Project Report

Best Rent - A Car Rental System

CS 411: Software Engineering

Instructor: Ahmed Ibrahim

Group Member :

Tianyi Bao (U89641075)

Xiang Li (U53450247)

Zijie Li (U91267087)

Zhangde Song (U06937573)

Problem definition:

Develop a system that a car rental company would be able to manage car rentals and customers would send orders to reserve their required car.

The specific functionality that the system should have includes the following:

* Search for cars that are available and the ones that are rented.
* Record how long a car will be rented.
* A rental fee is stored for every different car.
* Store and check customer information for further uses.
* Evaluate and update cars’ damage status.
* Add new cars and delete scrapped ones.

Project Objective

* Generate a clear car rental database so that adding and deleting rental information can be made in 3 steps.
* Develop an easy and clear website for customers to reserve cars for renting.
* Develop a clear system for car rental companies to manage their cars.

Stakeholders List

* Internal stakeholders: car rental companies

Interest: Selling and renting cars, a management system that helps organize car stock

Impact: car rental companies can manage their business more efficiently

Potential influence: Increasing Selling and earnings, Decreasing human resources cost

* External stakeholders: customers

Interest: renting cars effortlessly

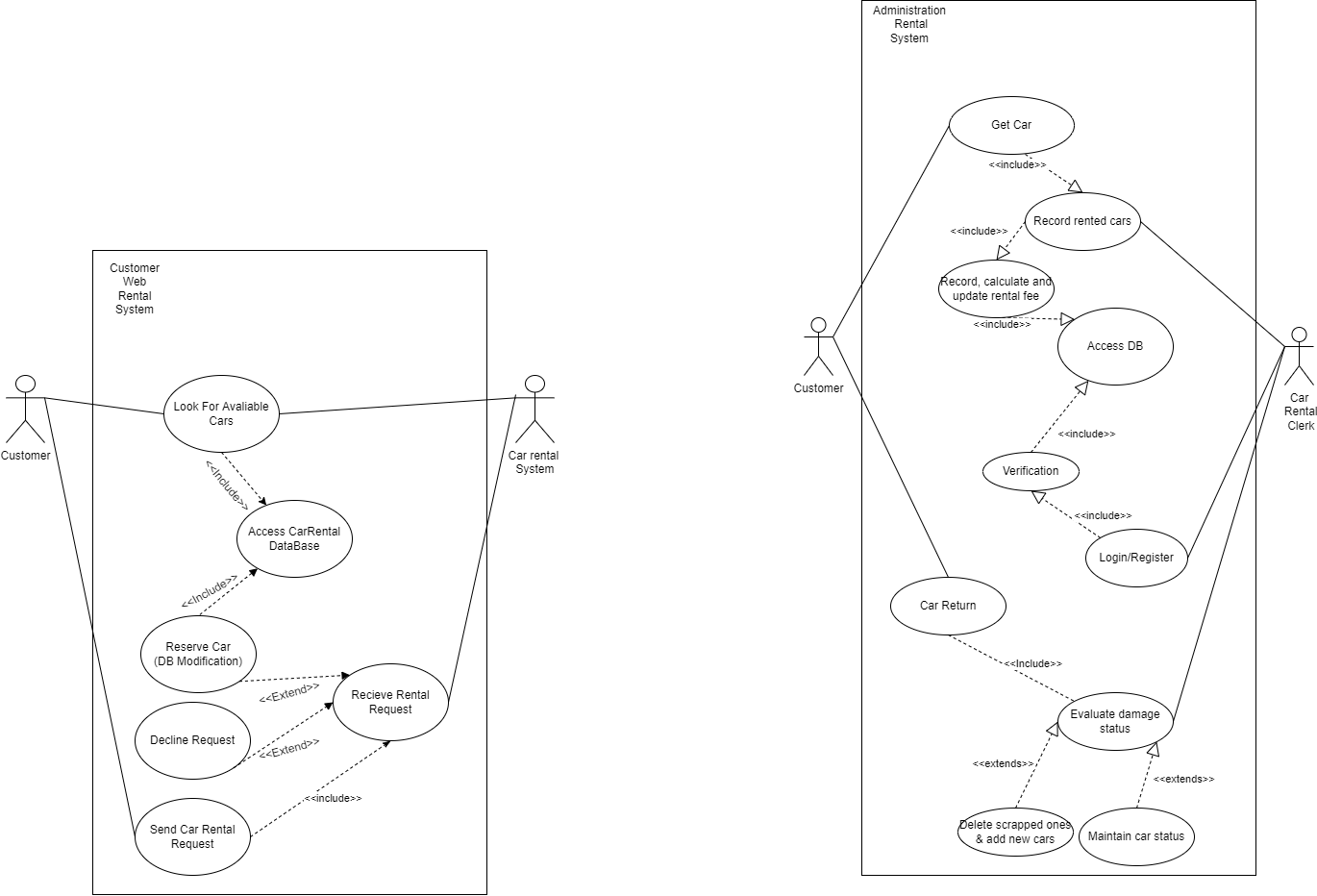
Impact: customers have a more user-friendly car rental system to interact with

Potential influence: Increase customer satisfaction

Success / Accept Criteria for each Stakeholder

* For internal stakeholders (car rental companies):
  + low learning cost,
  + Efficient to manage cars in stock and renting
* For external stakeholders (rental customers):
  + user-friendly interface to find the wanted car,
  + stable system to process requests,

Use Case Diagrams



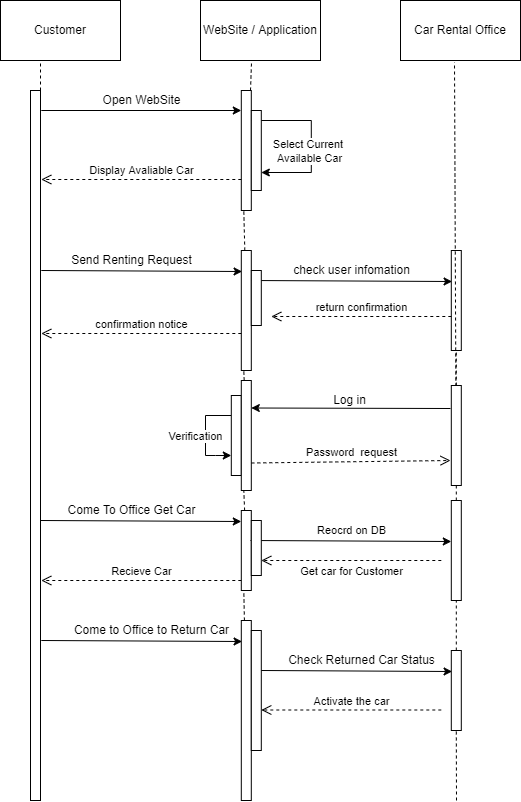
Graph #1: Use Case Diagrams(Web-Application & Computer Application)

Selected use case description

On the website end (left), the customer will be able to check which cars are available (rentable), and this includes the action that “Access the Car Rental Company DataBase”. The customer will be able to send their order to the database. The car rental system will receive this order. On the Car rental system side, it provides the information on the available car list. It needs to access the database to grab that information. The system will receive the customer information. If the order information is correct, the order request will be approved and the car will be reserved, including changes on the database. If not, the request will be declined.

On the application end (right), the customers will be able to “get cars” in the car rental office. First, the car rental clerk will need to log in to the system with administrator credentials. This will activate the system to verify the credentials in the database. Then, this will activate the clerk record of the rented car on the system, the system will calculate the actual fee and get an update on the database. When the user comes to return the car, the clerk will evaluate the returned car and make records on the database(it may include deleting scrapped cars, adding new cars, or maintaining car status.)

Detailed Sequence Diagram



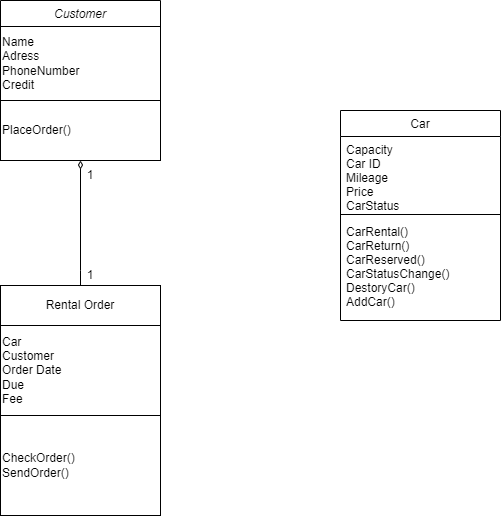
Graph #2: Detailed Sequence Diagram for the whole rental system

System Architecture

**Monolithic Architecture** is made to handle several jobs that are all related. A car renting system made for a specific company is targeting the functionality of finding the best car and renting the car to customers. The system job is very related. Such as, searching for the best-fit car(based on renting fee, car model, brand, capacity…), renting option, and verification page… A monolithic application is usually a complex application with multiple tightly related functions. In the car renting system, we have a web page for customers to find the right car, and we have a page for customers to choose their renting payment option. We also have an application for renting companies to manage car stock. To check and set the status of in-stock cars.

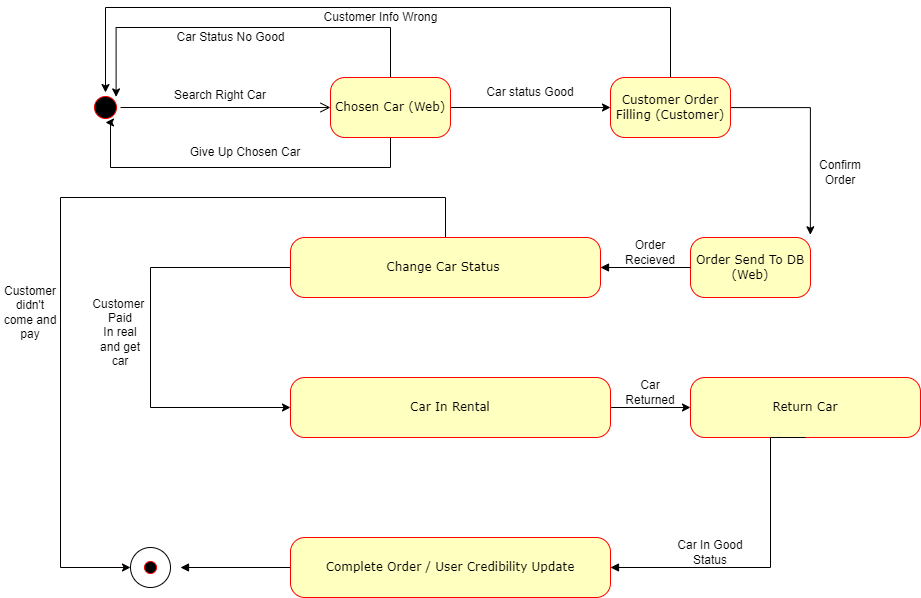
**The Layered System Architecture Pattern** is used in our design. At the very top of the layered architecture, it is the Web-Based Application which is for customers to send rental data. Under this layer, it is the administration application that is for rental officers to handle the management of the car data. At the very bottom, it is the SQL database to support the data that could operate on each level.

Detailed Class Diagram



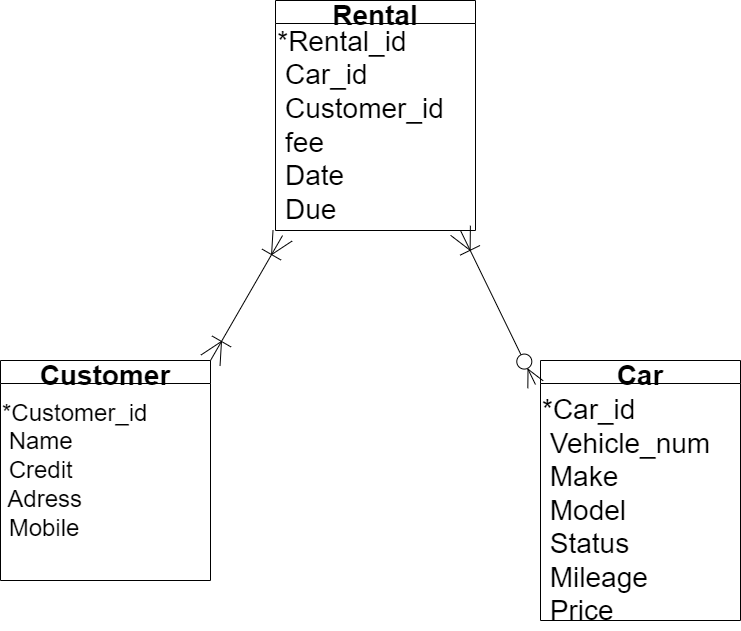
Graph #3: Detail Class Diagram - Three classes: Car, Rental Order, Customer

State Machine Diagram



Graph #4: State Machine Diagram - demonstrates the various state the software might be

ER-Diagram



Graph #5: ER-diagram - The foundation of MySQL database design

Git-Hub Resipotory Link

[https://github.com/lzivan/CS411\_Project](https://github.com/lzivan/CS411_Project.git)

Conclusion

Throughout this assignment, our team members learned a lot.

From a skill point of view, we learned java application development and how java interacts with databases using JDBC and Apache. Especially in Java terminal design, we learned how to use JFrame as a base model, then we could add buttons, labels, and tables to the Java terminal.

We have also learned how to write HTML and JSP like how java can be embedded into the webpage.

From a project management perspective, we learned the following lessons.

First, during the project planning stage, group members need to plan as much as possible before starting a project. As people work in groups, it is important to be on the same page. The planning part ensures that.

Secondly, the planned project schedule needs to leave enough time for contingencies. People may have different situations, as group members, it is our responsibility to help and accommodate. Besides that, if we could have the extra time it would be good.

Finally, group members need to have good communication with each other. As group members work together, people easily get different ideas. Arguments may occur. Also, it is possible that group members don’t have enough communication which leads to misunderstanding. Ideas and thoughts need to be conveyed throughout the project the first time, preventing further problems.

Appendix

Reference

Leaning Material of Java Application Development:

Java Car Rental System, from Tutus Funny

<https://www.youtube.com/playlist?list=PLuji25yj7oIII_Ep880KdESyu2n2VTZdH>

Java API Implemented in This Project:

java.sql.Connection

java.sql.DriverManager

java.sql.PreparedStatement

java.sql.ResultSet

java.sql.ResultSetMetaData

java.sql.SQLException

java.sql.Statement

java.util.Vector

java.util.logging.Level

java.util.logging.Logger

javax.swing.JOptionPane

javax.swing.table.DefaultTableModel

jakarta.servlet.ServletException;

jakarta.servlet.annotation.WebServlet;

jakarta.servlet.http.HttpServlet;

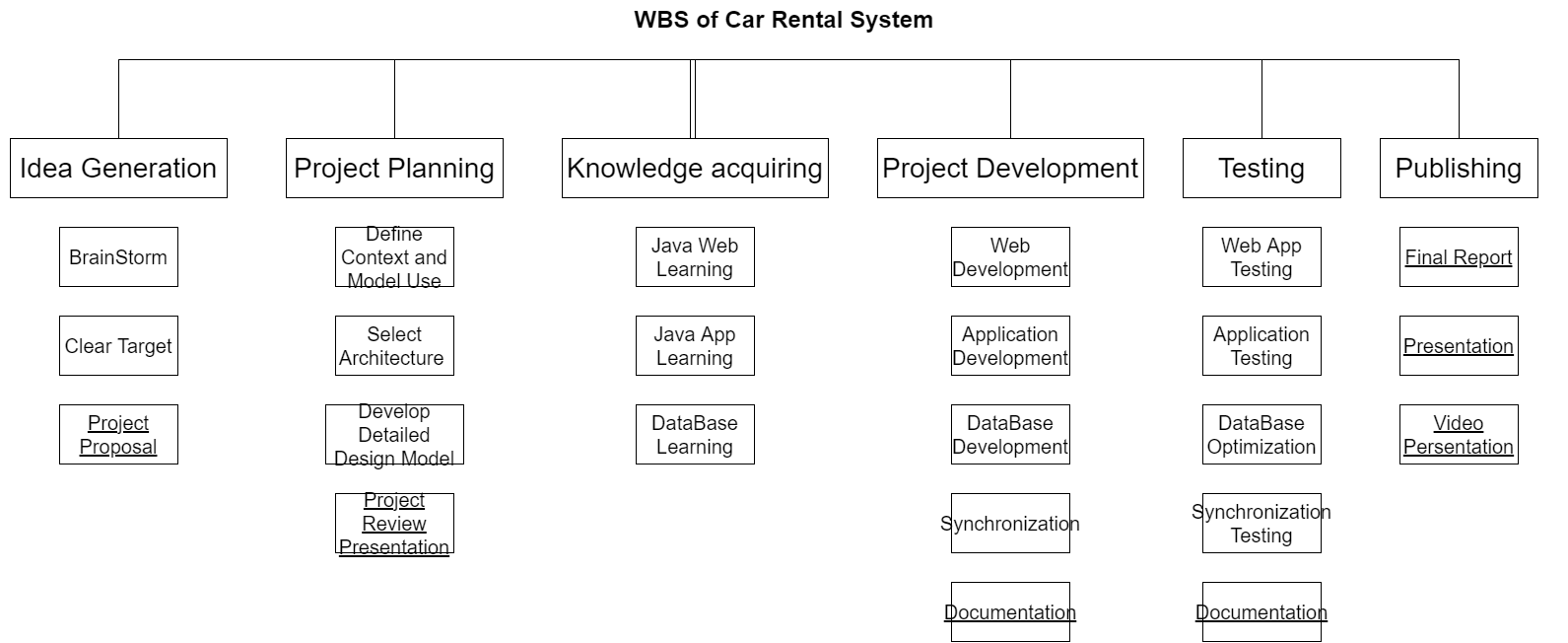
jakarta.servlet.http.HttpServletRequest;

jakarta.servlet.http.HttpServletResponse;

java.time.LocalDate;

java.time.temporal.ChronoUnit;

Project WBS



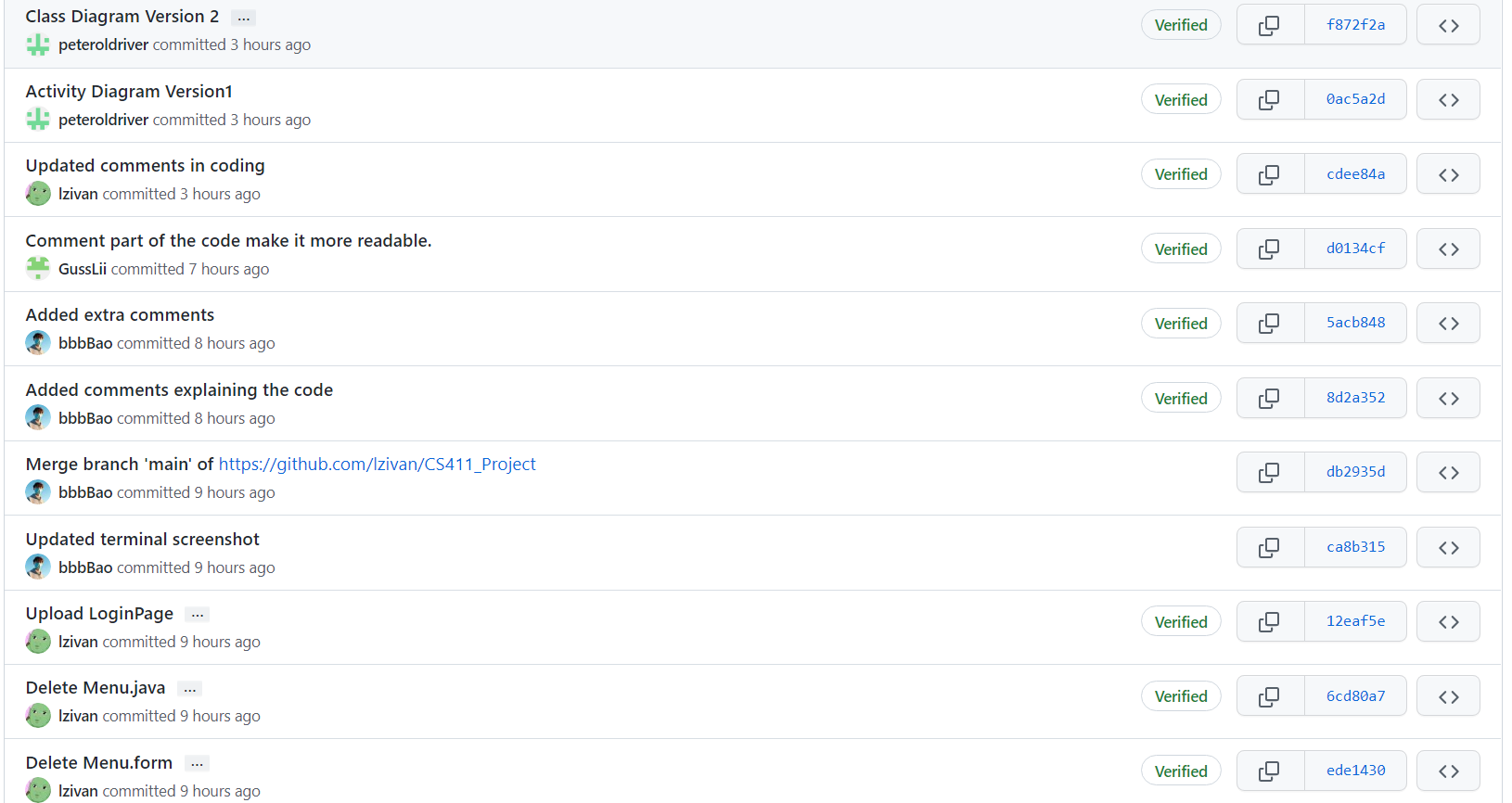
Graph #6: the Work Breakdown Structure - Various Work Necessary for Finishing This Project

Task Assignment Matrix



Graph #7: Task Assignment Matrix - The Responsibility Distribution Graph

Sample of Commits On the Selected Version Control System



Graph #8: Sample Commits On Selected Version Control System