EUROPEAN UNIVERSITY OF LEFKE
DEPARTMENT OF SOFTWARE ENGINEERING

**Semester Project**
**COMP 337**
**DATABASE MANAGEMENT SYSTEMS**

# CAR RENTAL SYSTEM

Name: Josiah Ayanfeoluwa Opaleke
Student number: 22146477

Name: Owolabi Joseph AyoOluwa
Student number: 22146431

**Submitted to Mr. Salman Khan**

LEFKE 2024

# Grading Sheet

| Student's Name | | | |
|---|---|---|---|
| Student's Number | | | |
| **Grading Criteria** | **Marks** | **Team Member 1** | **Team Member 2** |
| 1. Functionality | 10 | | |
| 2. ER Diagrams | 10 | | |
| 3. Database Working | 05 | | |
| 4. GUI Interface | 03 | | |
| 5. Report and Format | 02 | | |
| 6. Viva | 10 | | |
| Total | 40 | | |

**Comment's (if Required):**

**Teacher Signature: _____**

**Date: _____**

# Table of Contents

# Chapter 1: Project Overview

## Problem of Interest

### Problem Statement:

Managing a car rental across multiple locations and branches can be a very complex task. It involves tracking and managing car availability, ensuring efficient rental processes and maintaining records of all customers and records. The manual way of keeping track is somehow an outdated systems often leading to inefficiencies data loss, and poor customer experience.

### Objectives:

The main objective of our application is to provide a user-friendly GUI-based on desktop for car rental businesses to efficiently manage their operations and services.

### Goals:

a. Provide interface for tracking available cars and rent status
b. Simplify the management of cars in an inventory across different branches.
c. Enhance data insights for better decision-making.
d. Automate the process of customer rental processes, including contract generation and return status.

## Team Details

### Team Members:

- Josiah Ayanfeoluwa Opaleke - Lead Database Design
- Joseph AyoOluwa Owolabi - Lead Frontend Developer

## Job Descriptions

Josiah - Lead the development of the database design and schema, implementing the backend development for data storage, update and retrieval of data from the database.
Ayo - Led the development of the frontend, ensuring a user-friendly interface and implementing data handling logic to connect with the backend and also assisted in developing backend development.

| DAY NO | JOSEPH AYOOLUWA OWOLABI | JOSIAH AYANFEOLUWA OPALEKE | STATUS |
|---|---|---|---|
| **DAY 1** | Making relational schema | Making relational schema | Completed |
| **DAY 2** | Listing requirements | Designing basic user interface | Completed |
| **DAY 3** | Implementing the user interface | Creating SQL tables and queries | Completed |
| **DAY 4** | Implementing the user interface | Creating SQL tables and queries | Completed |
| **DAY 5** | Combining UI with database | Combining UI with database | Completed |
| **DAY 6** | Testing and debugging | Testing and debugging | Completed |
| **DAY 7** | Filling report | Filling report | Completed |

# Chapter 2: Programming Language Choice

Rationale for choosing Python and PyQt6

a. **Ease of Development:**

Python is known for its simplicity, which allowed us to write clean and readable codes to maintain code efficiency. We combined the frontend with PyQt6 framework to make it easier to develop the frontend GUI of our desktop application to make it modern and user friendly to use.

b. **Community Support:**

Python has a large well developed active community, making it easy to find solutions to our problems, tutorials, and libraries for almost any problem we encountered during development. Similarly, PyQt6 benefits from detailed documentation and community forums, aiding in troubleshooting and feature implementation made easy.

c. **Compatibility:**

Python's compatibility with multiple database systems, including SQLite and MySQL, ensured a smooth transition when switching databases. The in-built sqlite3 library in Python provided an easy and lightweight solution for database management without requiring additional configuration. We started with MySQL but eventually switched to SQLite because there were some issues with running Mysql connector and PyQt6 at the same time making the code only run in debug mode.

# Chapter 3: Database and Relations

## 3.1 Database Choice

We picked **SQLite** database
**Reason:**
We initially started with MySQL for capabilities in handling large datasets and multi-user environments but later we encountered an issue where MySQL was only operational in debug mode during development. This posed challenges in creating a stable production-ready application.

a. **Ease Integration:** SQLite integrates seamlessly with Python and easily through its built-in SQLite module, enabling fast and efficient database operations without additional configuration.

b. **Lightweight:** It self-contained, serverless database, making it easy to set up and use directly within the application. This eliminated the need for a separate server, simplifying deployment and reducing resource consumption.

c. **Portability:** Due to its small size, it is ideal for a desktop application where lightweight solution are preferred and needed.

d. **Sufficiency:** For a car rental system, SQLite provided all the necessary features for handling data storage and retrieval efficiently. Its support for SQL queries ensured compatibility with the existing database schema and logic developed during the MySQL phase.

## 3.2 Database Relations

a. Branch: The branch relation contains information about the branches of the car rental company, such as their names, locations, and contact details.

Contribution to Functionality:

- Enables the organization of cars based on their physical locations.
- Facilitates easy inventory management by branch.
- Allows customers or administrators to filter available cars by location.

b. Car: The car relation stores details about the vehicles available for rent, including their model, brand, availability status, daily rental price, and the branch they belong to.

Contribution to Functionality:

- Tracks each car's rental status (available or not) and price.
- Links cars to specific branches, helping administrators manage inventory at each location.
- Provides essential information for customer searches, such as car model and brand.

c. Customer: The customer relation holds details about the customers, such as their full name, contact information, email address, and driver’s license number.

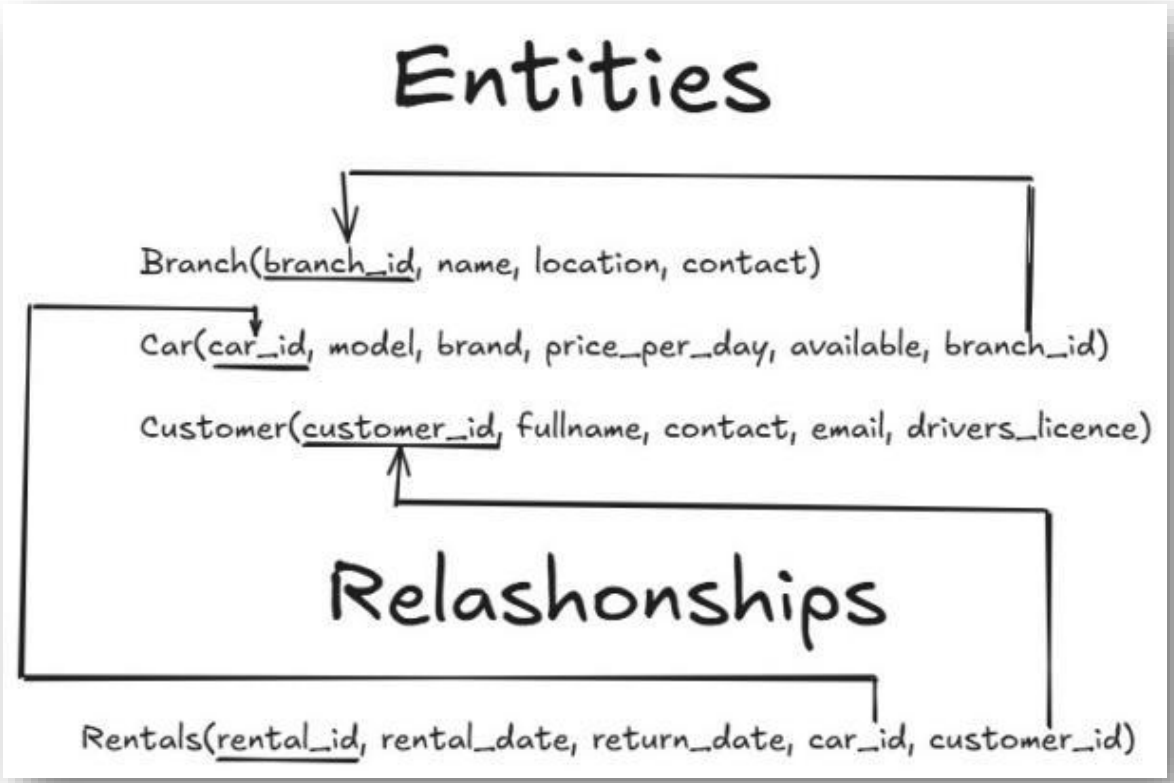
Contribution to Functionality:

- Serves as a central record of all customers renting cars.
- Ensures unique customer identification through email and driver’s license.

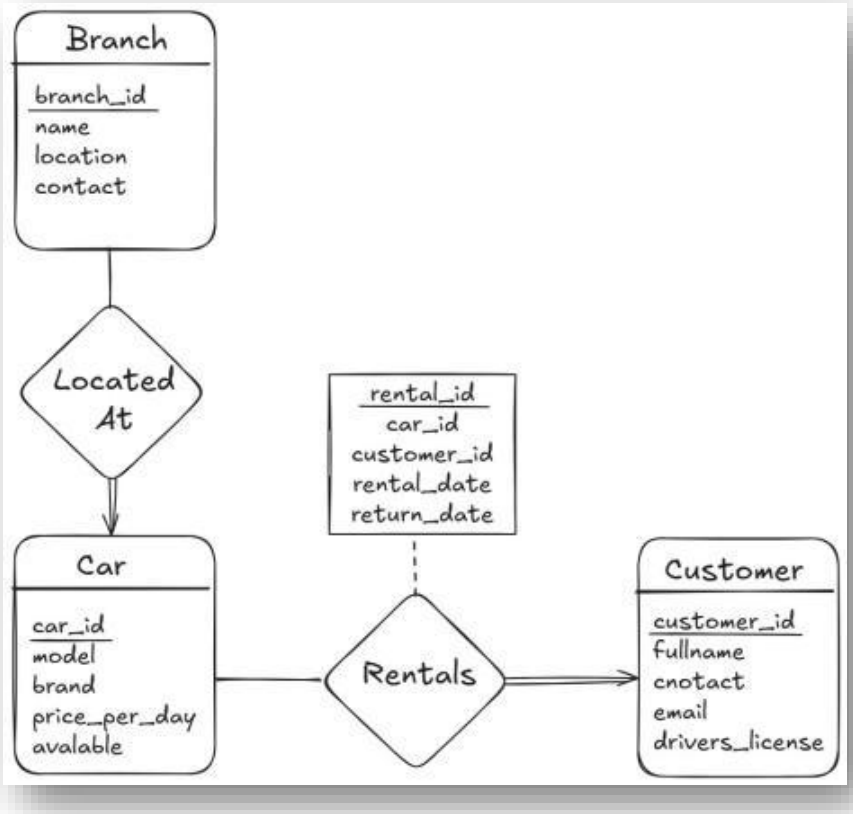
d. Rentals: this relation logs each rental transaction, including the car rented, the customer renting it, the rental start date, and the return date.

Contribution to Functionality:

- Tracks active and completed rentals, ensuring cars are returned on time.
- Links cars and customers, providing a clear record of rental history.
- Allows for automated updates to car availability when a rental starts or ends.


Entities
Branch(branch_id, name, location, contact)
Car(car_id, model, brand, price_per_day, available, branch_id)
Customer(customer_id, fullname, contact, email, drivers_licence)
Relashonships
Rentals(rental_id, rental_date, return_date, car_id, customer_id)

# ER Diagram


Branch
branch_id
name
location
contact
Located At
rental_id
car_id
customer_id
rental_date
return_date
Car
car_id
model
brand
price_per_day
avalable
Rentals
Customer
customer_id
fullname
cnotact
email
drivers_license

# Chapter 4: Functionality

## 4.1 Data Management Operations

### Insertion data:

- Branches: New branches can be added with details like name, location, and contact information.
- Cars: A new car can be added to the system by specifying their model, brand, branch association, rental price, and availability.
- Customers: A customer record are added with personal details such as full name, contact information, email, and driver's license to a car.
- Rentals: New rental transactions are created when a customer rents a car, recording the rental and return dates.

### Deletion:

- Cars: Cars that are sold or retired can be deleted from the system. Any associated rental records are automatically removed.
- Rentals: Records of rent is cleared when car is returned.

### Updating:

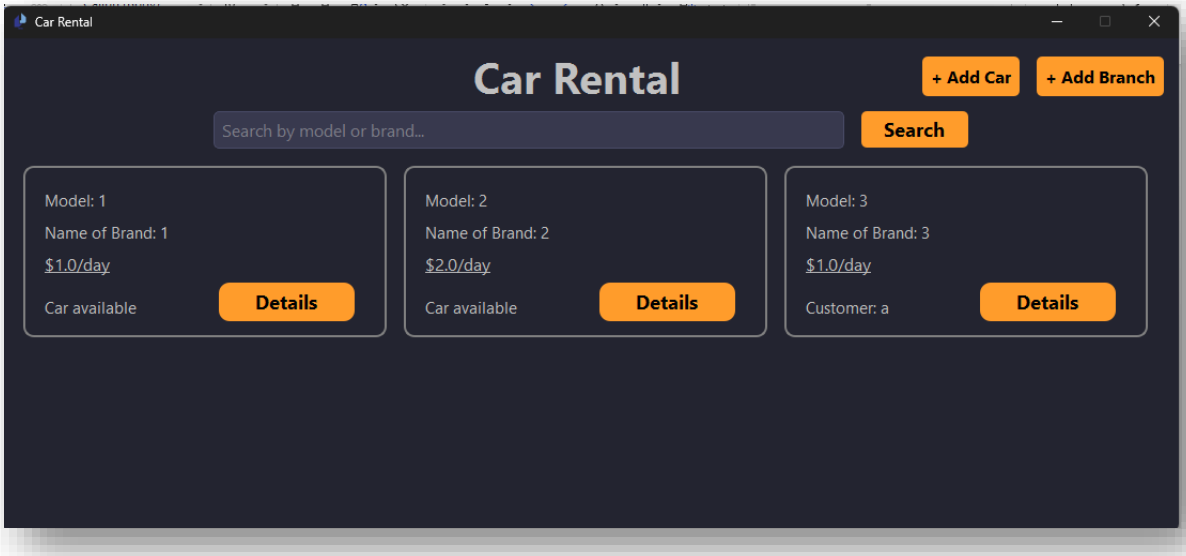
- Cars: Information like availability status, rental price, or branch association can be modified.
- Customers: Contact information or other personal details can be updated.
- Rentals: Rental records can be updated with return dates once the car is returned.

### Listing and Search:

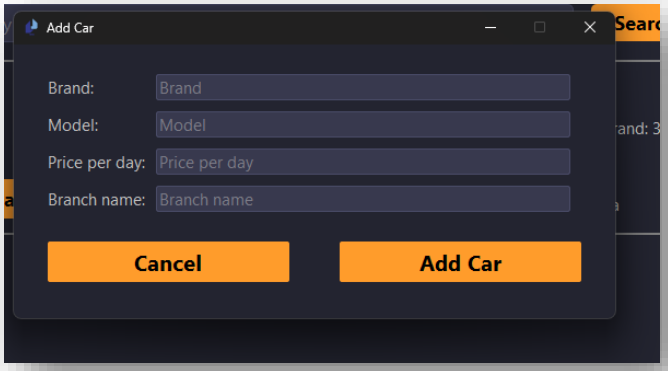
Cars are listed based on when added to the system which is updated in real-time across our database and GUI application, the search uses the brand or make of the car to locate and return the car details of the car specified.

# Chapter 5: Screenshots and Demonstration

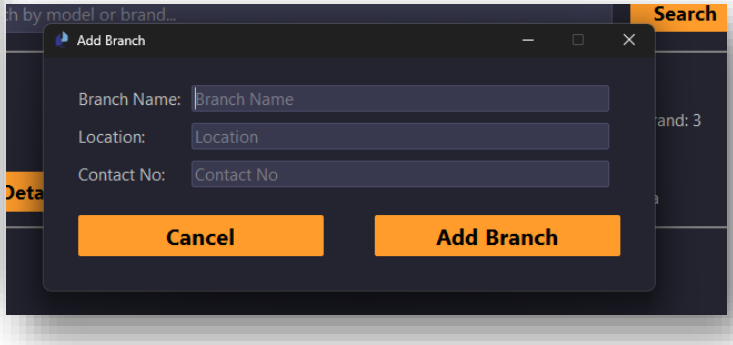
## 5.1 Main Screen - Listing of cars



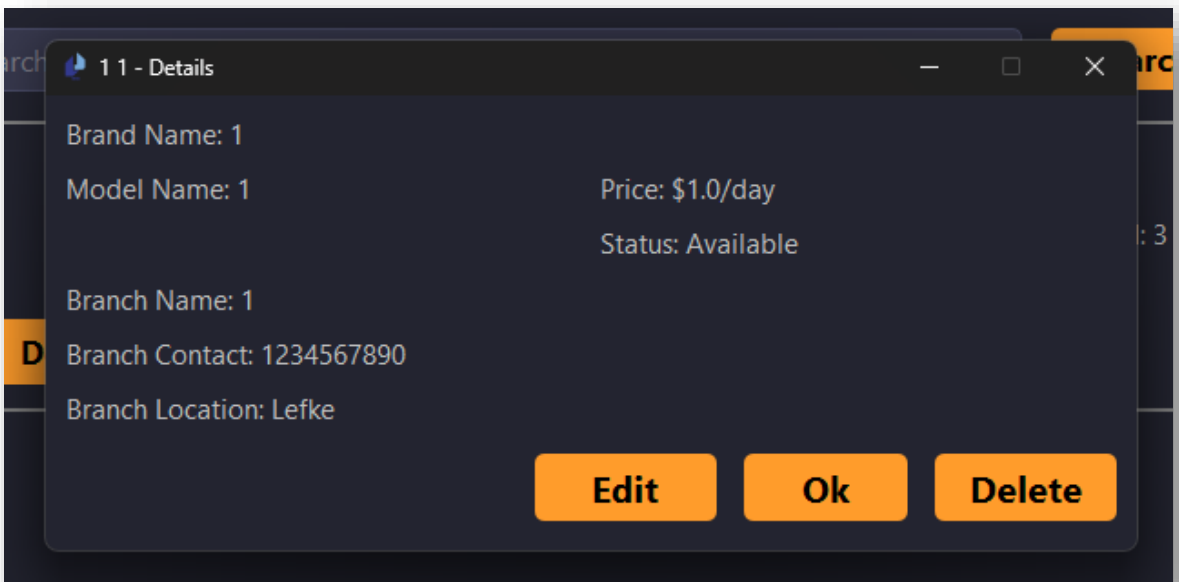

## 5.2 Adding a car



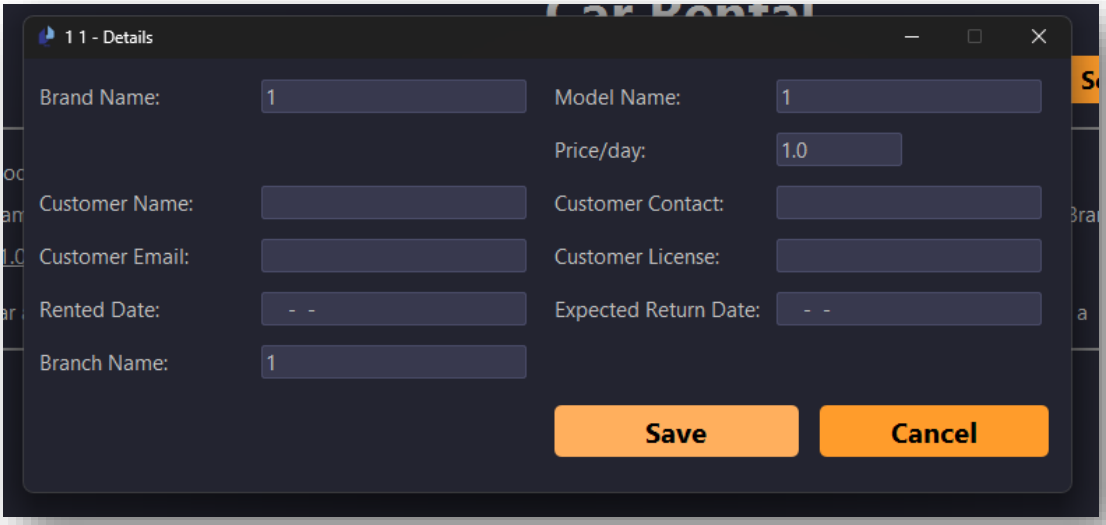

## 5.3 Adding a branch



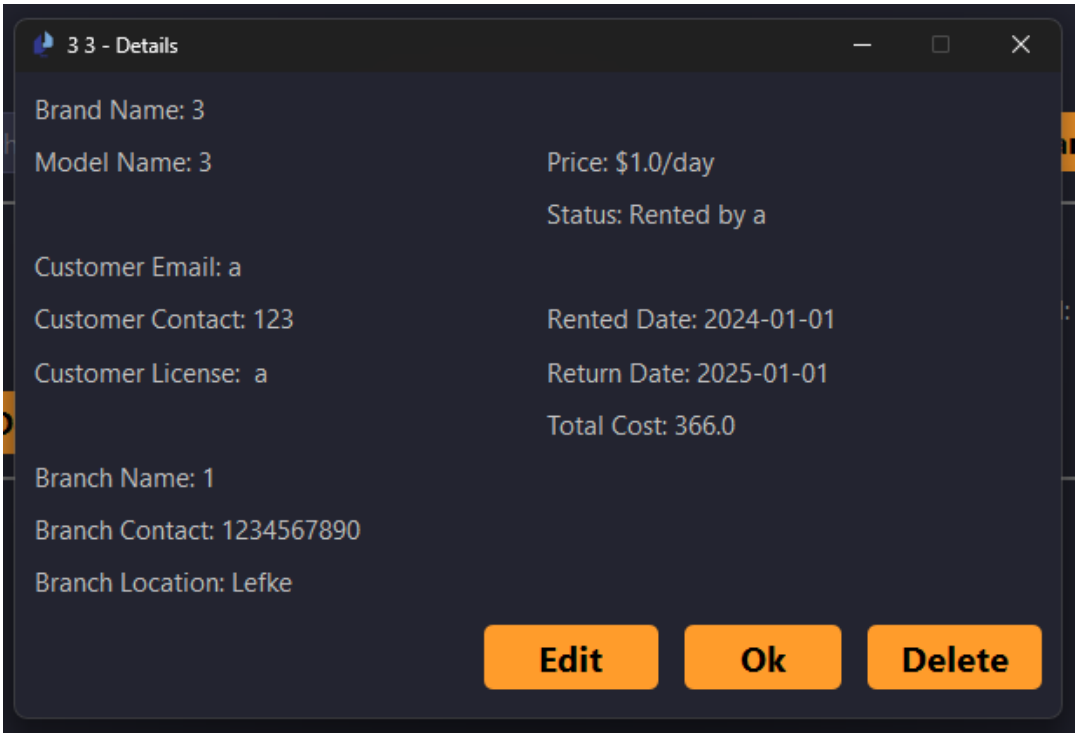

## 5.4 Available car



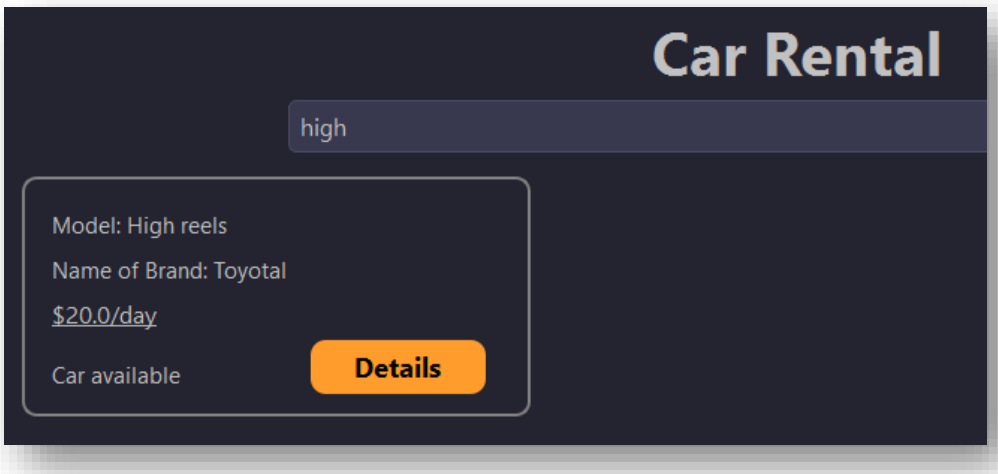
5.5 Renting out a car




5.6 Rented car model with customer detail




5.7 Search



# Chapter 6: Conclusion

## 6.1 Key Findings

The Car Rental System was effectively combining PyQt6, and SQLite to deliver a reliable and user-friendly application. Proper database design with primary keys and foreign key constraints ensured seamless relationships between branches, cars, customers, and rentals.

 Ay-dotcode/Car-Rental

## 6.2 Achievements

Successfully implemented some core functionalities like branch, car, customer, and rental management. Created an intuitive GUI that simplifies operations for administrators.
Transitioned to SQLite for stable and efficient database handling.

## 6.3 Challenges Encountered

Initial issues with MySQL connector showing exit code **3221225477** typically indicates an **access violation** necessitating adjustments and required switching to SQLite for a smooth compilation.
Designing a user-friendly interface with PyQt6 required iterative refinement. Data validation and time constraints were challenging but resolved through effective task management.

Despite challenges, the project achieved its objectives of streamlining car rental operations. The experience gained has been invaluable for tackling similar projects in the future.