Project Documentation

ISTE-330 Database Connectivity and Access

Travel Finder

Document (filename): TravelBookingSystemProposal.docx

Author(s): Ivan Miškić, Toni Gašpert, Toni Torbarac, Borna Škec

Date (last change): 13.3.2025

Version (last version): 1.0

| *Version* | *Description of Change* | *Author(s)* | *Date* |
| --- | --- | --- | --- |
| 1.0 | Initial project proposal | Borna Škec | 2025-03-13 |
| 1.1 | Addition to the 2.4 and 3.3 | Toni Gašpert | 2025-03-25 |
| 1.1 | Addition to the 2.5 and 3.1 | Toni Torbarac | 2025-03-25 |
| 1.1 | Addition to 2.6 and 3.2 | Ivan Miskic | 2025-03-28 |
| 1.2 | Finished 2.3 | Ivan Miskic | 2025-04-04 |
| 1.2 | Finished 2.4 | Toni Gašpert | 2025-04-04 |

Table of contents

[1 Introduction 3](#_2gz1q2azgqy0)

[1.1 Overview 3](#_bej1zaxe1b2w)

[1.2 Purpose and Scope 3](#_ukbn8opu631c)

[1.3 Background 3](#_lad2i86u9s0r)

[1.4 References 3](#_l0yu89hvlrxz)

[1.5 Document Overview 3](#_9goh0tzc3fe2)

[2 Problem Description and Solution Architecture 4](#_igelq8eh8s7l)

[2.1 Problem Description 4](#_xa8wnchboacq)

[2.2 Technologies and Architectural Design 4](#_3j7ohy6msff7)

[2.3 Database Layer and Database Connectivity Layer 4](#_a70aoc768bwi)

[2.4 Business Layer 4](#_3dwityclrc3h)

[2.5 Presentation Layer 5](#_so26dvtuqodh)

[2.6 Areas of particular concern 5](#_g8re143zbh97)

[3 Requirements 6](#_jqcf0dbcpc8g)

[3.1 Context 6](#_fa0r8c8z75ns)

[3.2 Functional Requirements 6](#_7irj5aa7cpnj)

[3.3 Other (Non-Functional) Requirements 6](#_w0t8myix2co)

[4 User Documentation 7](#_zgyx5v562wz5)

[4.1 Graphical User Interface Design 7](#_2lzf0kokm4m5)

[4.2 User Manual 7](#_6xuttjlhdf6y)

[5 Installation, Configuration and Acceptance Testing 8](#_k8pj8rkh4u26)

[5.1 Installation 8](#_13zl2idbxjfn)

[5.2 Configuration 8](#_w55oy92s6kfd)

[5.3 Acceptance Testing 8](#_pveurkc6lt4k)

[6 Final Remarks and Conclusion 9](#_vru7ofiqx8uu)

# Introduction

*[Most of this chapter should be part of the project proposal (****deliverable and milestone #0****), and it must be filled entirely in the first project documentation for the client (****deliverable and milestone #1****). This is an introduction to this document and the project – YOU CAN DELETE THIS TEXT LATER]*

Our travel finder app will be a trip and accommodation booking kind of application. It will also provide the user with options where to travel and after the location where to stay and how to get there. We will also mention background information about our team and relevant references. This introduction covers the basic idea of our project and will be further extended for milestone 1.

## Overview

*[Provide an overview of the project and application (that will be developed) and any additional information to place it in the context. Make a high-level description of the business domain and the problem your project and application will solve (and how). Provide a high-level description of the desired functionalities that will be later described in detail. Several sentences are expected (****deliverable and milestone #0 and #1****).]*

The Travel Booking System project aims to develop a similar app to Booking.com, TravelEgo, Airbnb,… Enabling users to choose and book various types of accommodations and traveling destinations. The system will connect travelers with provided options (locations, type of traveling, hotels, apartments, vacation rentals) through some interface. Users will search for accommodations based on location, dates, price, and more options with some detailed info about properties, and some kind of reservation confirmation will be displayed. The application will focus on database connectivity to store and retrieve information about the mentioned data.

## Purpose and Scope

*[Provide an overall description of this document and the project. Explain the purpose of this project and the intended clients. Additionally, discuss the scope of this document (what it describes in the context of the project and also what it will not describe) as well as the intended audience of this document (managers, developers, clients, etc.). Several sentences are expected (****deliverable and milestone #0 and #1****).]*

The purpose of this document is to cover topics of our project. This will demonstrate our ability to implement and design a database application with layered architecture. The intended audience are mainly our collegues and professor who will evaluate the proposal.

The scope of this project includes the development of our app that as mentioned allows users to pick destionations and based on that pick where they will stay, make bookings, and manage their reservations. We will most likely be developing the application using Java with JDBC for database connectivity and will feature some kind of graphical user interface.

## Background

*[Describe who is producing this document and why – you can say a few sentences about your team and why you are doing it. Several sentences are expected (****deliverable and milestone #0 and #1****).]*

We are a team of four: Ivan Miškić, Toni Gašpert, Toni Torbarac, Borna Škec. This project is being developed as our final project for the course with which we want to show understanding of database connectivity concepts and application development using a multi-layered architecture. We have selected a travel booking system as our project because we can make lot of different variations with that much information and try to implement as much of data manipulation with the database as possible and come up with a relevant application.

## References

*[List all necessary references and other associated documents, including links to any documentation (about programming languages, database, drivers, frameworks, libraries, tools, and similar software artifacts), used standards, whitepapers and policies, formats of deliverables you are going to produce, etc. Several sentences and lists with links are expected (****deliverable and milestone #0 and #1****).]*

Java Documentation: <https://docs.oracle.com/en/java/javase/17/>

MySQL Documentation: <https://dev.mysql.com/doc/>

JDBC API Documentation: <https://docs.oracle.com/en/java/javase/17/docs/api/java.sql/java/sql/package-summary.html>

JavaFX Documentation: <https://openjfx.io/javadoc/17/>

Structure: <https://www.oracle.com/java/technologies/dataaccessobject.html>

## Document Overview

*[Provide a short description of the document organization per chapter – what is described in each chapter. Several sentences are expected (****deliverable and milestone #0 and #1****).]*

Chapter 1 - provides an introduction to the project, its purpose, and background information.

Chapter 2 - outlines the problem description and proposed solutions.

Chapter 3 - lists all functional requirements for the application.

The remaining chapters will be completed in future milestones before the other versions and final project conclusions.

# Problem Description and Solution Architecture

*[This chapter should be started for the* ***deliverable and milestone #0****, and some parts (problem description and technologies and architectural design) should be finished for the* ***deliverable and milestone #1****. However, some parts of it will be produced and delivered in* ***deliverables and milestones #2, #3, and #4****, but no later than the final* ***deliverable and milestone #5****.]*

## Problem Description

*[This should be a detailed description of the problem that will be solved. Explain what a problem that you are trying to solve is (in detail), are there any other existing solutions, and how will your solution be in comparison to those (same or better). This should go much more into details than in the previous chapter. At least several sentences are expected (****deliverable and milestone #0 and #1****).]*

The travel applications are evolving significantly with the new technologies and more travelers now prefer to book accommodations online. But many existing platforms are either too complex or lack some kind of filters or fail to provide important information to travelers to make informed decisions.

Our application will aim to create a user-friendly platform that is simple in terms of choosing where to travel and where to stay with easy booking process.   
  
The system will provide:

Location picker with detailed options which will include location details, and staying information.

A straightforward booking process, simple navigation that will improve existing solutions by providing a more user friendly experience while keeping the main booking features expected from a travel application.

## Technologies and Architectural Design

*[****IMPORTANT:*** *Capture important software design decisions and describe them – why did you decide to do it this way and what technologies are you going to use. Do not forget to explain the choice of programming language(s) and layers, and rationale for this design of your application. Describe and sketch Software Architecture (picture) – decomposition of the software into layers/modules/units/components and provide detailed software design description for each part (layer). At least several sentences and an architectural design figure (with layers and technologies) with description are expected (****deliverable and milestone #0 and #1****). ]*

Database layer:

* MySQL relational database for data storage
* lables for locations, accomodation, time, date, descriptions, etc.

Data access layer:

* Java with JDBC for database connectivity
* Data Access Object pattern (needs to be explored more) for database operations
* Implementation of CRUD operations

Business logic layer:

* Java classes implementing business rules and logic
* Handling booking “processing”

Presentation layer:

* JavaFX for creating the user interface
* MVC pattern

Benefits:

* separation between different aspects of the application
* being able to modify one layer without significantly impacting others
* has potential for future scalability and feature additions

## Database Layer and Database Connectivity Layer

*[****IMPORTANT:*** *Provide database structure with ERD and database schema as well as data dictionary explanations of entities (tables) and their characteristics/attributes/properties (columns). At least several sentences are expected to describe the model. There should be a physical database model figure for the chosen database with 10-15 tables in a clear relationship, with a description of each database table. The data dictionary is more than welcome – you can do it in the usual table format in Word, where each row represents one attribute from a database table, and you should have it for all attributes and all database tables. Each row should hold information about that attribute, including table name, column name, the data type in the database, short description, constraints (e.g., primary or foreign key) and restrictions (format of the data), and (if unclear) sample data. You should start doing it for the* ***deliverable and milestone #1*** *and finish it no later than* ***deliverable and milestone #2****).]*

Model Description

Travel Finder database is made out of 13 normalized and interconnected tables that are structured around core concepts of user management, location information, accommodation details, booking processes and payment handling. The database is implemented in a way where primary and foreign keys enforce referential integrity, the structure also includes hierarchical data and junction tables for many-to-many relationships. We use JSON fields to provide extensibility for future requirements without schema changes, while maintaining a clean structure with proper normalization. This design supports our functional requirements while maintaining data integrity.

## Data Dictionary And Table Descriptions:

USERS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| user\_id | INT | Unique ID for users | 2 |
| username | VARCHAR(50) | User login name | Borna\_Skec |
| email | VARCHAR(100) | User email | BornaSkec@gmail.com |
| password\_hash | VARCHAR(255) | Hashed user password | 566432afcc74bb11b23 |
| first\_name | VARCHAR(50) | User first name | Borna |
| last\_name | VARCHAR(50) | User last name | Skec |
| phone\_number | VARCHAR(20) | User phone number | +1234567890 |
| date\_of\_birth | DATE | User date of birth | 03.03.2003 |
| address | VARCHAR(255) | User physical address | Zagreb, 15. Street |
| role | ENUM | User preferences | traveler |
| preferences | JSON | User role in system | {“notifications”: true} |
| created\_at | TIMESTAMP | Account creation time | 2025-03-15 14:40:00 |
| last\_login | TIMESTAMP | time of last login | 2025-03-20 10:45:00 |
| deleted\_at | TIMESTAMP | Account deletion time | NULL |
| is\_active | BOOLEAN | User active or not | TRUE |

The Users table stores the personal info and authentication details for all system users, this includes both travelers and administrators. This table supports user account management features like role assignments, preferences and privacy matches through soft deletes.

LOCATIONS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| location\_id | INT | Unique ID | 3 |
| name | VARCHAR(100) | Name of location | Miami |
| parent\_id | INT | parent reference to location | 4 |
| location\_type | ENUM | Locations type | City |
| latitude | DECIMAL(10,8) | Geographical latitude | 25.7617 |
| longitude | DECIMAL(11,8) | Geographical longitude | -80.1918 |
| description | TEXT | Info about location | Nice city |
| attractions | TEXT | Notable attractions | South Beach |
| geo\_data | JSON | Complex geo info | {“boundary\_points: [[25.7617,-80.1918]]”} |

The Location table is implementing a hierarchical geographical structure for countries, regions, cities and districts that have self-referential relationships. This table includes geographical coordinates, descriptions and attractions to support location-based searching.

ACCOMODATION\_TYPES TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| type\_id | INT | Unique ID | 2 |
| type\_name | VARCHAR(50) | name of type | Motel |
| description | VARCHAR(255) | Desc. of type | Accomodation with rooms |

This table categorizes properties that have specific types such as hotels, apartments, vacation homes and resorts. This standardized classification supports filtering accommodation listings.

AMENITIES TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| amenity\_id | INT | Unique ID | 1 |
| name | VARCHAR(50) | name of Amenity | WiFi |
| icon | VARCHAR(100) | reference of icon | wifi-icon.png |
| category | VARCHAR(50) | amenity category | Tech |

This Amenities table catalogs available features or services that accommodations may offer, such as WiFi, swimming pools or breakfast. Each amenity includes categorization and visual representation through icon references.

ACCOMODATIONS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| accommodation\_id | INT | Unique ID | 1 |
| name | VARCHAR(100) | Accommodation name | Good Hotel |
| type\_id | INT | Accomodation type | 1 |
| description | TEXT | Detailed desc. | Good Accomodation… |
| location\_id | INT | Reference to location | 4 |
| address | VARCHAR(255) | Street Address | Zagreb, 15. Street |
| star\_rating | DECIMAL(2,1) | Rating out of 5 | 4.3 |
| check\_in\_time | TIME | Standard check-in time | 13:00:00 |
| check\_out\_time | TIME | Standard check-out time | 12:00:00 |
| cancellation\_policy | TEXT | Cancellation terms | Free cancelation up to 48h |
| base\_price | DECIMAL(10,2) | Price per night | 299.99 |
| metadata | JSON | Extensive property data | {“avg\_ration”:4.7} |
| is\_verified | BOOLEAN | Official verified or not | TRUE |
| is\_active | BOOLEAN | Available or not | TRUE |
| created\_at | TIMESTAMP | when accommodation was added | 2025-01-10 09:00:00 |
| updated\_at | TIMESTAMP | Last update time | 2025-03-10 13:45:00 |

This table is the central repository for property listings with comprehensive details including name, location, pricing and policies. It supports verification status, rating information and extensible metadata for future attributes

ACCOMODATION\_AMENITIES Table:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| accommodation\_id | INT | Accomodation reference | 1 |
| amenity\_id | INT | Amenity reference | 3 |
| details | VARCHAR(255) | Additional info | high-speed WiFi |
| is\_premium | BOOLEAN | premium or not | TRUE |

This table is a intersection table that implements many-to-many relationships between accommodations and amenities. It provides additional context about how amenities are implemented at specific properties

ROOMS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| room\_id | INT | Unique ID | 1 |
| accommodation\_id | INT | Accomodation Parent | 4 |
| room\_name | VARCHAR(100) | Room name | Room no. 344 |
| description | TEXT | Room Description | Spacious balcony room |
| capacity | INT | Max guests | 3 |
| price\_modifier | DECIMAL(5,2) | Multiplier for base price | 1.25 |
| room\_size | INT | Size in square meters | 45 |
| bed\_type | VARCHAR(50) | Type of Bed(s) | Single person bed |
| features | JSON | Extensive room features | {“minibar”:true} |
| is\_available | BOOLEAN | Available or not | TRUE |

The Rooms table contains information about specific room offerings within accommodations, including capacity, pricing modifiers and features. It supports management with availability status and extensible attributes.

MEDIA TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| media\_id | INT | Unique ID | 1 |
| accommodation\_id | INT | Associated accomodation | 3 |
| room\_id | INT | Specific room ID | 12 |
| file\_path | VARCHAR(255) | Path to media file | /images/acc4/room.jpg |
| caption | VARCHAR(255) | Media description | Room with 2 beds |
| media\_type | ENUM | Type of media content | image |
| is\_primary | BOOLEAN | Whether main display image | TRUE |
| sort\_order | INT | Sequence of display | 2 |
| upload\_date | TIMESTAMP | when was media uploaded | 2025-01-15 09:45:00 |

This table manages the visual and interactive content for accommodations and rooms this includes images videos and virtual tours. This supports presentation management through primary flags and sort order.

BOOKING\_STATUSES TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| status\_id | INT | Unique ID | 1 |
| status\_name | VARCHAR(20) | Name of Status | Confirmed |
| description | VARCHAR(255) | Status Description | Booking Confirmed |

The Booking\_Statuses table defines the possible states of booking such as pending, confirmed, completed or canceled. It provides standardized status tracking for the booking lifecycle.

BOOKINGS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| booking\_id | INT | Unique ID | 1 |
| user\_id | INT | User who made booking | 2 |
| accommodation\_id | INT | Booked Accommodation | 4 |
| room\_id | INT | Specific room booked | 6 |
| reference\_number | VARCHAR(20) | Booking reference | BOOK23421415153532 |
| check\_in\_date | DATE | Starting date of stay | 2025-06-13 |
| check\_out\_date | DATE | End date of stay | 2025-06-20 |
| num\_guests | INT | Number of guests | 3 |
| status\_id | INT | Current booking status | 1 |
| total\_price | DECIMAL(10,2) | Total Cost of booking | 860.00 |
| special\_requests | TEXT | Special Requests | Bottle of wine in room |
| booking\_extras | JSON | Group Booking Information | {“group\_id”:466} |
| created\_at | TIMESTAMP | When the booking was made | 2025-03-18 10:15:00 |
| modified\_at | TIMESTAMP | Last updated timestamp | 2025-03-19 14:20:00 |
| modified\_by\_user\_id | INT | User who made changes | 2 |
| booking\_history | JSON | Log status of changes | [{“form\_status”: 1}] |

This Table records reservation details that include references to users, accommodations, rooms, dates and pricing. It implements a comprehensive tracking of booking changes and supports group bookings.

PAYMENT\_METHODS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| method\_id | INT | Unique ID | 1 |
| method\_name | VARCHAR(50) | Name of payment | PayPal |
| processor | VARCHAR(50) | Payment processor info | PayPal |
| is\_active | BOOLEAN | Method available or not | TRUE |

The Payment\_Methods table catalogs available payment options such as credit cards, PayPal, or bank transfers with processor information. This controls availability of payment methods through active status flags

PAYMENTS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| payment\_id | INT | Unique ID | 1 |
| booking\_id | INT | Associated Booking | 6 |
| method\_id | INT | Payment Method Used | 2 |
| amount | DECIMAL(10,2) | Payment amount | 860.00 |
| currency | CHAR(3) | Currency Code | EUR |
| exchange\_rate | DECIMAL(10,6) | Currency conversion rate | 1.000000 |
| transaction\_id | VARCHAR(100) | External transaction reference | TX2342521324124 |
| payment\_status | ENUM | Status payment | Completed |
| payment\_type | ENUM | Type of transaction | Full |
| payment\_details | JSON | processor data | {“card\_type”:”MasterCard”} |
| payment\_date | TIMESTAMP | When it was processed | 2025-03-18 10:20:00 |
| parent\_payment\_id | INT | Reference to original payment | NULL |

This table records all of the financial transactions that support multiple currencies, installation payments and refunds. It includes processor-specific details and self-referential relationships for payment tracking.

REVIEWS TABLE:

| Field Name | Data Type | Description | Example |
| --- | --- | --- | --- |
| review\_id | INT | Unique ID | 1 |
| booking\_id | INT | Associated Booking | 3 |
| user\_id | INT | User who wrote review | 7 |
| accommodation\_id | INT | Reviewed accomodation | 3 |
| rating | INT | Numeric rating | 4 |
| comment | TEXT | Review Text | Great Place |
| response | JSON | Response to review | {“text”:”Thanks”} |
| response\_by\_user\_id | INT | User who responded | 2 |
| response\_date | TIMESTAMP | When response was added | 2025-06-23 08:15:00 |
| moderation\_status | ENUM | Review approval status | Approved |
| review\_date | TIMESTAMP | Review submission | 2025-07-22 15:45:00 |
| is\_verified | BOOLEAN | Stay verified or not | TRUE |

This reviews table stores the feedback and ratings from users that have integrated management response capabilities and moderation workflow.

## Business Layer

*[Provide Business Layer description, design and explanation, as well as describe connections to Database Connectivity Layer and Presentation Layer. At least several sentences are expected to describe how the Business Layer will be constructed, what its purpose is, and how it communicates with Database Connectivity Layer (below) and Presentation Layer (above). You should start doing it for* ***deliverables and milestones #1 or #2*** *and finish it no later than* ***deliverable and milestone #3****). ]*

Central part of the application which holds all the logic and rules required to process the user actions and interface with the underlying database through the Data Access Layer is the Business Layer. It records the data received from the graphical user interface (Presentation Layer) and makes sure all the data passed through it is validated, processed, and structured just fine.

The implementation of this layer uses Java programming to create a set of plain Java classes that execute the following functionalities: 1.Creating and modifying bookings, 2.Validating travel dates and availability, 3.Calculating costs or discounts, 4.Processing cancellations and modifications, 5.Managing user sessions or temporary data.

The Business Layer maintains two interfaces which are:

* Through JavaFX GUI the Presentation Layer accepts user interaction by processing queries and booking requests while providing output results with confirmation displays.
* Through JDBC DAO the system interacts with MySQL database by employing DAO classes that facilitate database query operations.

Some of the responsibilities of the Business Layer are: 1.User Services, handles user registrations,login and profile updates, endorses password hashing and user validation rules, 2.Accommodation Services, retrieves accommodations by filter, includes logic for checking availability, retrieving amenities and calculating base pricing, 3.Booking Services, manages booking creation, updates and cancellations, checks date conflicts and room availability, 4.Payment Services, so validates payment transactions, handels partial and full payment flows, and supports refunds, 5.Reviwe Service, that allows verified users to submit a review.

Examples of Business logic can be: prevention of overlapping bookings for the same room, verifying the users before allowing them to leave reviews, and applying refund logic.

This organizational approach ensures modularity because changes can be made in one component without impacting the other components of the application. The business rules which include items such as refund policies and seasonal discounts can be adjusted at this layer without needing changes in either the user interface or the database structure.

The main advantage of this design model rests on its ability to separate different functional areas. A service class exists for each business domain including users, bookings, payments and accommodations which serves to create modular units that can be tested separately. The architectural design defines specific boundaries which divide system components by their logical portions and clearly define the duties across different layers.The architecture also promotes scalability. With this structure the application expands while new functions such as notification capabilities or coupon systems and third-party connection points can be integrated without impacting its core functionality.

Another major advantage is maintainability. The business logic remains isolated from user interface and data access code which enables personnel to modify rules through a single location that avoids disturbances to GUI or database queries.

Finally, the design enhances security. The application protects other system levels against vulnerabilities through its business layer centralization of access control validation and user authentication validation and data checking functions.

## 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 09Business 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.*

The outermost layer of the application, and JavaFX will be used to develop an easy-to-use and pleasing-to-the-eye graphical user interface (GUI). It handles all the interactions with the user, ranging from inputting travel information, displaying available hotels, and confirming reservation bookings. Input forms, result tables, confirm dialogues, and filters (location, date, price) will be part of the interface. The layer communicates with the Business Layer directly to receive user input and deliver processed output.

The GUI forwards the input to the Business Layer, for example, when the user tries to create a new reservation or inquires about existing reservations. This is where the user request is validated and the reasoning behind the request is processed. Then the messages, or output, are sent to the user, which is displayed on the GUI.

The GUI will be implemented using the Model-View-Controller (MVC) pattern. Controllers will handle communication between backend logic and JavaFX views, making it maintainable and with good separation. In addition to supporting all the required booking functionality, the UI should be responsive and easy to navigate for travellers.

## 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****.]*

One of the particular concerns could be the level of styling available with JavaFX this could potentially lead to inconsistent visual appearance across the application. In order to mitigate this we can establish a minimal but consistent style guide early in development.

# 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****.]*

| Requirement | Description | Responsible user |
| --- | --- | --- |
| Accommodation browsing | Users can view detailed information about accommodations including descriptions | Travelers |
| Booking management | Users can make, modify, and cancel bookings | Travelers |
| Search picking | Users can recieve search results by specific amenities, property types, star ratings, etc. | Travelers |
| Notification system | Users receive notifications for booking confirmations something like a pop up of conformation | All users |
| Payment simulation | The system will simulate payment processing for bookings | Travelers |

## 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****.]*

The Travel Booking System will be a desktop system that will be accessing a MySQL database using JDBC. Its home environment is its own system where the application can run freely given that Java and the database server are installed. It is simple to install and test locally as it does not depend on web hosting or cloud computing infrastructure.

Although the initial version of the system would itself replicate features like payment processing and notifications, it is intended to be extensible. Later it can be supplemented with services that are handy such as payment gateways, mapping APIs and weather information. Since the app is tiered, such additions can be made without expecting any substantial structural modifications.

The design comprises Presentation Layer (JavaFX GUI), Business functionality Layer (Java classes with main functionality) and Data Access Layer (DAO classes with JDBC) to have an end-to-end solution for vacation accommodation search, booking, and management. Every module will talk to other modules to keep the solution flexible and scalable and offer a seamless experience to the user.

## 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****).]*

### 

| **Requirement ID** | **Requirement Name** | **Description** | **Responsible User** |
| --- | --- | --- | --- |
| 1 | User Registration | Users are able to create an account by providing name, email, password, and contact info. System will validate email formats and password strength. | Travelers |
| 2 | User Authentication | Secure login using email and password. The system will verify credentials against stored data similar to the last exercise. | Travelers, Administrators |
| 3 | User Profile Management | Users will be able to view and edit their profile information, including personal details, and payment methods. | Travelers |
| 4 | Location Search | Search for travel destinations by city, country, region, or point of interest. | Travelers |
| 5 | Advanced Filtering | Some kind of filtering will be applied so we make searching easier. | Travelers |
| 6 | Date Selection | Select check-in and check-out dates. The system will validate date selections. | Travelers |
| 7 | Accommodation Details | Detailed information about accommodations, including descriptions will be available. | Travelers |
| 8 | Booking Creation | Accommodation selection and complete the booking process by providing required information. | Travelers |
| 9 | Booking Confirmation | When successfully booked, the system must generate a unique booking reference number and display a confirmation summary with all booking details. | Travelers |
| 10 | Booking Management | Users must be able to view, modify, and cancel bookings that will also be some kind of pop ups. | Travelers |
| 11 | Booking History | There will be some kind of booking history, with some sorting available for the users to choose from. | Travelers |
| 12 | Payment Processing | The system must simulate payment processing, allowing users to select payment methods and complete transactions securely. | Travelers |
| 13 | Price Calculation | The system must calculate total price based on accommodation rates, selected dates, number of guests, and any special promotions or discounts. | System |
| 14 | Notification System | Some kind of notifications for booking confirmations, upcoming stays, and any changes to existing bookings through the application interface. | System |
| 15 | Content Management | Admins must be able to add, edit, or remove accommodation listings, including updating property details, availability, and pricing. | Administrators |

## 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****).]*

The non-functional requirements set technical limitations together with performance benchmarks and operational parameters which result in a robust and scalable and user-focused Travel Finder application. The core functionalities receive guidance through these technical elements while implementation decisions get supported by them.

Hardware and Software Requirements

Hardware:

* Client machine (minimum specs):
* CPU: Dual-core 2.0 GHz or better
* RAM: 4 GB or more

Software:

* Web server (e.g., Apache Tomcat)
* MySQL server
* Software:
* Java SE 17 SDK
* JavaFX 17+ for GUI development
* MySQL Server 8.0+
* IntelliJ IDEA or Eclipse IDE
* MySQL Workbench (optional for DB management)

Programming Interfaces and APIs:

1.JDBC (Java Database Connectivity):

The Java application uses JDBC to create and run communication processes with the MySQL database.

2.JavaFX API:

JavaFX API includes tools that support developers to create contemporary graphical user interfaces as well as manage user interaction.

3.DAO (Data Access Object) Pattern:

A DAO implementation will abstract database interactions while encapsulating every data access procedure away from SQL code to maintain separation between SQL logic and business functions.

Operational Requirements:

The system operates through its desktop-based standalone application form.

The application starts by linking to the MySQL database where it operates continuously while the database stays online.

The application follows two essential steps to connect to its database because it relies on configuration files or hard-coded connection strings.

Internet access becomes necessary only when implementing third-party APIs which include maps or weather data at a later time.

Performance Requirements:

The application must generate replies for user-directed requests (such as bookings or searches) within one second.

The system will optimize database queries by implementing both indexing and constraints to reduce system response times.

A shared local network using up to 10 simultaneous users will be supported in the multitudinous user mode of the application.

Availability:

The application operates in any moment when clients use it together with Java installation and database service availability.

The version will not require uptime specifications since it operates without remote server hosting.

Error Handling:

The application will utilize structured exception handling techniques for each parts of the system architecture:

* Database layer: handles SQL exceptions and connection issues
* Business logic: handles invalid states and rule violations

Users will receive easily understandable error messages through the UI layer interface.

The application will generate logs for use in debugging purposes and tracing errors.

The system will validate user input at both the client interface and before database entry confirms data consistency.

Security Requirements:(this part changed as the project goes on)

User Authentication:

Basic login system with username/password for travelers and possibly an admin account.

1.Role-Based Access Control (RBAC):

The user system grants admin complete access to additional functions such as listing management yet trails only obtain the ability to search and book.

2.Input Sanitization:

The system will cleanse all user submissions to stop SQL injection and similar cyber breaches.

3.Secure Credentials Handling

# 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).]*