<Rent a Car>

Object Design

<1.0>

<16.03.2025>

<Nehir Karameşe>

<Ayşe Korkut>

<Sıla Özcan>

<Ata Kılıç>

<Tuğçe Yeşilyurt>

<Hümeyra Bilgin>

<Eylül Buket Soğukbulak>

<Umut Enis Gögçe>

<Arda Aydoğdu>

<Arda Gül>

<Kaan Berk Kıral>

Prepared for

SOFT3102 Software Engineering Practice



Table of Contents

[1. Introduction 1](#_Toc193209593)

[1.1. Object Design Trade-offs 1](#_Toc193209594)

[1.2. Interface Documentation Guidelines 1](#_Toc193209595)

[1.3. Definitions, Acronyms, and Abbreviations 1](#_Toc193209596)

[1.4. References 2](#_Toc193209597)

[2. Packages 2](#_Toc193209598)

[3. Class Interfaces 3](#_Toc193209599)

OBJECT DESIGN DOCUMENT

Object Design Document (ODD) describes object design trade-offs made by developers, guidelines they followed for subsystem interfaces, the decomposition of subsystems into packages and classes, and the class interfaces. The ODD is **used** to exchange interface information among teams and **as a reference during testing**. The audience for the ODD includes system architects (i.e., the developers who participate in the system design), developers who implement each subsystem, and testers.

Among three approaches to generate ODD, we follow “**ODD embedded into source code**” approach in SE301, since the other methods create many redundancies, inconsistencies.

The initial version of the ODD can be written soon after the subsystem decomposition is stable. Both packages and class interfaces can be generated from source code (comments!) by using a tool, which is named Javadoc. Keeping material for the ODD with the source code enables the developers to maintain consistency more easily and rapidly.

# Introduction

The Rent a Car project aims to offer users a secure and efficient rental service through a web app. The primary motivation behind its development is to make car rentals accessible anytime and anywhere. Ultimately, the project seeks to delight users with the app and transform the car rental experience for the modern digital era.

## Object Design Trade-offs

For the trade-offs:

**• Response Time vs. Scalability:**  
Since our project is dedicated to delivering a fast and seamless rental experience, we’ve opted to prioritize rapid response times even if it means accepting some limitations in scalability.

**• Off-the-shelf vs. Custom Solutions:**  
Under normal circumstances, we would evaluate the merits of integrating pre-built components against developing a tailored solution. However, in our current situation, the only viable option is to develop the system in-house.

## Interface Documentation Guidelines

Classes are named using singular nouns, with underscores separating words when a name consists of multiple words. In contrast, methods are identified by verb phrases, while fields and parameters are designated using noun phrases.

## Definitions, Acronyms, and Abbreviations

**User**: any person who uses the system.

**Admin**: administrator of the system, can add and remove cars.

**RAD:** Requirement Analysis Document.

**GUI:** A GUI or graphical user interface is a form of user interface that allows users to interact with electronic devices through a graphical interface.

**JSON:** or JavaScript Object Notation, is a minimal, readable format for structuring data. It is used primarily to transmit data between a server and web application.

## References

At this time, there is no existing system to be replaced by our new development. Instead, we are now on the third iteration of our solution, which builds on earlier versions by introducing improved features and resolving previous issues.

# Packages

This section describes how the system is modularized into packages and details the file organization. The source code is organized into distinct packages that encapsulate specific subsystems and functionalities, ensuring a clean separation of concerns and ease of maintenance. Although our development environment generates many additional configuration files, we list only the essential paths that contain the core source code. For example, the primary code is located under:

TECHSAN-RAC/

├── index.html

├── assets/

│ ├── css/

│ │ └── style.css

│ ├── js/

│ │ ├── main.js

│ │ ├── register.js

│ │ ├── createCar.js

│ │ ├── displayCars.js

│ │ ├── searchCars.js

│ │ ├── sortCars.js

│ │ ├── filterCars.js

│ │ └── makeReservation.js

│ └── images/

└── config/

└── config.json

# Class Interfaces

