Отчет о проверке № 8065712

Дата выгрузки: 2023-05-21 13:16:36
Пользователь: sidorovstasd7@gmail.com, ID: 8065712

Отчет предоставлен сервисом «Антиплагиат»
на сайте antiplagius.ru/

Информация о документе

№ документа: 8065712
Имя исходного файла: ИКБО-20-21_КРШППЯД_Сидоров_С_Д.pdf
Размер файла: 1.12 МБ
Размер текста: 29003
Слов в тексте: 3530
Число предложений: 295

Информация об отчете

Дата: 2023-05-21 13:16:36 - Последний готовый отчет
Оценка оригинальности: 99%
Заимствования: 1%

99.06% 0.94%

Источники:

Доля в тексте	Ссылка
7.50%	https://www.lambdatest.com/blog/spring-testing/

Информация о документе:

МИНОБРНАУКИ РОССИИ Федеральное государственное бюджетное образовательное учреждение высшего образования "МИРЭА - Российский технологический университет" РТУМИРЭА Институт информационных технологий (ИТ) Кафедра инструментального и прикладного программного обеспечения (ИиППО) КУРСОВАЯ РАБОТА по дисциплине: Шаблоны программных платформ языка Джава по профилю: Разработка программных продуктов и проектирование информационных систем направления профессиональной подготовки: 09.03.04 "Программная инженерия" Тема: Приложение "Каршеринговая компания" Студент: Сидоров Станислав Дмитриевич Группа: ИКБО-20-21 Работа представлена к защите_____(дата)_____/Сидоров С.Д./ Руководитель: старший преподаватель Зорина Наталья Валентиновна Работа допущена к защите_____(дата)_____/Зорина Н.В./ Оценка по итогам защиты: _____ _____ / _____ / _____ / _____ / (подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших защиту) М. РТУ МИРЭА. 2022 г. УДК 4.4 Сидоров С.Д. Приложение "Каршеринговая компания"/ Курсовая работа по дисциплине "Шаблоны программных платформ языка Джава" профиля "Разработка программных продуктов и проектирование информационных систем" направления профессиональной подготовки бакалавриата 09.03.04 "Программная инженерия" (2-ый семестр) / руководитель старший преподаватель Н.В. Зорина / кафедра ИиППО Института ИТ МИРЭА - с. 37, табл. 0, ист. 7 Целью работы является создание серверного программного приложения на тему "Каршеринговая компания". В рамках работы осуществлен краткий анализ аналогов веб-приложения по выбранной тематики, разработано приложение с использованием фреймворка Spring, произведено тестирование приложения и проверка на антиплагиат. Sidorov S.D. Application "Car-sharing company"/ Coursework on the subject of "Java Platform Programming Patterns" in the profile of "Software Product Development and Information Systems Design" of the Bachelor's degree program in "Software Engineering" (2nd semester). The paper is 37 pages long, contains 0 tables, and 7 sources, and was supervised by Senior Lecturer N.V. Zorina from the Department of Computer Science and Engineering at the Institute of Information Technology at MIEM. The purpose of the work is to create a server-side software application on the topic of "Carsharing company". The work includes a brief analysis of web application analogs on the chosen topic, the development of an application using the Spring framework, testing of the application, and checking for plagiarism. М. МИРЭА. Ин-т ИИ. Каф. ИиППО. 2023 г. Сидоров С.Д. Аннотация В курсовой работе описывалось создание интернет-ресурса, на тему "Каршеринговая компания". Работа содержит анализ предметной области разрабатываемого интернет-

ресурса, создание веб-страниц интернет-ресурса с использованием технологий Spring Framework и тестирование разработанного приложения. В введении обосновывается актуальность выбранной темы, определяется цель работы и задачи, подлежащие решению для её достижения, описываются объект и предмет исследования используемые методы и информационная база исследования, а также кратко характеризуется структура КР по разделам. В основной части содержится материал, необходимый для достижения цели КР. Основная часть включает в себя общие сведения (в частности, наименование интернет-ресурса, перечисление прикладного программного обеспечения, необходимого для разработки и функционирования интернетресурса, а также названия языков и технологий, с помощью которых реализован интернет-ресурс), описание функционального назначения интернет-ресурса и его логической структуры, описание разработки и функций программного приложения, тестирование работы приложения. В заключении последовательно излагаются теоретические выводы, которые были сформулированы в результате выполнения данной курсовой работы. Курсовая работа на 37 листах, содержит 14 рисунков, 7 использованных источников, 6 листингов. The term paper describes the creation of an internet resource on the topic of "Car-sharing company". The work includes an analysis of the subject area of the developed internet resource, the creation of web pages of the internet resource using Spring Framework technologies, and testing of the developed application. The introduction justifies the relevance of the chosen topic, defines the goal of the work and the tasks that need to be solved to achieve it, describes the object and subject of the research, the methods and information base of the research, and briefly characterizes the structure of the course paper by sections. The main part contains the material necessary to achieve the goal of the course paper. The main part includes general information (in particular, the name of the internet resource, a list of application software necessary for the development and operation of the internet resource, as well as the names of languages and technologies used to implement the internet resource), a description of the functional purpose of the internet resource and its logical structure, a description of the development and functions of the software application, and testing of the application's operation. The conclusion presents the theoretical conclusions that were formulated as a result of the completion of this course paper. The course paper is 37 pages long and contains 14 figures, 7 used sources, and 6 listings. СОДЕРЖАНИЕ Обозначения и сокращения.....7 Введение..... 8 1. СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ВЕБ-ПРИЛОЖЕНИЮ..... 9 1.1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ.....9 ВЫВОДЫ К РАЗДЕЛУ 1.....9 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА.....11 2.1 Проектирование веб-приложения..... 11 2.2 Выбор средств и технологии ведения разработки.....14 2.3 Структура программного приложения..... 15 2.4 Описание классов программного приложения..... 18 Выводы к разделу 2.....28 3 ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ.....29 3.1 Тестирование регистрации и авторизации..... 29 3.2 Тестирование пользовательского функционала..... 30 3.3 Тестирование функций администратора..... 32 Выводы к разделу 3.....35 ЗАКЛЮЧЕНИЕ..... 36 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ ..... 37 Обозначения и сокращения БД - база данных; СУБД - система управления базами данных; URL - унифицированный указатель ресурса; http - протокол передачи данных. Введение В настоящее время темп жизни среднестатистического человека значительно увеличился по сравнению с предыдущими столетиями. Люди стали все больше ценить своё личное время и комфорт. Данный факт повлиял на обширное распространение различных сервисов, существование которых ранее невозможно было даже представить. В наше время стали доступны различные приложения для доставки продуктов и товаров, для аренды квартир и техники, а также множество других онлайн сервисов улучшающих повседневную жизнь и позволяющих тратить все меньше времени на закрытие бытовых потребностей. Одним из них является каршеринг. Каршеринг позволяет пользователям арендовать различные транспортные средства за поминутную оплату, что позволяет людям совершать повседневные поездки по цене такси с комфортом личного автомобиля. Целью данной курсовой работы является разработка приложения "Каршеринговая компания" с использованием Spring Framework, JDK и IntelliJIDEA. Для упрощения разработки процесс был поделён на несколько частей: 1. Анализ предметной области разрабатываемого веб-приложения; 2. Выбор средств ведения разработки; 3. Разработать веб-приложение с использованием IntelliJ IDEA, GitHub, JDK, Spring Framework, JEE; 4. Провести тестирование приложения. В результате приложение должно обладать функционалом, необходимым для управления элементами каршеринговой компании, а также логикой функционирования, соответствующей современным стандартам разработки приложений. 1. СБОР И АНАЛИЗ ТРЕБОВАНИЙ К ВЕБ-ПРИЛОЖЕНИЮ 1.1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ В данной курсовой работе предметной областью является исследование веб-приложений на тему "Каршеринговая компания". Каршеринговая компания позволяет пользователям получать доступ к автомобилю на короткий срок за определенную плату, что позволяет людям, не имеющим личный автотранспорт пользоваться удобным средством перемещения, при этом забирая расходы на обслуживание, ремонт, обновление и до оснащения автопарка на себя. На данный момент существует большое количество различных каршеринговых компаний, каждая из которых обладает своими собственными особенностями в реализации базового функционала. В качестве исследуемых аналогов были выбраны одни из самых популярных приложений, предоставляющих услуги каршеринга, такие как: "Яндекс Драйв", "Делимобиль", "BelkaCar". Данные приложения были проанализированы на предмет представляемого функционала. В ходе анализа были выделены определённые функции, присущие каждому из выше представленных сервисов, представляющие собой основу приложения "Каршеринговая компания", которые будут реализованы при дальнейшей разработке. Из них был составлен список услуг необходимы для предоставления услуг каршеринговой компании: 1. Регистрация новых пользователей; 2. Аренда транспортных средств; 3. Просмотр информации о совершенных ранее арендах; ВЫВОДЫ К РАЗДЕЛУ 1 Исходя из вышеперечисленного можно

сделать следующий вывод. Вебприложение должно иметь регистрацию пользователей, старт и окончание аренды, добавление, изменение и удаление информации о транспортных средствах, изменение данных о пользователях, а также предоставлять возможность работникам просматривать список аренд, изменять системные данные, тарифные планы, параметры доступа пользователей и категории транспортных средств. 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА 2.1 Проектирование веб-приложения Для удобного хранения данных была использована СУБД PostgreSQL. Далее была разработана БД (Рисунок 2.1). В БД входит 10 таблиц: 1. users (таблица пользователей). Таблица хранит в себе персональные данные пользователя, роль, указатель на уровень и пароль. Колонка snpassport нужна для уникального идентифицирования пользователя колонка full_name - полное имя пользователя, колонка date_of_birth - дата рождения пользователя, колонка password - зашифрованный пароль пользователя. Колонка ID_level хранит id текущего уровня пользователя. 2. userlevel (таблица уровней). Таблица, предоставляющая список уровней, которыми может обладать пользователь. Колонка ID_level является уникальным идентификатором записи, level_name содержит название уровня. 3. grouplevel (таблица групп). Таблица, предоставляющая список групп, к которым могут быть отнесены различные транспортные средства. Колонка id_group является уникальным идентификатором для каждой записи, group_name содержит название группы. 4. Permission (таблица разрешений) Таблица, содержащая информацию о уровне пользователя необходимом для управления данной группой транспортных средств. Колонка id_permission является уникальным идентификатором для каждой записи, id_group - id группы транспортных средств, id_level - id уровня пользователя необходимого для управления данной группой. 5. Vehicle_model (таблица моделей) Таблица, содержащая информацию о представленных в автопарке моделях транспортных средств. Колонка id_model - уникальный идентификатор каждой модели, колонка model_name содержит название модели. 6. Vehicle_brand (таблица брендов) Таблица, содержащая информацию о представленных в автопарке брендах транспортных средств. Колонка id_model - уникальный идентификатор каждой модели, колонка model_name содержит название модели. 7. Vehicle_name (таблица имен) Таблица, содержащая информацию о комбинациях моделей и брендов у транспортных средств присутствующих в автопарке. Колонка id_vehicle_name - уникальный идентификатор каждой комбинации, колонка id_brand - id бренда транспортного средства, id_model - id модели транспортного средства. 8. Vehicle_work_model (таблица рабочих моделей) Таблица, содержащая информацию о группах транспортных средств одной модели. Колонка id_vehicle_work_model - уникальный идентификатор каждой рабочей модели, колонка price_per_hour - цена транспортного средства в час, колонка model_photo_name - название отображаемой фотографии на странице поиска, колонка id_vehicle_name - id имени транспортного средства, колонка id_group - id группы, к которой принадлежит транспортное средство. 9. Vehicle (таблица транспортных средств) Таблица, содержащая информацию о каждом транспортном средстве автопарка. Колонка VIN - уникальный идентификатор каждого транспортного средства, колонки color, state, place содержат информацию о цвете, статусе и местоположении транспортного средства, колонка id_vehicle_work_model - id рабочей модели транспортного средства. 10. Rent (таблица аренд) Таблица, содержащая информацию о каждой аренде проведенной пользователями. Колонка id_rent содержит уникальный идентификатор каждой аренды. Колонки duration, starting_point, end_point, start_time, end_time содержат основную информацию об аренде такую, как длительность, местоположение старта, местоположение конца, время начала и время конца аренды. Колонки VIN и snpassport, содержат уникальные идентификаторы транспортного средства и пользователя соответственно. Рисунок 2.1 - Схема БД 2.2 Выбор средств и технологии ведения разработки Для разработки в качестве IDE была выбрана Intellij IDEA - интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java и разработана JetBrains. В качестве основного браузера использовался Opera. Для тестирования http запросов был использован Postman - платформа для совместной разработки API. Для работы с БД был использована СУБД PostgreSQL. Основным языком программирования был выбран Java. Также были использованы фреймворки Spring Framework, Spring Security, Spring Boot. Для общения с БД была выбрана библиотека Hibernate и для удобной работы библиотека Lombok - java-библиотека, которая автоматически подключается к вашему редактору и инструментам сборки. Был использован Bootstrap для оформления сайта. А также использована система сборки Maven - фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM, являющемся подмножеством XML. Приложение работает на основе MVC-шаблона (Model, View, Controller). Это схема разделения данных приложения, UI и управляющей логики на три отдельных компонента: модель, представление и контроллер - таким образом, что модификация каждого компонента может осуществляться независимо. 2.3 Структура программного приложения Корневым пакетом программного приложения является ru.mirea.SidorovSD (Рисунок 2.2) и содержит следующие пакеты: 1. controllers - содержит классы контроллеров, предоставляющие данные пользователю. 2. Models - пакет, содержащий классы взаимодействующие с БД. 3. Services - содержит классы сервисов, реализующих внутреннюю бизнес-логику. 4. Repos - содержит интерфейсы организующие взаимодействие с БД. 5. DTO - содержит классы, представляющие данные для отправки или получения из вне. Все файлы конфигурации располагаются в основном пакете. Также папка resources вынесенная за основной пакет, содержит конфигурационный файл application.properties, а также папки static и templates необходимые для функционирования клиентской части. Рисунок 2.2 - Содержимое корневого пакета Рисунок 2.3 - Содержимое папка resources 2.4 Описание классов программного приложения Изначально был сгенерирован пустой проект с помощью Spring Intializr, где были выбраны следующие библиотеки: 1. Spring Web; 2. Spring Data JPA; 3. Lombok; 4. Spring Security; 5. PostgreSQL Driver; 6. Thymeleaf; 7. Spring Boot. Далее было решено начать с настройки взаимодействия между приложением и БД. Для достижения данной цели были созданы класса пакета Models. Каждый класс был помечен аннотацией @Entity для указания того, что конкретный класс является сущностью. Для связывания сущности с определенной таблицей в БД использовалась аннотация @Table с параметром name равным названию таблицы в БД. Также для автоматической генерации геттеров и сеттеров использовалась аннотация @Data фреймворка Lombok. Атрибуты класса были помечены аннотацией @Column, задав параметру name название колонки в БД, что позволило Hibernate присвоить нужное значение атрибуту. В качестве примера сущности был представлен класс User. (Листинг 2.1) Листинг 2.1 - Фрагмент кода сущности User @Entity @Data @Table(name =

"users") public class User { @Id @Column(name = "SNPassport") private String snpassport; @Column(name = "full_name") private String fullname; @Column(name = "date_of_birth") private String date_of_birth; @Column(name = "password") private String password; @Column(name = "username") private String username; @Column(name = "role") private String role; @JoinColumn(name = "id_level", referencedColumnName = "id_level") private int idLevel;}} Для того чтобы получить данные из БД, необходимо создать интерфейс, расширяющий JpaRepository. Это позволяет вызывать нужный метод без необходимости постоянно прописывать SQL запросы. Интерфейсы были размещены в пакете Repos и помечены аннотацией @Repository для обнаружения Hibernate. Пример интерфейса представлен в Листинге 2.2. Листинг 2.2 - Фрагмент кода UserRepository @Repository public interface UserRepo extends JpaRepository { User findBySnpassport(String username); List findByIdLevel(int idLevel); List findByRole(String role); } Далее был создан сервисный слой для реализации бизнес-логики и взаимодействия с данными из БД. Файлы сервисов были помещены в пакет Services и были помечены аннотацией @Service. Пример сервиса изображен в Листинге 2.3. Листинг 2.3 - Фрагмент кода UserService @Service @Slf4j public class UserService { @Autowired private final UserRepo userRepo; @Autowired private final LevelRepo levelRepo; @Autowired private final RentRepo rentRepo; public UserService(UserRepo userRepo, LevelRepo levelRepo, RentRepo rentRepo) { this.userRepo = userRepo; this.levelRepo = levelRepo; this.rentRepo = rentRepo; } public List getAll(){ return userRepo.findAll(); } public List getAllByLevel(int idLevel){ return userRepo.findByIdLevel(idLevel); } public List getAllByRole(String role){ return userRepo.findByRole(role); }Продолжение Листинга 2.3 public User findBySnpassport(String snpassport) { return userRepo.findBySnpassport(snpassport);} public void saveUser(User user){ if (isUserExist(user.getSnpassport())) return null; String encodedPassword = new BCryptPasswordEncoder().encode(user.getPassword()); user.setPassword(encodedPassword); user.setRole("USER"); user.setIdLevel(1); userRepo.save(user);} public String changeUserInfo(String snpassport, String fullname, String dateOfBirth, String