Project Documentation

ISTE-330 Database Connectivity and Access

Auto Market

Document (filename): AutoMarket\_Project\_Documentation

Author(s): Franko Fister, Mladen Orsulic, Dorian Drazic Karalic

Date (last change): 2024-03-24

Version (last version):1.0

|  |  |  |  |
| --- | --- | --- | --- |
| *Version* | *Description of Change* | *Author(s)* | *Date* |
|  |  |  | yyyy-mm-dd |
|  |  |  |  |

Table of contents

[1 Introduction 3](#_Toc35520322)

[1.1 Overview 3](#_Toc35520323)

[1.2 Purpose and Scope 3](#_Toc35520324)

[1.3 Background 3](#_Toc35520325)

[1.4 References 3](#_Toc35520326)

[1.5 Document Overview 3](#_Toc35520327)

[2 Problem Description and Solution Architecture 4](#_Toc35520328)

[2.1 Problem Description 4](#_Toc35520329)

[2.2 Technologies and Architectural Design 4](#_Toc35520330)

[2.3 Database Layer and Database Connectivity Layer 4](#_Toc35520331)

[2.4 Business Layer 4](#_Toc35520332)

[2.5 Presentation Layer 5](#_Toc35520333)

[2.6 Areas of particular concern 5](#_Toc35520334)

[3 Requirements 6](#_Toc35520335)

[3.1 Context 6](#_Toc35520336)

[3.2 Functional Requirements 6](#_Toc35520337)

[3.3 Other (Non-Functional) Requirements 6](#_Toc35520338)

[4 User Documentation 7](#_Toc35520339)

[4.1 Graphical User Interface Design 7](#_Toc35520340)

[4.2 User Manual 7](#_Toc35520341)

[5 Installation, Configuration and Acceptance Testing 8](#_Toc35520342)

[5.1 Installation 8](#_Toc35520343)

[5.2 Configuration 8](#_Toc35520344)

[5.3 Acceptance Testing 8](#_Toc35520345)

[6 Final Remarks and Conclusion 9](#_Toc35520346)

# Introduction

## Overview

The AutoMarket project wants to transform the automotive marketplace through web shops/app platforms. This digital solution aims to make the process of buying and selling vehicles secure, user-friendly, and efficient.

## Purpose and Scope

This paper is the steps it took in making AutoMarket. It's a way to show what we're planning to do and how we're going to do it. It's a guide for us and for the readers who want to understand our work.

## Background

Our team is a mix of sophomore IT students at RIT Croatia. We're working together to make AutoMarket the next big thing.

## References

## Document Overview

Each chapter represents one step we needed to make to develop AutoMarket. From start to finish, we'll talk about all the steps in between and give details on how we're making it happen.

# Problem Description and Solution Architecture

## Problem Description

Now, if you want to buy or sell a car, it can be confusing and complicated. You might have to go to different dealership’s or spend hours scrolling thru websites. We've noticed that sometimes, people decide not to buy due to being scared. Other websites are out there, but they are focused on profit, and not user satisfaction. Our goal with AutoMarket is to make this whole thing as simple as possible. We want you to be able to find/sell a car from home, know exactly what you're getting, and not worry about losing money. Our app will be better because it's going to be really easy to use, super safe, and have everything about every car right there for you.

## Technologies and Architectural Design

For our AutoMarket project, we're using Java for the behind-the-scenes work because it's reliable and can handle a lot of users. For the website part that people interact with, we're using React because it makes the site fast and easy to use. To organize all the information about cars and sales, we're using MySQL Workbench with MySQL database because it's good at storing data and making sure everything is in order. These tools help us build a website that's easy to use, safe, and can grow as more people use it, aiming to make buying and selling cars online as easy as taking the dog to walk.

## Database Layer and Database Connectivity Layer

The database model is composed of multiple related tables with diverse functions of car manufacturers, types of cars that “belong” to them, customers, orders, order details, and administrators. The “manufacturers” table stores general information of manufacturers, such as name, country, website. There is at least one, but more potentially one to many cars associated with each manufacturer. This association is managed through the “cars” table , which maintains information on the car model, year, price and quantity in stock; customers’ information like name, e-mail, phone and address is kept in the “customers” table. Orders are recorded in the “orders” table , which manages data concerning the order date and total amount and status. The “order\_details” table stores items within the order which comprises quantity and the price per unit. Administrators managing the system are stored in the "administrators" table, with attributes such as name, email, and password. These tables are interconnected through foreign key relationships, ensuring data integrity and facilitating efficient querying and management of car sales and related activities.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table Name | Column Name | Data Type | Description | Constraints | Sample Data |
| manufacturers | manufacturer\_id | INT | Unique identifier for manufacturers. | Primary Key | 1 |
| manufacturers | manufacturer\_name | VARCHAR(255) | Name of the manufacturer. | NOT NULL | Tesla |
| manufacturers | manufacturer\_country | VARCHAR(100) | Country where the manufacturer is based. |  | USA |
| manufacturers | manufacturer\_website | VARCHAR(255) | Website of the manufacturer. |  | https://www.tesla.cm |
| cars | car\_id | INT | Unique identifier for cars. | Primary Key | 1 |
| cars | manufacturer\_id | INT | Foreign key referencing manufacturers. | Foreign Key | 1 |
| cars | car\_model | VARCHAR(255) | Model of the car. | NOT NULL | Model S |
| cars | car\_year | YEAR | Year of the car's manufacture. |  | 2022 |
| cars | car\_price | DECIMAL(10, 2) | Price of the car. |  | 79990.00 |
| cars | car\_color | VARCHAR(50) | Color of the car. |  | Red |
| cars | car\_engine | VARCHAR(100) | Engine type of the car. |  | Electric |
| cars | car\_stock\_quantity | INT | Quantity of the car in stock. |  | 5 |
| customers | customer\_id | INT | Unique identifier for customers. | Primary Key | 1 |
| customers | customer\_name | VARCHAR(255) | Name of the customer. | NOT NULL | John Doe |
| customers | customer\_email | VARCHAR(255) | Email of the customer. | UNIQUE | johndoe@example.com |
| customers | customer\_phone | VARCHAR(50) | Phone number of the customer. |  | 555-0101 |
| customers | customer\_address | TEXT | Address of the customer. |  | 123 Elm Street, Springfield |
| orders | order\_id | INT | Unique identifier for orders. | Primary Key | 1 |
| orders | customer\_id | INT | Foreign key referencing customers. | Foreign Key | 1 |
| orders | order\_date | DATE | Date of the order. |  | 2023-03-24 |
| orders | order\_total | DECIMAL(10, 2) | Total amount of the order. |  | 79990.00 |
| orders | order\_status | ENUM | Status of the order. |  | Completed |
| order\_details | order\_detail\_id | INT | Unique identifier for order details. | Primary Key | 1 |
| order\_details | order\_id | INT | Foreign key referencing orders. | Foreign Key | 1 |
| order\_details | car\_id | INT | Foreign key referencing cars. | Foreign Key | 1 |
| order\_details | car\_quantity | INT | Quantity of the car in the order. |  | 1 |
| order\_details | car\_price\_per\_unit | DECIMAL(10, 2) | Price per unit of the car. |  | 79990.00 |
| administrators | administrator\_id | INT | Unique identifier for administrators. | Primary Key | 1 |
| administrators | administrator\_name | VARCHAR(255) | Name of the administrator. | NOT NULL | Franko Fišter |
| administrators | administrator\_email | VARCHAR(255) | Email of the administrator. | UNIQUE | ff1574@rit.edu |
| administrators | administrator\_password | VARCHAR(255) | Password of the administrator. | NOT NULL | Hashed password |

## Business Layer

The Business Layer, implemented in Java, constitutes the core logic responsible for managing business processes such as order processing, inventory management, and user authentication. Through a series of classes encapsulating specific functionalities, such as `OrderProcessor` and `InventoryManager`, this layer orchestrates the flow of data and logic within the application. Using JDBC the Business Layer communicates with the Database Connectivity Layer to perform CRUD (Create, Read, Update, Delete) operations on the underlying database tables. Meanwhile, it interacts with the Presentation Layer by providing interfaces or APIs for seamless communication, ensuring efficient exchange of data and maintaining separation of concerns within the application architecture.

## Presentation Layer

[Provide Presentation Layer description, graphical user interface (GUI) design, including structure, layout and explanations, as well as a description of used technologies. At least several sentences are expected to describe how the Presentation Layer will be constructed, what its purpose is, and how it communicates with Business Layer (below) and the users (clients). You can also include all possible actions, menus, and options. You should start by doing some prototypes or wireframes for **deliverables and milestones #1 to #3** and finish it no later than **deliverable and milestone #4**. In the final **deliverable #5** you can even provide some screenshots.

## Areas of particular concern

[In this chapter, you should provide identification of areas of particular note or concerns. It could be about prerequisites (which must be respected) and assumptions, as well as possible risks for your project. Those could be related to an organization, planning, resources, technologies, and availability, as well as team members. You can describe a plan on how to mitigate those risks. You should start doing it for **deliverables and milestones #1 to #3** and finish it no later than **deliverable and milestone #4**.]

# Requirements

[This chapter should be started for the **deliverable and milestone #0**, and some parts (context and functional requirements) should be finished for the **deliverable and milestone #1**. However, some parts of it will be produced and/or changed later in **deliverables and milestones #2, #3, and #4**.]

## Context

[Provide a description of the application in the broader context, how it will work within the environment of other systems (e.g., payment systems if there is some kind u purchasing involved), with explanations as applicable. The context of a system refers to the connections and relationships between the application and its environment. At least several sentences are expected and it relates to **deliverables and milestones from #1 to #5**.]

## Functional Requirements

[**IMPORTANT:** List, name and explain all key functionalities – there should be approx. 15 functional requirements listed. You should provide a table where each requirement is named (short code or name, could combine letters with numbers) and a detailed description, as well as who is responsible or performing the action associated with this functional requirement. You could also use use-cases (diagrams or descriptions), including use-case names, actors, events flow, exceptions, and special requirements. Include user requirements if necessary (users, roles, privileges) and associate with specific functional requirements. This should be started for the **deliverable and milestone #0**, and the proposal msut be agreed upon and finished with **deliverable and milestone #1**. Later, with the approval of the client, it could be revised in **deliverables and milestones #2 - #4**).]

## Other (Non-Functional) Requirements

[Describe the non-behavioral and non-functional requirements, including hardware and software requirements (e.g., platforms needed to support this application), programming interfaces, and any operational requirements (how the system will run and communicate with environment). You could also provide information about application availability (time of day or week), general performance (how fast it should be in client responses), capacity (how many concurrent users or connections it will support), error handling (how is it handled), conventions used, security and similar if necessary. This should be started for the **deliverable and milestone #1**, and it could be revised in **deliverables and milestones #2 - #5**).]

# User Documentation

[Usually, this chapter should be started later, and at least partially filled for the **deliverable and milestone #3**, and should be finished with **deliverable and milestone #4**. However, some parts of can be changed later, even in **deliverable and milestone #5**.]

## Graphical User Interface Design

[You should provide user design and user experience description, as well as a description of used technologies. This should be started for the **deliverable and milestone #3**, and should be finished with **deliverable and milestone #4**.]

## User Manual

[This should provide expected usage of the available functionalities, could be divided per user roles, and should include screenshots with detailed descriptions. This should be started for the **deliverable and milestone #3**, and should be finished with **deliverable and milestone #4**.]

# Installation, Configuration and Acceptance Testing

[Usually, this chapter should be started later, and at least partially filled with **deliverable and milestone #3 or #4**, and should be finished with **deliverable and milestone #5**.]

## Installation

[Provide a technical manual – prerequisites and installation process description details. Should be finished with **deliverable and milestone #5**.]

## Configuration

[Technical manual should also hold configuration detail and default values for this project to work. Should be finished with **deliverable and milestone #5**.]

## Acceptance Testing

[Some acceptance testing should be performed to determine if the requirements are met – you should describe typical usage and tests to be executed for the application. Should be finished with **deliverable and milestone #5**.]

# Final Remarks and Conclusion

[Usually, this chapter should be started later, and at least partially filled with **deliverable and milestone #3 or #4**, and should be finished with **deliverable and milestone #5**. You should summarize the experiences, both in terms of the produced results and work on the project. List all project deliverables, as well as positive (and negative) experiences and concerns. Comment on missing functionalities and possibilities for improvement and extensions. Estimate project effort (person-hours) and how it was distributed in time and per team roles. This chapter can also include a work log summary for all team members (for each day who did what).]