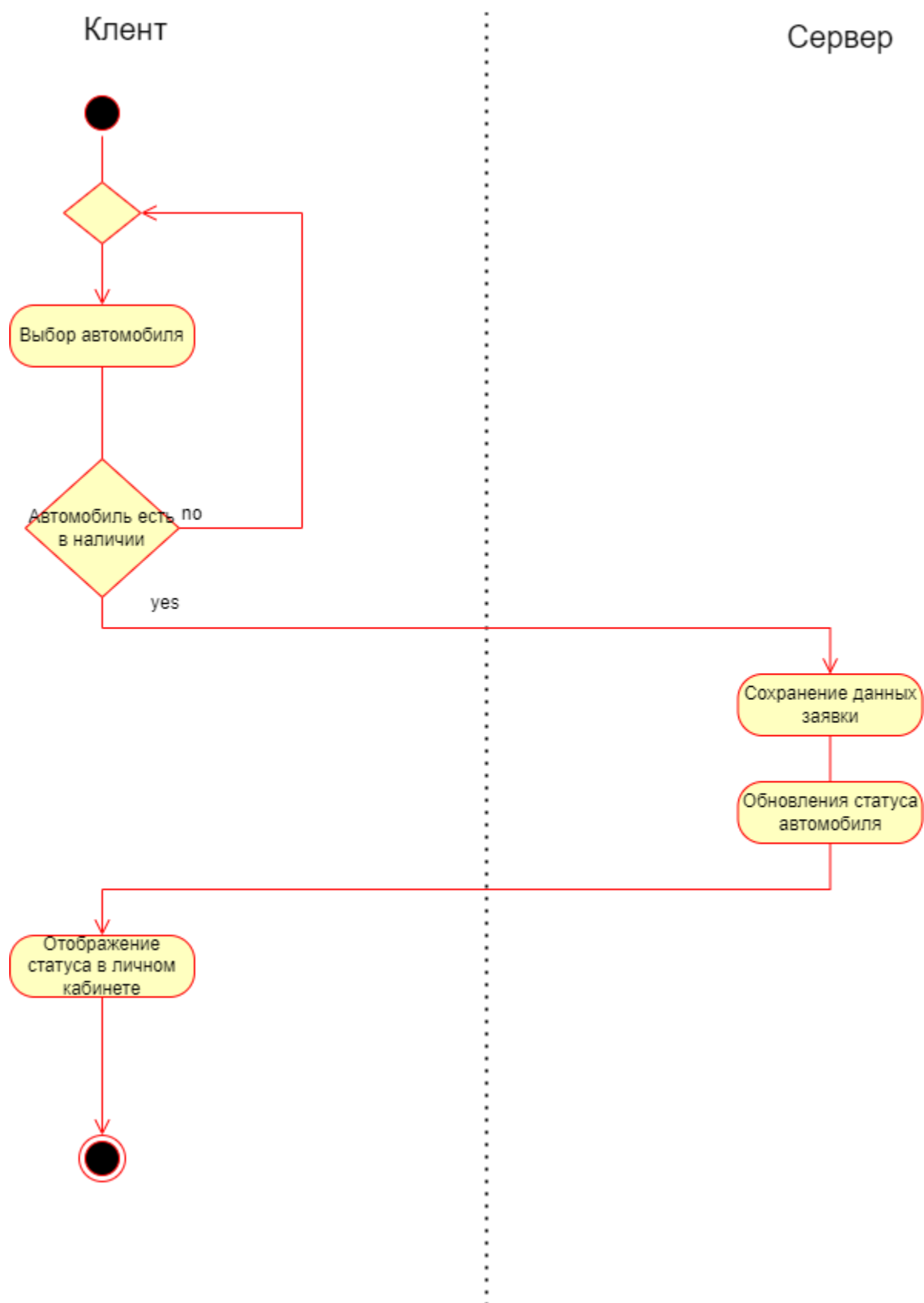


Рисунок 3.3.4.2 – Диаграмма деятельности процесса создания заявки

3.3.5. Разработка серверной части

Серверная часть приложения выполнена с помощью фреймворка Java Spring Framework, который позволяет быстро и удобно создать основу для будущего Java приложения.

В качестве системы управления базами данных был выбран PostgreSQL [7][8].

Для того, чтобы реализовать регистрацию и авторизацию в веб-приложении, была создана сущность User, которая имплементирует класс UserDetails, а сама база данных включает в себя таблицу “users” с уникальным идентификатором пользователя (id), именем пользователя (username), зашифрованным паролем пользователя (password), настоящим именем человека (name), а также телефоном пользователя (phone) и Email (email). Запись в базу данных происходит при регистрации пользователя после того, как тот введет все данные и нажмет кнопку “REGISTRATION”, чтение из базы данных происходит при авторизации пользователя, когда он вводит данные из регистрации для входа и нажимает кнопку “LOGIN”.

Также была добавлена таблица “roles”, в которой каждому пользователю по id присваивается роль – либо USER, либо ADMIN. Отличаются они уровнем доступа и возможностями – пользователям с ролью USER доступен личный кабинет, история заказов и возможность создания новых заявок, а пользователи с ролью ADMIN могут управлять данными пользователя, а также видеть всю информацию о пользователе, сделавшем заказ.

При успешной авторизации зарегистрированного ранее пользователя в личном кабинете появляется возможность ознакомиться с историей своих заказов и их статусом, а также изменения личных данных, таких как номер телефона, email и т.д.

Для хранения пользовательских заявок была добавлена таблица “orders”, в которой хранятся все отправленные заявки от всех пользователей. В эту таблицу включен уникальный идентификатор заказа (id), марка авто пользователя

(carMark), номер авто (carNumber), даты начала и конца бронирования авто (startDataAndTime и endDataAndTime соответственно), идентификатор пользователя данной заявки (userId) и стоимость (price).

Для хранения данных об автомобилях была добавлена таблица “cars”, в которой хранятся данные обо всех авто, имеющих в автопарке, а именно марка автомобиля (mark), его номер (numder), уникальный идентификатор (id), стоимость за аренду в минуту (carPrice) и статус занятости (reserved)

3.3.6. UML-диаграммы классов экранов

Диаграмма всех классов разрабатываемого приложения представлена на рисунке 3.3.6.1

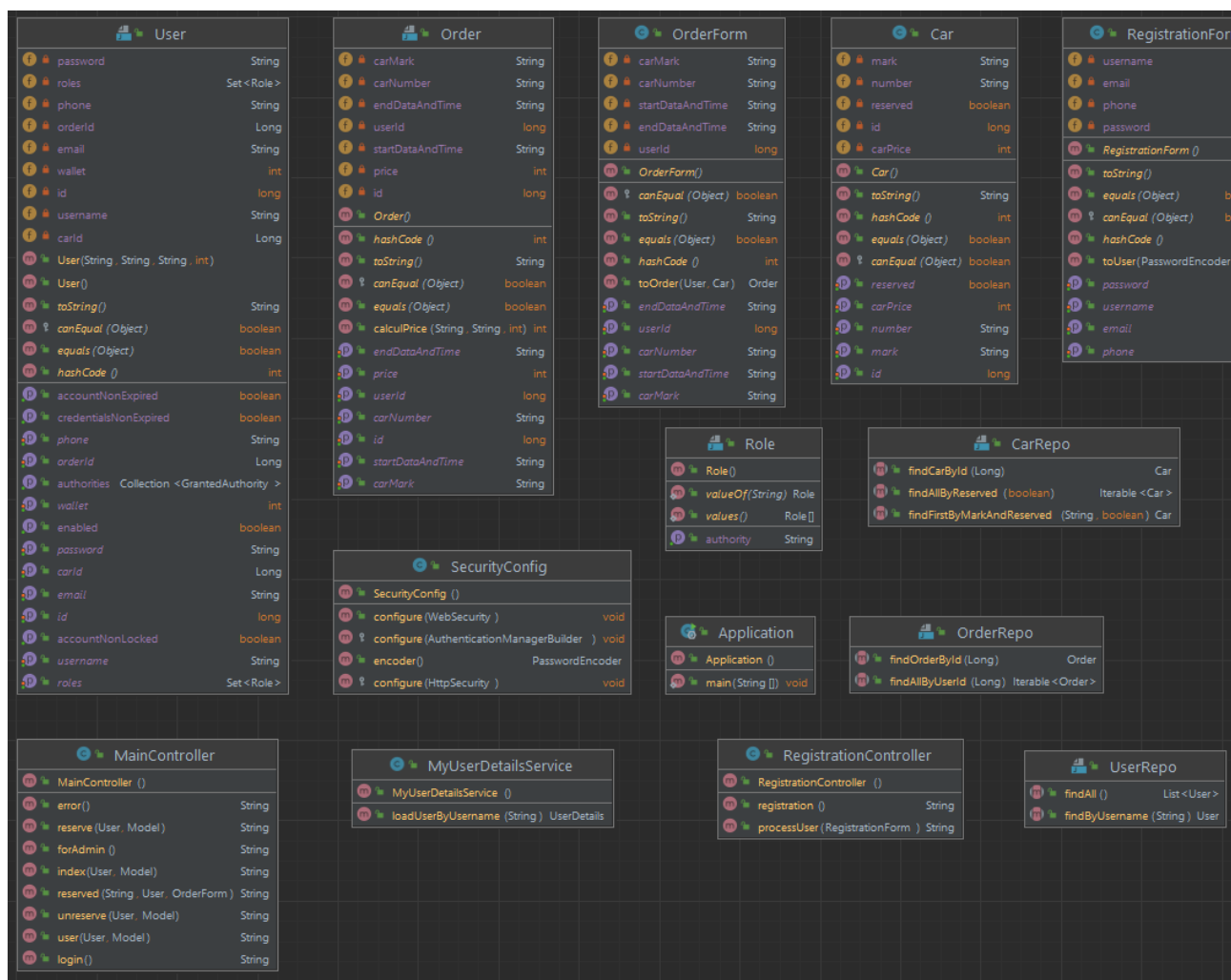


Рисунок 3.3.6.1 – Диаграмма классов приложения

3.3.7. Разработка интерфейса приложения

Интерфейс клиентской части приложения был разработан с помощью React и состоит из следующих страниц:

- 1) Главная страница;
- 2) Страница с авторизацией пользователя;
- 3) Страница с регистрацией новых пользователей;
- 4) Страница с личным кабинетом пользователя;
- 5) Страница с представленными автомобилями и тарифами на их аренду;
- 6) Страница администратора со списком всех пользователей и заявок (доступна только пользователям ADMIN);
- 7) Страница администратора с информацией о пользователе и его заказах (доступна только пользователям ADMIN).

На рисунке 3.3.7.1 представлена главная страница.



Рисунок 3.3.7.1 – Главная страница ресурса

На рисунке 3.3.7.2 представлена предварительная версия страницы регистрации новых пользователей.

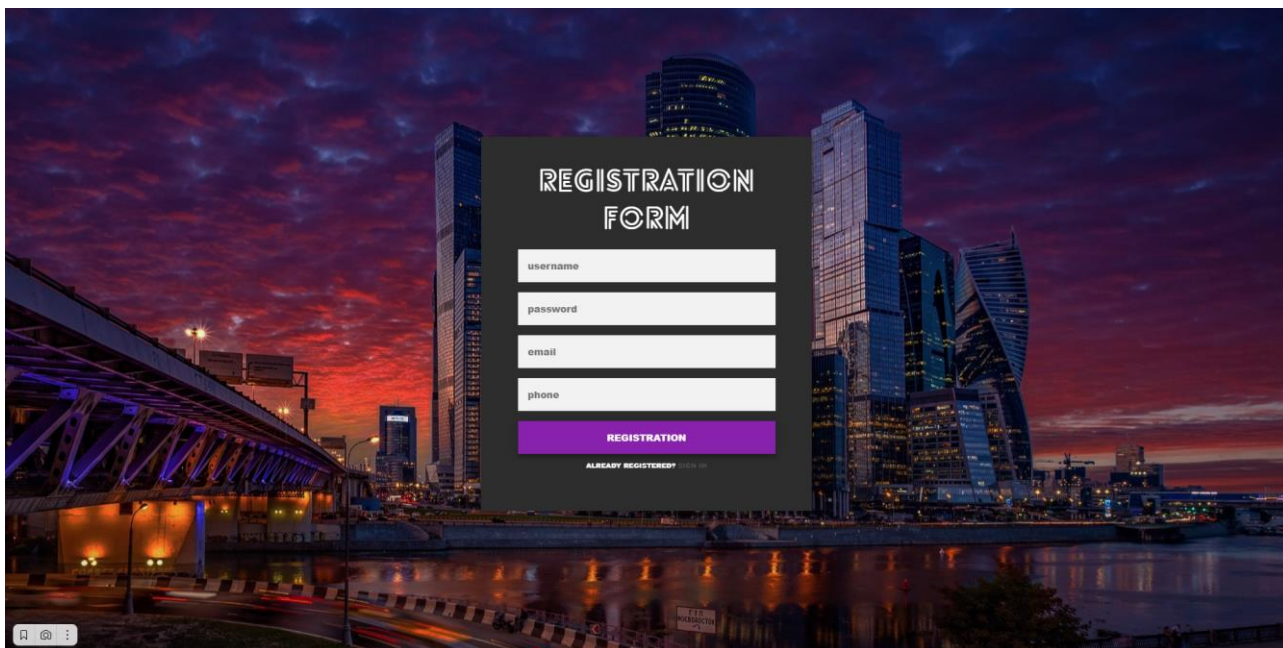


Рисунок 3.3.7.2 – Страница с регистрацией новых пользователей.

3.3.8. Выводы к разделу

В ходе выполнения работы было реализовано полноценное веб-приложение, содержащее информацию об автосервисе, систему авторизации и регистрации, систему обновления базы данных заказов и страницы менеджмента пользователей. Также были разграничены роли обычных пользователей и администратора, а также администратору был предоставлен доступ к странице менеджмента пользователей и их заказов.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы были получены навыки создания веб-приложений при помощи языка программирования Java и фреймворка Spring Framework на основе системы автоматической сборки Maven, а также JavaScript-библиотеки React для разработки интерфейса. Были сформированы и закреплены компетенции путем практического использования знаний, умений и навыков, полученных в рамках теоретического обучения, а также выработан самостоятельный подход к решению конкретных профессиональных задач на примере разработки и создания клиент-серверного приложения.

1. Создано клиент-серверное веб-приложение с применением технологий React, а также Spring Data Jpa, и Spring MVC, входящих в состав Springи системы сборки Maven.

2. Реализована серверная часть приложения, включающая систему регистрации и авторизации и отправку заказов в базу данных.

3. Протестирована работоспособность веб-приложения.

Все условия и задачи курсовой работы были выполнены в полном объеме и корректно работают согласно требованиям.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Spring Framework – обзор – URL:
<https://coderlessons.com/tutorials/java-tehnologii/uchis-vesne/spring-framework-obzor#:~:text=Spring%20%E2%80%93%20%D1%81%D0%B0%D0%BC%D0%B0%D1%8F%20%D0%BF%D0%BE%D0%BF%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%B0%D1%8F%20%D1%81%D1%80%D0%B5%D0%B4%D0%B0%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8,Java%20%D1%81%20%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%BC%20%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%BC%20%D0%BA%D0%BE%D0%B4%D0%BE%D0%BC>
2. Краткое знакомство с Maven – URL: <https://tproger.ru/articles/maven-short-intro/>
3. React – URL: <https://ru.wikipedia.org/wiki/React>
4. HTML – URL: <https://ru.wikipedia.org/wiki/HTML>
5. CSS – URL: <https://ru.wikipedia.org/wiki/CSS>
6. MVC — модель-представление-контроллер – URL: <https://web-creator.ru/articles/mvc>
7. Документация PostgreSQL – URL: <https://www.postgresql.org/docs>
8. Добавляем БД PostgreSQL к RESTful сервису на Spring Boot. Часть 1 – URL: <https://javarush.ru/groups/posts/2579-dobavljajem-bd-k-restful-servisu-na-spring-boot-chastjh-1>

ПРИЛОЖЕНИЕ

1. Исходный код разработанного веб-приложения доступен по ссылке:
<https://github.com/RoNiN-23/JavaReact.git>