UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Web Applications A.Y. 2023-2024
Homework 1 – Server-side Car Rental System

Master Degree in Computer Engineering
Master Degree in ICT for Internet and Multimedia

Deadline: 29 April, 2024

Group Acronym	Car Rental System	
Last Name	**First Name**	**Badge Number**
Chemello	Francesco	2121346
Dolati	Elnaz	2071516
De Souza Farias	Gabriella Ingridy	2099980
Pellegrini	Luca	2122860
Safavi	Seyedreza	2071558
Sadin	Ahmad	2071555

1 Objectives

The primary objective of our Car Rental web application project is to design and implement a robust and efficient platform aimed at facilitating the management of car rental operations. This endeavor is driven by the overarching goal of providing a comprehensive toolset to streamline daily activities for car rental companies while concurrently enhancing customer experiences within the realm of car rentals.

Specifically, our project aims to achieve the following objectives:

1. Efficient Car Management: Develop a centralized system for tracking and managing a fleet of rental cars, ensuring optimal utilization and maintenance. This objective encompasses tasks such as adding, editing, and removing vehicle listings, updating vehicle information (e.g., availability, pricing, specifications), and implementing alert mechanisms for low stock levels.

2. Enhanced Customer Management: Create a comprehensive platform for managing customer information, reservations, and communication channels. This entails facilitating seamless reservation processes for customers, enabling them to make and modify bookings effortlessly, and providing avenues for direct communication between staff and customers to address inquiries and resolve issues promptly.

3. Streamlined Reservation Handling: Implement an efficient reservation handling system that allows customers to make and modify reservations seamlessly. Staff members should be equipped with tools to review and manage reservations, ensuring efficient allocation of vehicles based on availability and customer preferences.

4. Automated Billing and Invoicing: Develop automated billing processes to generate invoices for reservations and additional services. This includes tracking payments, managing late fees, and generating financial reports for analysis and reconciliation.

5. Effective Inventory Management: Establish a centralized system for monitoring vehicle availability and generating alerts for low stock levels. The objective is to ensure that the right cars are available at the right time and location to meet customer demand, thereby optimizing inventory control and maximizing revenue potential.

6. Comprehensive Reporting and Analytics: Provide comprehensive reports and analytics on various aspects of the rental business, including reservation trends, revenue generation, and customer demographics. These insights will empower data-driven decision-making, enabling the company to refine strategies and drive business growth effectively.

7. Security and Access Control: Implement robust security measures, including role-based access control mechanisms, to safeguard sensitive functionalities and data. Ensuring authorized personnel have access to specific features will enhance overall system security and protect against unauthorized access or misuse.

By accomplishing these objectives, our Car Rental web application endeavors to significantly enhance operational efficiency, elevate customer satisfaction levels, and position the company for long-term success and competitiveness in the dynamic car rental industry landscape.

2 Main Functionalities

The Car Rental web application boasts a comprehensive suite of functionalities meticulously designed to optimize rental operations and elevate customer experiences. These functionalities are structured into distinct modules, each serving a pivotal role in facilitating seamless interactions between users and the system. Our academic discourse on the main functionalities of the Car Rental web application is as follows:

1. **User Management Module**:

This module encompasses functionalities related to user account creation and administration. It facilitates the registration of staff members within the system, granting them unique login credentials for secure access. Administrative privileges empower designated personnel to oversee user accounts, ensuring proper access control and data security.

2. **Vehicle Management Module:**

The Vehicle Management module facilitates efficient oversight of the rental fleet. Staff members can seamlessly add, update, or remove vehicle listings, thereby maintaining accurate and up-to-date information regarding vehicle availability, pricing, and specifications. This module ensures that customers are presented with a comprehensive selection of vehicles to meet their rental needs.

3. **Reservation Handling Module:**

The Reservation Handling module streamlines the reservation process, enabling customers to make and modify bookings with ease. Staff members are equipped with tools to review and manage reservations, facilitating efficient allocation of vehicles based on availability and customer preferences. This module plays a pivotal role in ensuring a seamless booking experience for customers.

4. **Billing and Invoicing Module:**

Automated billing processes are central to the Billing and Invoicing module, generating invoices for reservations and additional services. Staff members can track payments, manage late fees, and generate financial reports for analysis and reconciliation. This module enhances financial transparency and efficiency, ensuring smooth financial operations within the rental business.

5. **Inventory Management Module:**

The Inventory Management module monitors vehicle availability in real-time, generating alerts for low stock levels to facilitate timely replenishment of the fleet. By optimizing inventory control, this module ensures that the right vehicles are available at the right time and location to meet customer demand, thereby maximizing revenue potential.

6. **Reporting and Analytics Module:**

- The Reporting and Analytics module provides comprehensive reports and analytics on various aspects of the rental business, including reservation trends, revenue generation, and customer demographics. These insights empower data-driven decision-making, enabling the company to refine strategies and drive business growth effectively.

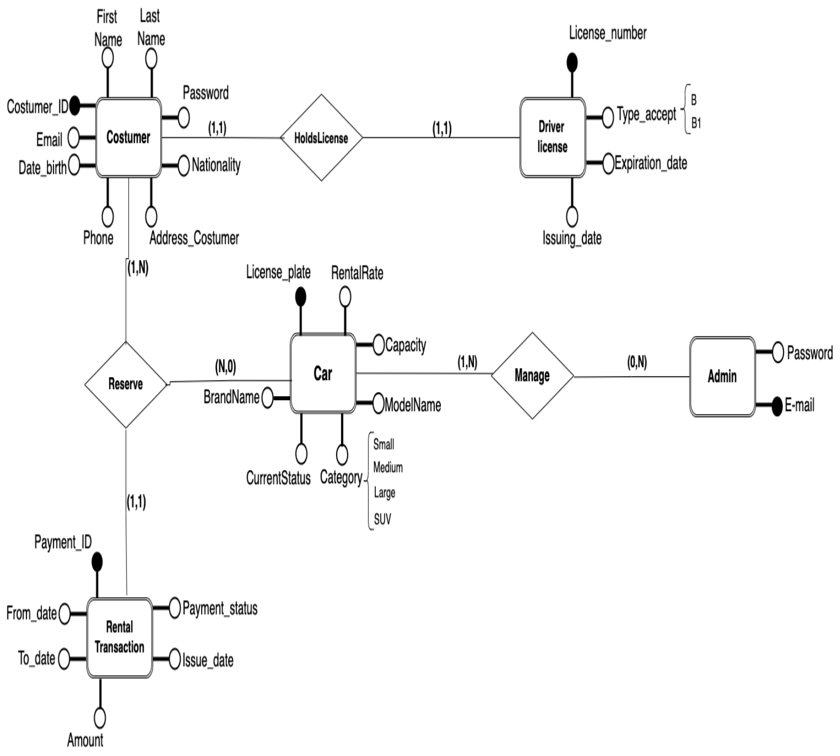
7. **Security and Access Control Module:**

Robust security measures, including role-based access control mechanisms, are integrated into the Security and Access Control module to safeguard sensitive functionalities and data. This module ensures that only authorized personnel have access to specific features, enhancing overall system security and protecting against unauthorized access or misuse.

By incorporating these functionalities, the Car Rental web application aims to optimize rental operations, enhance customer satisfaction, and foster long-term success in the competitive car rental industry landscape.

3 Data Logic Layer

3.1 Entity-Relationship Schema



3.2 Other Information

The Entity-Relationship (ER) schema for the Car Rental web application outlines key entities, their attributes, and relationships:

Entities: Customer, License, Car, Rental Transaction, Admin.
Relationships: HoldsLicense, Reserve, Manage.
Attributes: Unique identifiers (IDs), customer details, rental information, administrative data. Normalization: Ensures data integrity and efficiency.

This schema serves as the foundation for organizing and managing data within the application, supporting seamless rental operations and administrative functions.

Entity	**Attributes**	**Description**
Customer	Customer_ID	Identifier for the customer
	First Name	First name of the customer
	Last Name	Last name of the customer
	Email	Email address of the customer
	Date of Birth	Birth date of the customer
	Phone	Phone number of the customer
	Password	Password for customer login
	Nationality	Nationality of the customer
	Address	Address of the customer
License	License_number	Identifier for the license
	Issuing_date	Date of issuing the license
Car	Car_ID	Identifier for the car
	BrandName	Brand name of the car
	ModelName	Model name of the car
	Capacity	Capacity of the car
	CurrentStatus	Current status of the car
	Category	Category of the car
Rental Transaction	Payment_ID	Identifier for the payment
	From_date	Start date of rental
	To_date	End date of rental
	Amount	Rental amount
	Payment_status	Status of payment
Admin	Admin_ID	Identifier for the admin
	Type_accept	Acceptance type
	B1	B1 code
	Expiration_date	Expiration date of license
	E-mail	Email address of the admin

Table 2: Entity-Attribute-Description Table for ER Schema

4 Presentation Logic Layer

In our Car Rental web application's Presentation Logic Layer, we've crafted intuitive pages to streamline user interaction:

- The Registration Page allows swift account creation.
- The Add Cars Page simplifies fleet management.
- The User Dashboard offers personalized account insights.
- My Rental Cars Page facilitates document uploads.
- The Car Information Page provides detailed rental insights.
- The Home Page serves as a user-friendly entry point.

This setup ensures a seamless user experience and efficient navigation throughout the platform.

4.1 Home Page

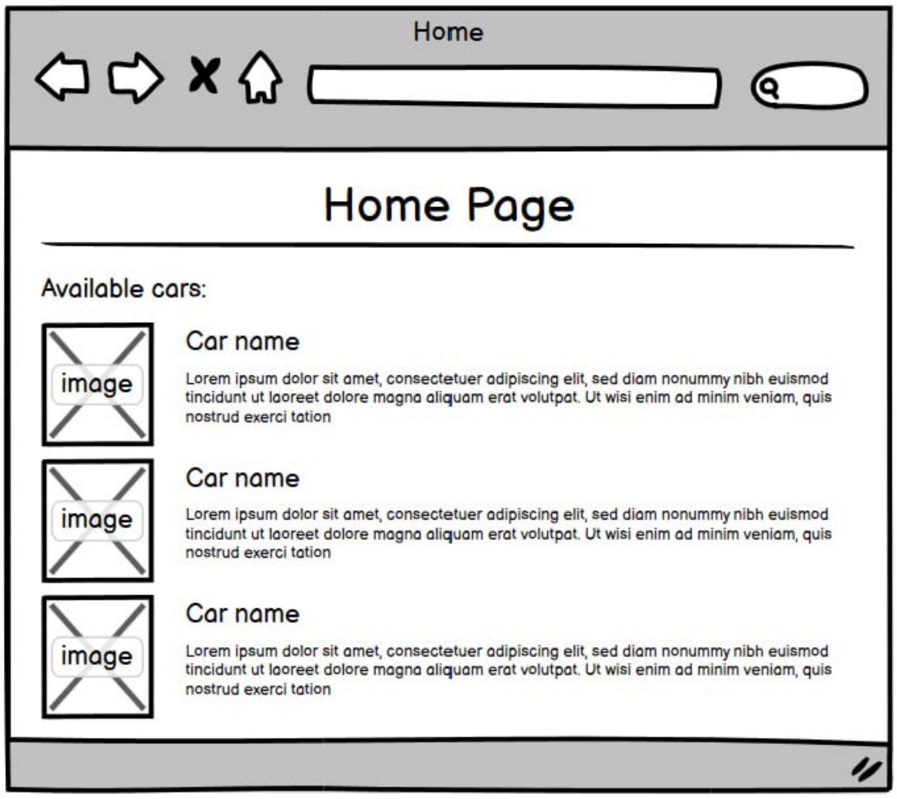

Home
Home Page
Available cars:
image
Car name
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
image
Car name
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
image
Car name
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation

Figure 1: Home Page

The Home Page serves as the main entry point, welcoming users with an overview of available cars, promotions, and featured vehicles. It provides a search and browsing interface, user sign-in options, customer testimonials, and informational resources to guide users seamlessly through the platform.

4.2 Registration Page:

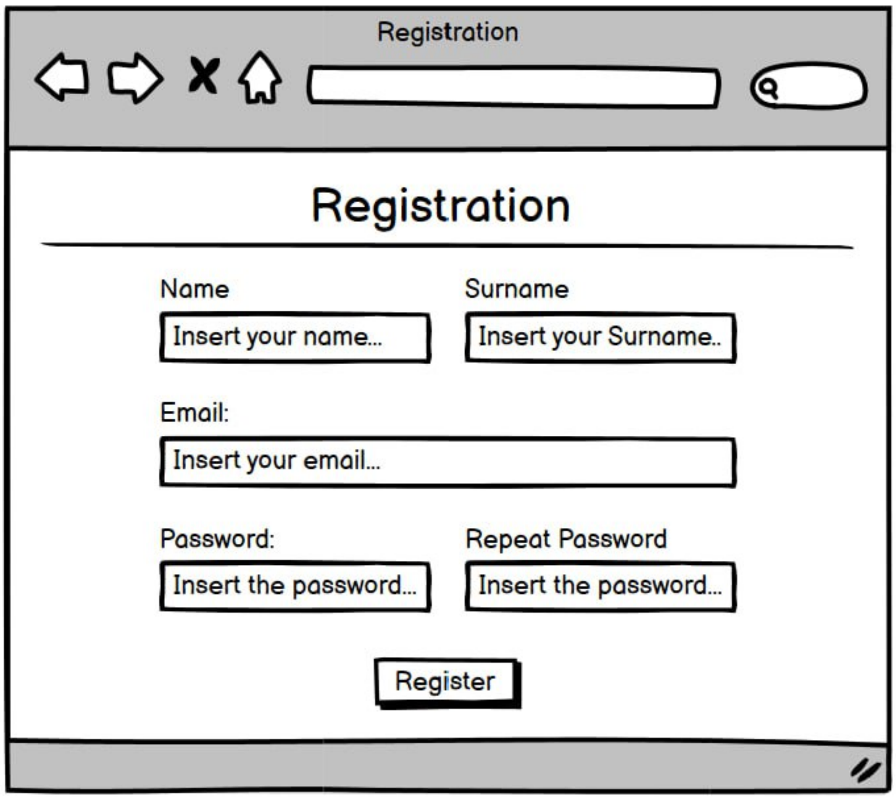


Figure 2: Registration Page

The Registration Page allows new users to create accounts by providing essential information such as their name, email address, password, and contact details. The registration form is user-friendly and intuitive, guiding users through the process step by step. Once the form is submitted, the user's account is created, and they gain access to personalized features and functionalities within the application. The Registration Page may also include optional fields for users to provide additional information, such as their preferences or interests, to tailor their experience further. Robust validation ensures that all entered data is accurate and secure, protecting user privacy and preventing fraudulent.

4.3 Add Cars Page:

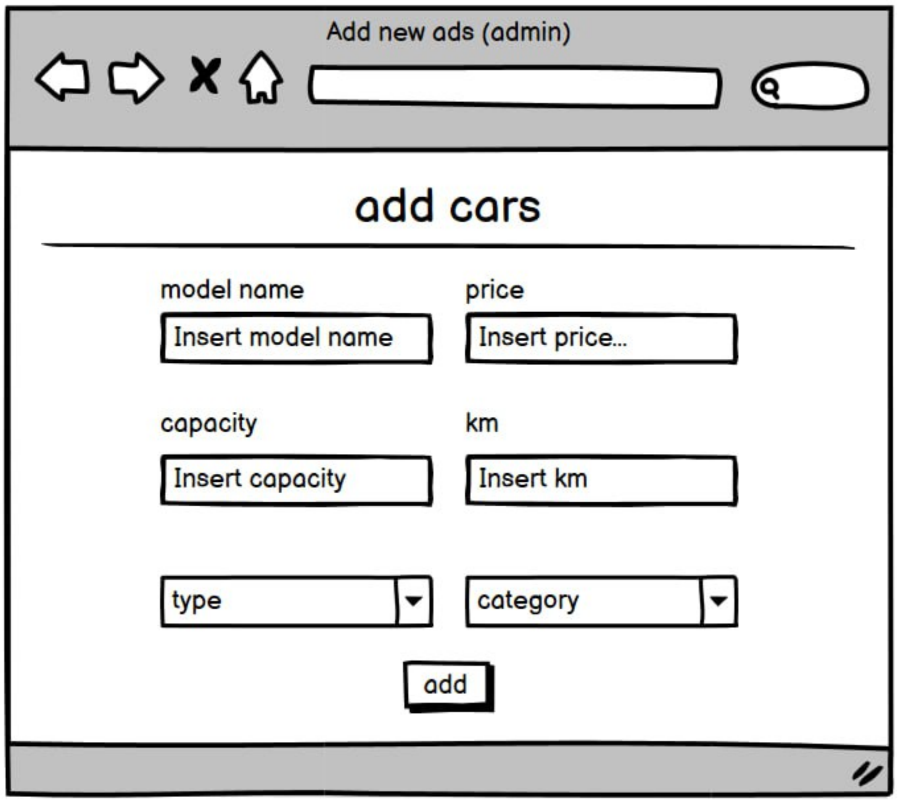


Figure 3: Add Cars Page

The Add Cars Page is accessible to administrators or authorized users responsible for managing the rental fleet. It features a comprehensive form where users can input detailed information about new cars to be added to the inventory. The form includes fields for the car's license plate number, rental rate, capacity, model name, category, current status, brand name, and any additional specifications or features. Users can upload photos of the car to provide visual representations for potential renters. Once the form is submitted, the new car is added to the rental inventory, and its details are updated in the database. The Add Cars Page streamlines the process of adding new vehicles, ensuring that the rental fleet remains up-to-date and well-maintained.

4.4 User Dashboard:

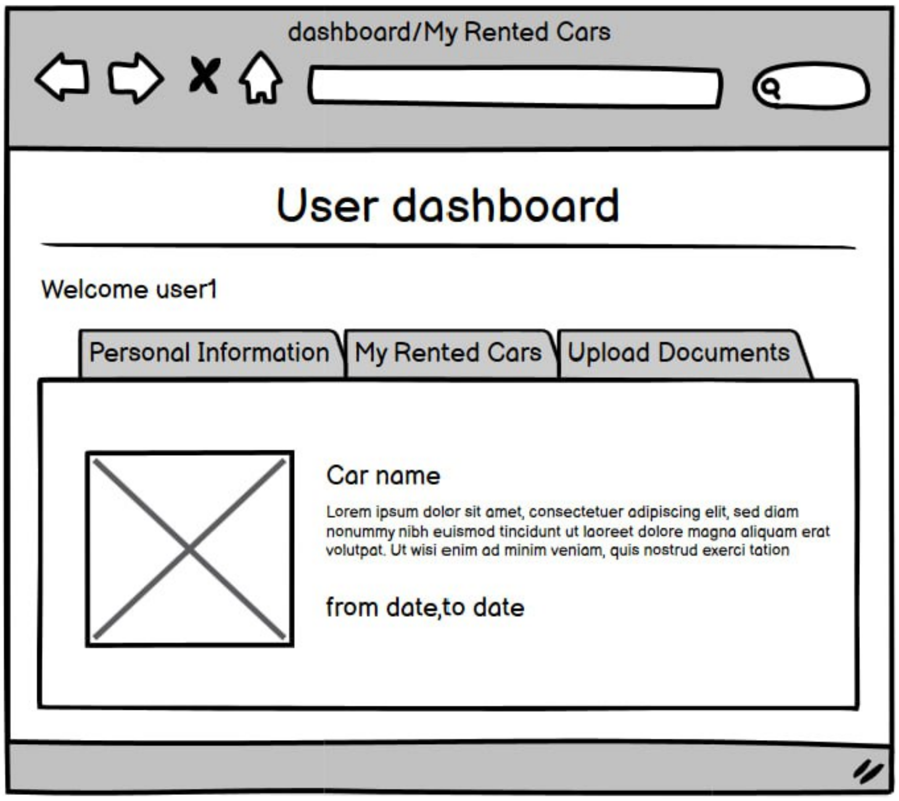

dashboard/My Rented Cars
User dashboard
Welcome user1
Personal Information
My Rented Cars
Upload Documents
Car name
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
from date,to date

Figure 4: User Dashboard Page

The User Dashboard serves as a centralized hub where registered users can access and manage their account information. It provides a personalized view of the user's profile, including their name, contact details, rental history, and preferences. Users can easily update their profile information, such as changing their password or updating their contact information, directly from the dashboard. The dashboard also displays any ongoing rental transactions, including the dates, duration, and status of each rental. Users can track their rental history, view past transactions, and access digital copies of rental agreements or receipts. The User Dashboard enhances user engagement and satisfaction by providing a convenient and intuitive interface for managing account activities and rental-related tasks.

4.5 Car Information Page:

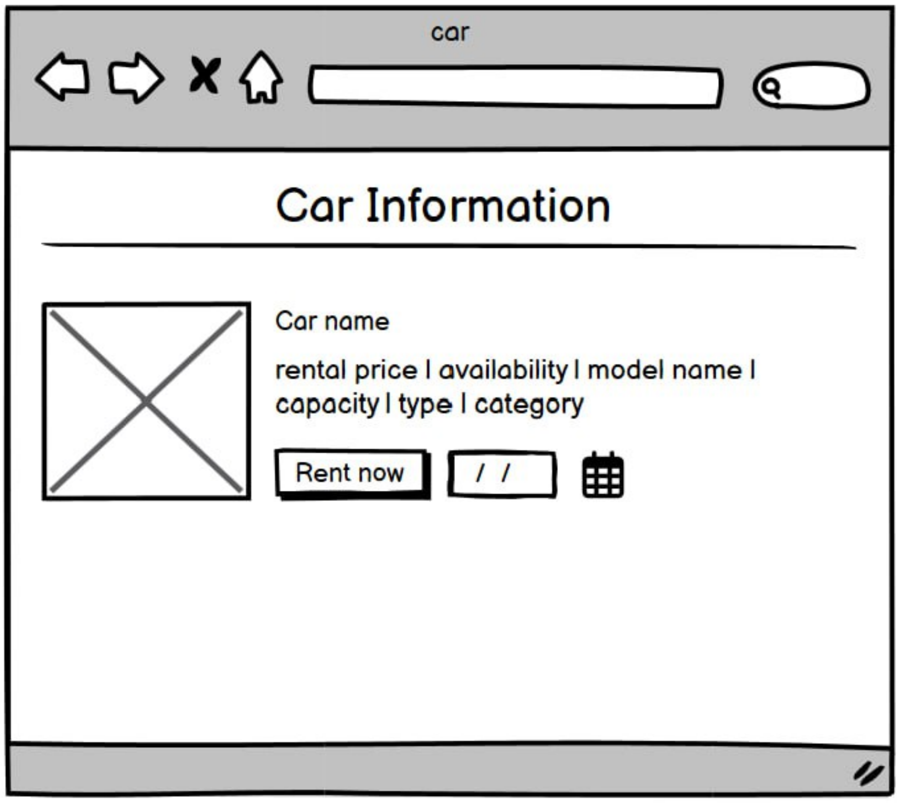
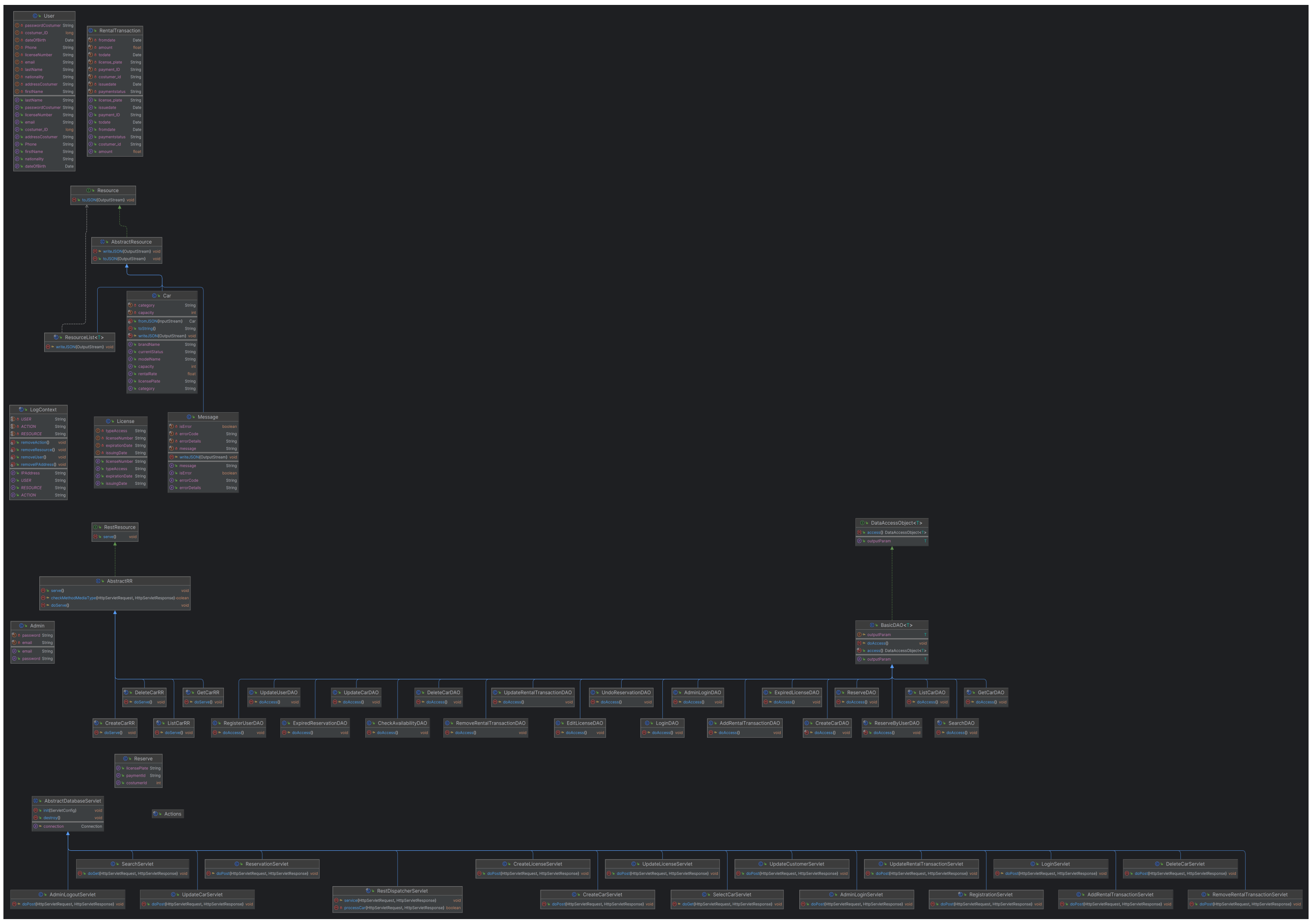

car
Car Information
Car name
rental price I availability I model name I
capacity I type I category
Rent now
/ /

Figure 5: Car Information Page

The Car Information Page provides detailed insights into individual cars available for rental. It features comprehensive information about each car, including specifications, features, rental history, and associated fees. Users can view high-quality photos of the car from different angles to get a better understanding of its appearance and condition. The page includes detailed descriptions of the car's make, model, year, mileage, and any special features or amenities it may have. Users can also see the availability of the car for their desired rental dates and check the rental rates and fees associated with renting the car. Clear call-to-action buttons prompt users to initiate the rental process, such as requesting a reservation or contacting customer support for assistance. The Car Information Page helps users make informed decisions about their rental choices by providing comprehensive and detailed information about each car in the rental inventory.

5 Business Logic Layer

5.1 Class Diagram



Component	Description
DAO	Data Access Objects (e.g., ‘AddRentalTransactionDAO‘, ‘AdminLoginDAO‘) facilitate interactions with data storage, enabling seamless data retrieval and manipulation for rental transactions, user authentication, and vehicle listings.
REST	RESTful resources (e.g., ‘AbstractRR‘, ‘CreateCarRR‘) serve as API endpoints, defining and handling incoming requests to provide functionalities such as creating cars and listing available vehicles.
Service Layer	Classes (e.g., ‘Admin‘, ‘Car‘, ‘RentalTransaction‘) encapsulate business logic for administrative tasks, vehicle management, and rental transactions, facilitating efficient operation of the application.
Servlets	Servlets (e.g., ‘AdminLoginServlet‘, ‘CreateCarServlet‘) handle HTTP requests and responses, serving as entry points for processing user interactions and orchestrating communication between the client and server.

Table 3: Class Diagram Components

5.2 Sequence Diagram

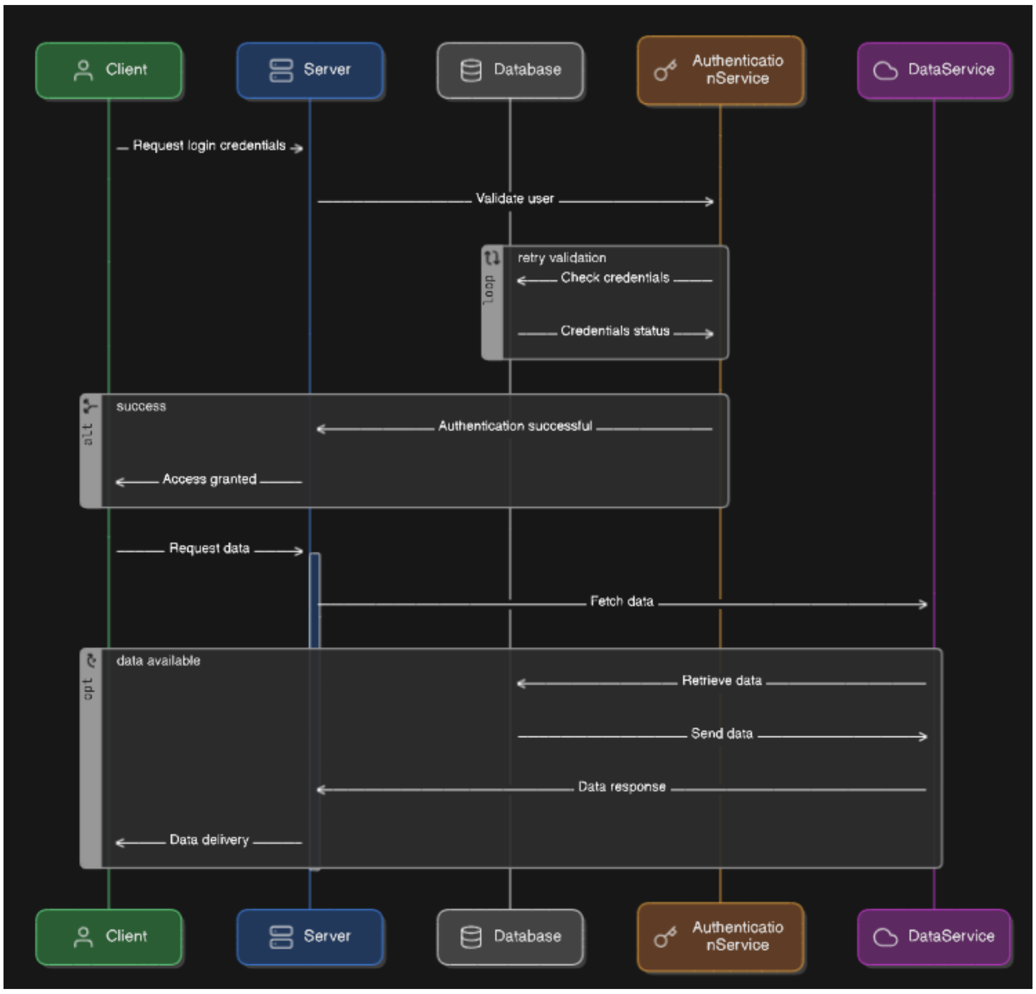


Figure 6: Login Sequence diagram

Description:

The login process begins when the client sends a request to the server with the user's credentials. The server verifies the provided username and password against stored data. If the credentials are valid, the server responds with a success message, granting the user access. This sequence diagram outlines the essential steps involved in the login process and emphasizes the importance of authentication for system security.

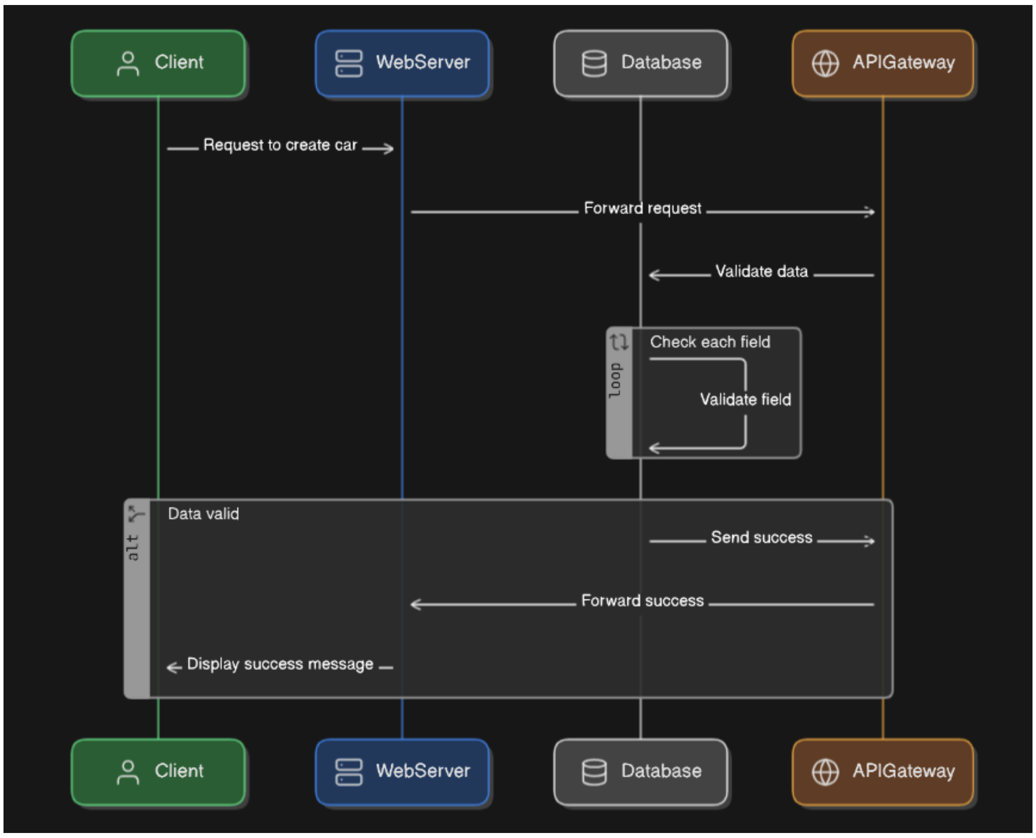


Figure 7: Car Sequence Diagram

Description:

The car sequence diagram illustrates the process flow involved in managing cars within the system. It visually depicts the interactions between various components, including clients, servlets, DAOs (Data Access Objects), resources, and databases.

In this sequence, the client initiates the process by sending a request to the server, often facilitated by a servlet. The servlet then communicates with the appropriate DAO to perform operations related to cars, such as creation, updating, or reservation. The DAO interacts with the database to retrieve or modify car-related data accordingly. For instance, in a car reservation scenario, the sequence might show the client requesting to reserve a car. The servlet processes this request and communicates with the reservation DAO to verify car availability and update the reservation status in the database. Finally, the servlet responds to the client, indicating the success or failure of the reservation process.

Overall, the car sequence diagram provides a visual representation of the steps involved in managing cars within the system, aiding in understanding the process flow and interactions between different system components.

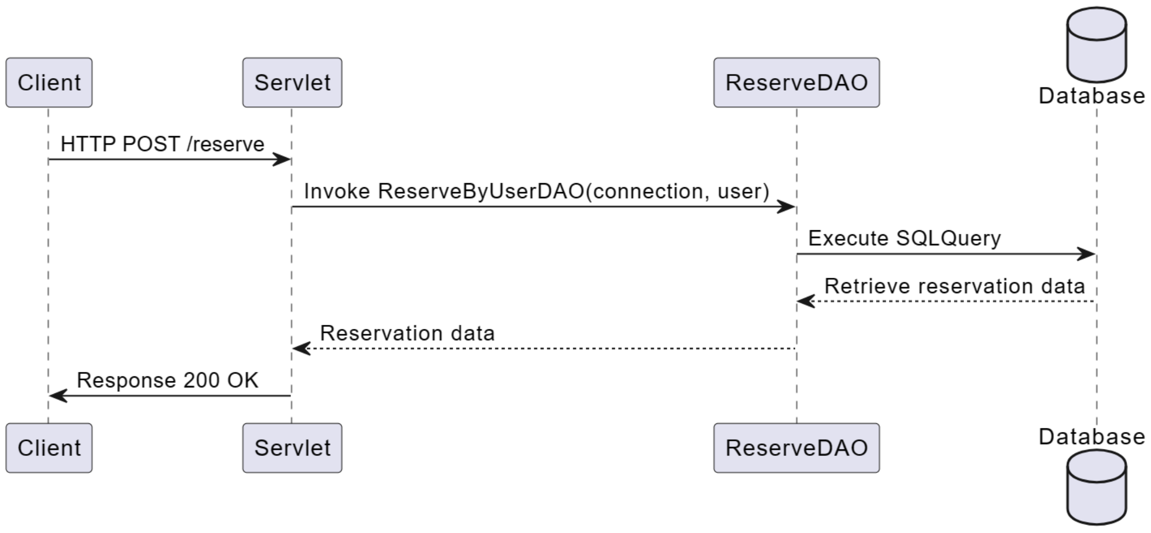


Figure 8: Reserve Sequence Diagram

Description:
The car reservation sequence diagram illustrates the process of reserving a car within the system.
The client initiates the process by sending an HTTP POST request to the servlet, triggering a series of interactions. The servlet interacts with the ReserveDAO to access the database and retrieve reservation data for the specified user. Following this, the database processes the request, retrieves the necessary data, and returns it to the servlet. Finally, the servlet responds to the client with an HTTP status code of 200 OK, indicating a successful reservation. This sequence diagram provides a clear overview of the interactions involved in the car reservation process, from client request to server response.

5.3 REST API Summary

We used postman REST client to test and debug our REST API. here we share list of routes and a documentation with other developers:

URI	Method	Description	Filter
/rest/car	GET	get the list of cars	no
/rest/car	POST	make a new car record	no
/rest/car/{licenseplate}	GET	get a car by its license plate	filters cars by license plate
/create-car	POST	make a new car record	no
/update-car	POST	update a car record	no
/search-car	GET	get the list of cars	can filter with cars parameters
/create-customer	POST	Registering a new customer	no
/edit-customer	POST	Registering a new customer	no
/login	POST	Login a customer	no
/create-licence	POST	Creating a new Licence	no
/edit-licence	POST	update a car record	no
/add-transaction	POST	create a payment transaction	no
/remove-transaction	POST	remove a payment transaction	no
/update-transaction	POST	remove a payment transaction	no

Table 4: Describe in this table your REST API

5.4 REST Error Codes

Error Code	HTTP Status Code	Description
E300 - E4A1 - E4A3 - E4A7	400	Bad Request
E404 - E5A3	404	Not Found
E4A5	405	Method Not Allowed
E4A2	406	Not Acceptable
E5A2 - E5A4	409	Conflict
E4A4	415	Unsupported Media Type
E200 - E5A1	500	Internal Server Error

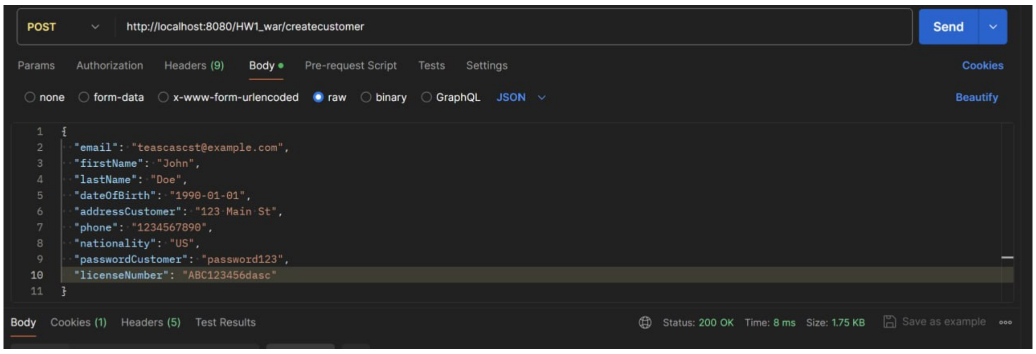
Table 5: Describe in this table your REST API

5.5 REST API Details

Register Form

REST API details for the Register form include endpoints and request/response formats for handling user registration. The form collects essential user details such as email, first name, last name, birth date, address, phone number, nationality, password, and license number.

- URL: `/createCostumer`
- Method: `Post`
- Data Parameters:



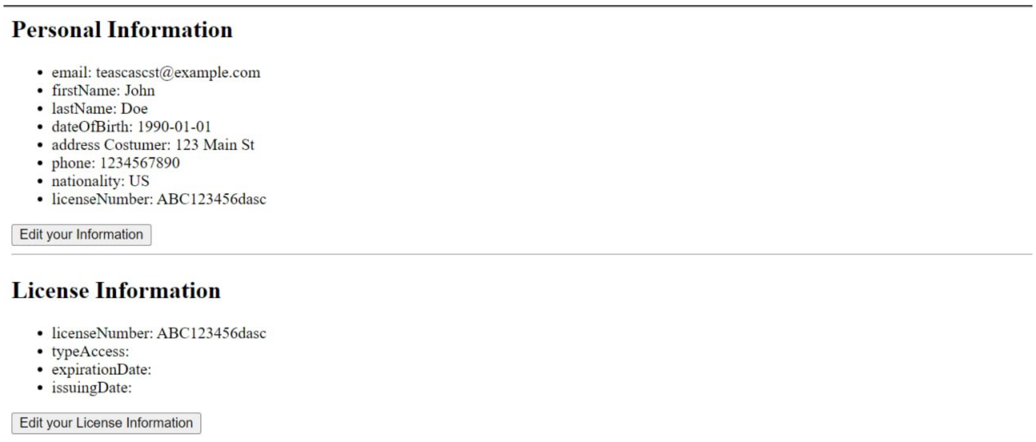
POST http://localhost:8080/HW1_war/createcustomer

```
{
  "email": "teascascst@example.com",
  "firstName": "John",
  "lastName": "Doe",
  "dateOfBirth": "1990-01-01",
  "addressCustomer": "123 Main St",
  "phone": "1234567890",
  "nationality": "US",
  "passwordCustomer": "password123",
  "licenseNumber": "ABC123456dasc"
}
```

Status: 200 OK Time: 8 ms Size: 1.75 KB

Figure 9: Register Form Data Parameter

- Success Response:



Personal Information

- email: teascascst@example.com
- firstName: John
- lastName: Doe
- dateOfBirth: 1990-01-01
- address Costumer: 123 Main St
- phone: 1234567890
- nationality: US
- licenseNumber: ABC123456dasc

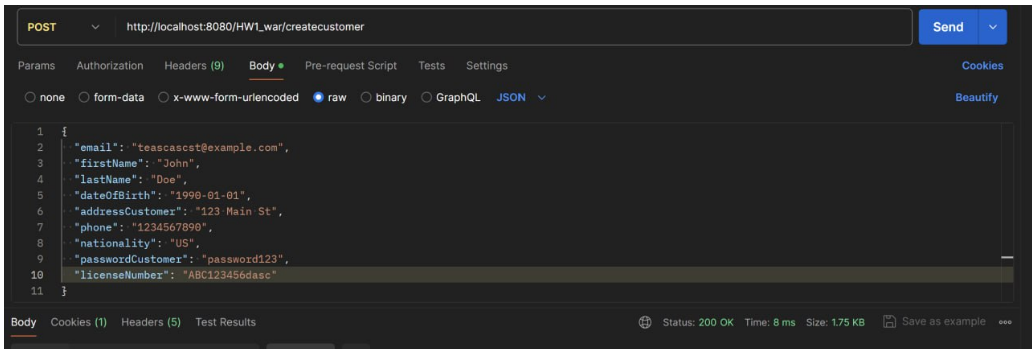
License Information

- licenseNumber: ABC123456dasc
- typeAccess:
- expirationDate:
- issuingDate:

Figure 10: Create Costumer Success Response

Update Car

- URL: `/updateCar`
- Method: `Post`
- Data Parameters:



POST http://localhost:8080/HW1_war/createcustomer

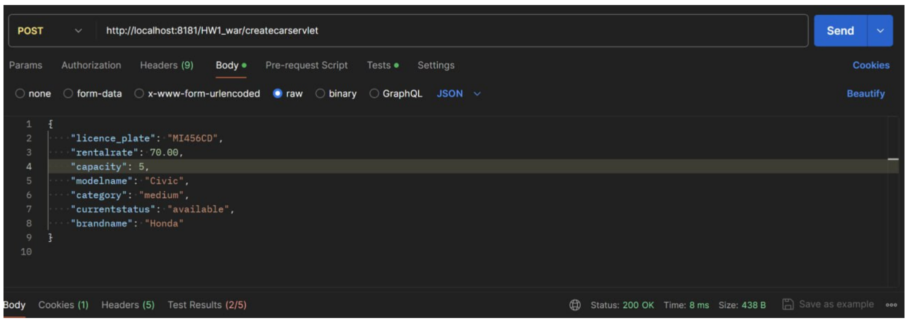
Body: raw, JSON

```
{
  "email": "teascascst@example.com",
  "firstName": "John",
  "lastName": "Doe",
  "dateOfBirth": "1990-01-01",
  "addressCustomer": "123 Main St",
  "phone": "1234567890",
  "nationality": "US",
  "passwordCustomer": "password123",
  "licenseNumber": "ABC123456dasc"
}
```

Status: 200 OK Time: 8 ms Size: 1.75 KB

Figure 11: update Car Data Parameter

- Success Response:



POST http://localhost:8181/HW1_war/createcarservlet

Body: raw, JSON

```
{
    "licence_plate": "MI456CD",
    "rentalrate": 70.00,
    "capacity": 5,
    "modelname": "Civic",
    "category": "medium",
    "currentstatus": "available",
    "brandname": "Honda"
}
```

Status: 200 OK Time: 8 ms Size: 438 B

Figure 12: Update Car Success Response

CAR REST APIs

Here we show some spesific REST API endpoints for managing the car entity. this APIs are under "/rest/car" URL:

- URL: `the URL to retrieve it :  /rest/car`
- Method: `Method to retrieve it:  GET`
- URL Parameters: none
- Data Parameters: none
- Success Response:

```
        {
          "resource-list": [
            {
              "car": {
                "licenseplate": "HR23791B",
                "rentalrate": 50.0,
                "capacity": 5,
                "category": "medium",
                "currentstatus": "available",
                "brandname": "Toyota"
              }
            },
            {
              "car": {
                "licenseplate": "T6237FX5",
                "rentalrate": 70.0,
                "capacity": 4,
                "category": "small",
                "currentstatus": "available",
                "brandname": "Honda"
              }
            }
          ]
        }
```

Listing 1: Response Example

- Error Response:

```
        500 :  Internal Server Error
```

Listing 2: Possible Errors

- URL: `the URL to retrieve it :  /rest/car`
- Method: `Method to retrieve it:  POST`
- URL Parameters: none
- Data Parameters:

```
{
    "car":{
    "licenseplate":"H3361WE0",
    "rentalrate": 60.0,
    "capacity": 4,
    "category":"medium",
    "currentstatus":"available",
    "modelname": "X5",
    "brandname": "BMW"

    }
}
```

Listing 3: Response Example

- Success Response:

```
{
     "car":{
     "licenseplate":"H3361WE0",
     "rentalrate":60.0,
     "capacity":4,
     "category":"medium",
     "currentstatus":"available",
     "brandname":"BMW"
     }
 }
```

Listing 4: Response Example

- Error Response:

```
500 :  Internal Server Error (Cannot create the car: unexpected error. |
    Cannot create the car: unexpected database error.)
400 : Bad Request (Cannot create the car: no Car JSON object found in
   the request.)
409: Conflict (Cannot create the car: it already exists.)
```

Listing 5: Possible Errors

- URL: `the URL to retrieve it :  /rest/car/{licenceplate}`
- Method: `Method to retrieve it:  GET`
- URL Parameters: none
- Data Parameters:none
- Success Response:

```
      {
          "car":{
          "licenseplate":"H3361WE0",
          "rentalrate":60.0,
          "capacity":4,
          "category":"medium",
          "currentstatus":"available",
          "brandname":"BMW"
          }
       }
```

Listing 6: Response Example

- Error Response:

```
       500 :  Internal Server Error (Cannot get car: unexpected error. | Cannot
           get car(s): unexpected database error.)
```

Listing 7: Possible Errors

- URL: `the URL to retrieve it : /rest/car/{licenceplate}`
- Method: `Method to retrieve it: DELETE`
- URL Parameters: none
- Data Parameters:none
- Success Response:

```
       {
            "car":{
            "licenseplate":"H3361WE0",
            "rentalrate":60.0,
            "capacity":4,
            "category":"medium",
            "currentstatus":"available",
            "brandname":"BMW"
            }
        }
```

Listing 8: Response Example

- Error Response:

```
        500 :  Internal Server Error (Cannot delete the car: unexpected database
             error.)
        404 : Not Found ("Car " + lisenceplate + " not found. Cannot delete it."
             | Cannot delete the car: wrong format for URI /car/{licenceplate}.)
        409: Conflict (Cannot delete the car: other resources depend on it.)
```

Listing 9: Possible Errors

6 Group Members Contribution

Francesco Chemello played a pivotal role in our group project, contributing extensively across various domains. He led the development of essential components such as the Car DAO, Card DAO, Rental Transaction DAO, Search DAO, Expired License DAO, Expired Reservation DAO, and Undo Reservation DAO. In addition to these backend implementations, Francesco spearheaded the creation of service layers, encompassing functionalities related to Car, Card, Rental Transaction, and Reserve Services. His expertise also extended to servlet implementations, where he meticulously handled all aspects concerning Car, Card, and Rental Transaction functionalities. On the database front, Francesco contributed significantly by designing the ER schema and ensuring thorough documentation by adjusting code for JavaDoc generation. Furthermore, he undertook crucial code maintenance tasks, addressing issues in files like BasicDAO, Actions, Abstract Database Servlet, Web.xml, and Maven Configuration/POM. To guarantee seamless integration, Francesco conducted comprehensive testing using Maven to validate the code's packaging and functionality within our project framework.

Elnaz Dolati As a crucial member of our team, I played a pivotal role in shaping the foundation and functionality of our project. I took the lead in setting up the initial project structure, ensuring it was conducive to smooth development and collaboration. Additionally, I focused on building a secure authentication system to safeguard our project's integrity. In terms of documentation, I made significant contributions to the project report, particularly in detailing the CAR REST APIs, where I provided thorough insights and explanations.

Gabriella Ingridy De Souza Farias Contribution on ER schema, tables and insert SQL, some code of Reserve and the functionality SearchByUser.

Luca Pellegrini I developed all the REST API: CreateCar, DeleteCar, GetCar and ListCar creating and modifying the related DAO and servlets. I contributed to the integration of the database into the web application making necessary modifications to both the database structure and the web application code.

Seyedreza Safavi played a pivotal role in our project, ensuring robust database connectivity and functionality. He enabled database connection via a context.xml file and developed essential login and register servlets and DAOs, rigorously tested with sample data. Additionally, Reza implemented edit DAOs and servlets for user and license information updates, enhancing project flexibility. He optimized project performance by modifying key classes like User, License, LogContext, and Message. Notably, Reza streamlined data management by implementing a database function to write simultaneously to customer and license tables. He also created all necessary JSPs, facilitating data reception, database interaction, result display, and seamless navigation. Furthermore, Reza modified the web.xml file to effectively connect all servlets and JSPs. He conducted comprehensive testing of Francesco's DAOs and servlets, ensuring compatibility with a simplified database structure. Moreover, he handled exceptions effectively, enhancing the robustness of the application.

Ahmad Sadin was heavily involved in shaping the majority of the report's content. He took the lead in defining key functionalities, crafting the ER-schema, outlining objectives, establishing the data logic layer, and developing class and sequence diagrams. Additionally, Ahmad played a vital role in shaping the presentation logic layer, business logic layer, and specifying REST API details. Moreover, he closely collaborated with Seyed Reza on various tasks, including refining JSP files and implementing user servlets and DAOs. These combined efforts under Ahmad's guidance ensured the successful completion of the project and the achievement of its objectives.