Travel Wizards – a travel booking management application

Proiect realizat de:

Obrocea Traian – 1306A

Cosmin Curuliuc Ștefan – 1306A

Diaconu Rareș-George – 1306A

Huminiuc Simona – 1309A

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 2

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 6

3.3 Software Interfaces 10

3.4 Communications Interfaces 11

4. System Features 11

4.1 System Feature 1 11

4.2 System Feature 2 (and so on) 11

5. Other Nonfunctional Requirements 14

5.1 Performance Requirements 14

5.2 Safety Requirements 14

5.3 Security Requirements 14

5.4 Software Quality Attributes 14

5.5 Business Rules 14

6. Other Requirements 15

Appendix A: Glossary 15

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Diaconu Rares-George | 3.04.2024 | Compiling everyone’s sections | 1.0 |
| Diaconu Rares-George | 25.04.2024 | Updating the functionalities | 2.0 |

# Introduction

## Purpose

The purpose of this document is to specify the software requirements for a travel reservation system that facilitates travel reservation and management for various stakeholders in the tourism industry. Like other platforms already in use, the system is designed to facilitate the booking and management of travel reservations. The main objective of the system is to add functions that facilitate efficient communication between travel agents, transport service providers and boarding agents. This document lists specific functions available for each operator, including ticket management, boarding management, ticket validation, and reservation creation and cancellation.

## Document Conventions

This document follows the IEEE standard formatting for software development. The standard defines a regular formatting this document follows including writing to be done in third-person, passive voice as well as readable and grammatically correct text.

## Intended Audience and Reading Suggestions

This document is tailored for various stakeholders involved in the development, implementation, and testing of Travel Wizards software. These stakeholders include developers, project managers, marketing staff, users, and testers. The document starts off with an overview of the functions and specifications for this software in section 2, then moves on to describe the requirements for interfacing with external hardware and software in section 3. It is suggested that all audiences of this document start with section 2 first to get a general idea of the software requirements. Testers should next read sections 5.1 through 5.4 (performance, safety, security requirements and software quality attributes). This is to get an idea of how the software will affect them and the system they are running it on, as well as the aspirations for quality. Next a tester should read section 3.1 (user interfaces) followed by all of section 4 (system features). Reading the document in this order will give the tester an idea of what to expect in the interfaces at first glance, and then they may test all the individual functions to make sure they adhere to the specifications.

After reading section 2, software engineers should read the remaining sections in order because this document was designed specifically for the purpose of developing the system. The developer needs to get an overall idea in section 2 of the platform. Then, how it needs to interface with everything else in section 3 (so they have an idea of what tools to use and possibly how they should use them). Section 4 is the most important to a software engineer because it describes all the functions of the software in great detail and it will help with making decisions in writing actual code for the system. Section 5 is considered least important but the engineer should still read it to make sure their platform has adhered to the given ideals.

## Product Scope

A comprehensive travel booking system is intended to offer the tourism industry stakeholders a single, efficient platform for effortless booking and travel management. It aims to improve the travel experience by providing users with automated processes, access from anywhere, flexibility in travel planning, easy access to booking information and better collaboration between all parties involved in the travel process.

## References

N/A

# Overall Description

## Product Perspective

Travel Wizards is a sophisticated software solution crafted with C# and utilizes Windows Presentation Foundation (WPF) for its user interface, which is defined using XAML (Extensible Application Markup Language). Unlike its predecessors, which relied on disjointed methods of handling bookings, the Travel Reservation System aims to streamline processes, boost efficiency, and enhance convenience for travel agents, transportation companies and boarding agents. While the Travel Reservation System operates independently, it seamlessly interacts with various systems, such as transportation databases. These integrations ensure compatibility and facilitate the smooth exchange of data between the Travel Reservation System and external parties. Powered by the .NET framework, the system capitalizes on the strength and flexibility of C# to enable efficient code organization and robust integration capabilities with underlying data storage systems. The user interface, designed using XAML in WPF, ensures a visually appealing and intuitive experience for users.

## Product Functions

Major functions that Travel Wizards must perform for its end users are as follows:

* Allow transport companies to introduce new routes for the upcoming days and months.
* Provide a comprehensive display of all the routes that a company operates.
* The transport company has the possibility of changing the price of a route or canceling it altogether.
* Facilitate communication and collaboration between travelers, travel agents, transportation companies, and boarding agents.
* Allow the travel agent to search for the best available route for the client.
* Provide a reservation list for every customer of the travel agent.
* The boarding agent can check-in every passenger that has boarded the plane, ferry, train or bus and at the end has the option to generate a report.

## User Classes and Characteristics

Travel Wizards should be designed to be intuitive to use, thus any employee of a transport company or a travel agent should be able to do their work without any problems. Interfaces must be simple, descriptive, and easy to use, while also being comprehensive in the information that can be relied upon by the database.

## Operating Environment

Since Travel Wizards is being developed in C#, it can be used on all operating systems. Users on a PC, Mac, or some UNIX operating system shouldn't have a problem running the software. It is being

developed in C# 12, and thus the user is required to have a compatible version of the .Net Framework

installed on their system, which can be found free.

## Design and Implementation Constraints

Since the Travel Reservation System is being designed and implemented within a single semester as a project for the Faculty of Automatics and Computers Iasi, time is the primary constraint for implementation. Integration challenges may arise during the process of ensuring seamless connectivity with external systems. Additionally, cross-platform compatibility may be limited by hardware constraints or dependencies on operating systems, thus restricting options for compatibility across various platforms and devices. Adherence to industry standards and regulations for data privacy and security further constrain system design. Moreover, overcoming challenges related to expertise in specific technologies and tools may require additional training or collaboration efforts.

## User Documentation

The user guide will be available in a form accessible via the system’s archives. It will follow industry norms for user guides to guarantee clarity, accessibility, and ease of use for individuals with expertise.

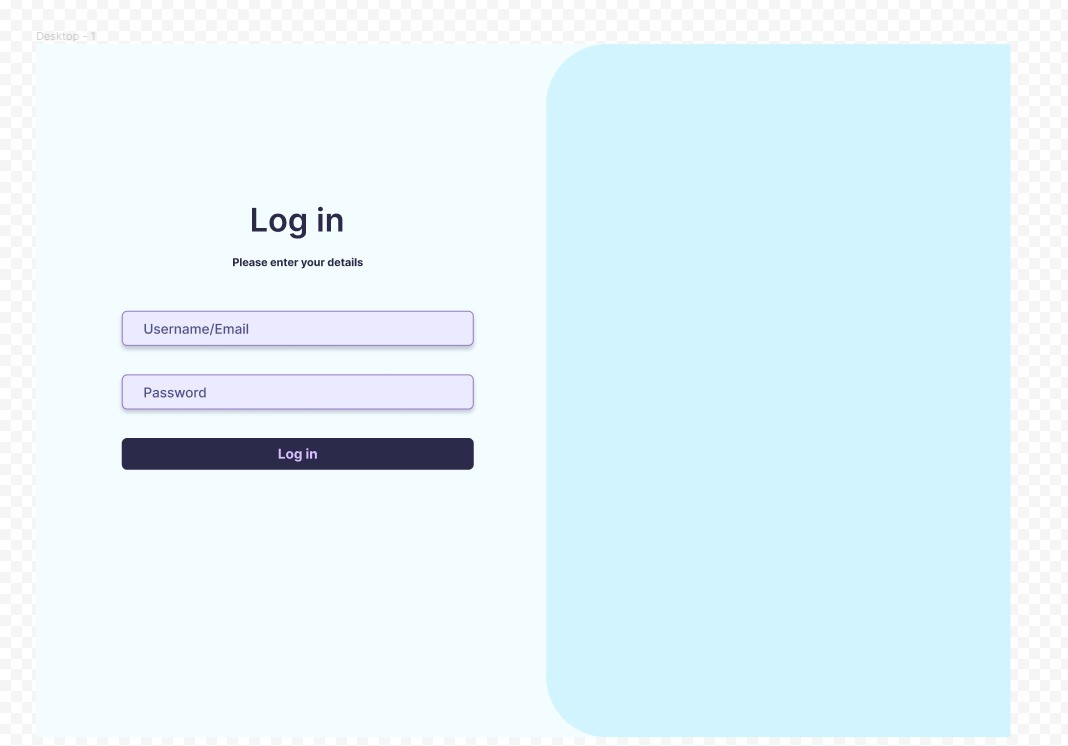
## Assumptions and Dependencies

Our reliance will be on existing software components that have been developed earlier to manage tasks in the Travel Reservation System. Additionally, we expect that the technology stack we have selected including C# and WPF for the user interface will effectively cater to the system’s needs like dealing with updates and limited network communication. Nevertheless, comprehensive testing is crucial to verify that the chosen technology can support the desired intricacies and features of Travel Wizards.

# External Interface Requirements

## User Interfaces

The first interaction that the user has with the software is the login page where the users can securely login into their accounts.



The login page provides users with secure access to their accounts

Depending on the user, there are three main different user interfaces that can pop up:

* A screenshot of a web page

  Description automatically generatedFor the travel agent, a reservation can be made by introducing the name of the traveler, the departure city, and the arrival location. First, all the available routes will be displayed on the right side of the interface and the best route can be picked. Also, there is more information if the travel agent presses on the plus sign to see the layovers and the stops along the route. Then, the travel agent can also see all the routes that the client has reserved on the right side of the screen as well as pressing the plus sign to see all the stops and layovers. Moreover, the travel agent does have the option to cancel the reservation.

Travel Agent Interface: Create reservations by inputting traveler details, select routes, and explore additional information on layovers and stops

Travel Agent Interface: View client reservations, explore route details, and manage reservations, including cancellation options

A screenshot of a web page

Description automatically generated

* A screenshot of a computer

  Description automatically generatedFor the transport company, the employee can see two tabs on the left side of his screen, the first one displays the option to add a new route into the database. The transport company has the possibility to introduce the departure location, the arrival, the type of transport, the price, the departure time, the arrival time, and the frequency of the route. The second tab displays all the routes that the company operates, and the employee can see a lot of information about each route. Also, in this tab the price can be updated, and each route can be canceled. The employee has the option to search for or sort rides based on filters like departure, arrival, or price and to delete all the search results.

A screenshot of a computer

Description automatically generated

Transport company interface: Add New Routes Including Departure, Arrival, Transport Type, Price, Departure Time, and Arrival Time

Transport Company Interface: Manage routes, update prices, and perform various actions such as route cancellation and search filtering

* A screenshot of a computer

  Description automatically generatedFor the boarding agent, at first there will be an interface that will require the agent to input the current location and to choose from the available routes in the next two hours. Then is displayed a list of all the passengers that will board that ride and the option to press a button when that traveler has boarded. At the bottom of the page, there is a button when the boarding is finished, and a report can be generated to see the total number of passengers on board.

Boarding agent interface: inputting the current location and route for the check-in.

A screenshot of a computer

Description automatically generated

Boarding Agent Interface: Track passenger boarding status and generate reports for ride attendance

## Hardware Interfaces

Hardware Interfaces will include a mouse, the keyboard, and the display monitor. The Travel Recognition System requires the basic standard input and output devices used in computing environments. Each user will be able to press a button with the left click of the mouse and to type something with the help of the keyboard.

## Software Interfaces

The Travel Reservation System relies on software connections to work effectively. To begin with it communicates with a PostgreSQL Database Management System (DBMS) using the Npgsql library to store and manage data, like user accounts, bookings, routes, and pricing details. Moreover, the system is built using the C# programming language. Relies on the.NET Framework for functionality. The.NET Framework offers tools and assistance for the Travel Reservation System, making its operation smooth. Additionally, the user interface is developed using Windows Presentation Foundation (WPF), providing a visually appealing and intuitive experience for users. These software connections, which involve integrating Npgsql for PostgreSQL communication ensure interactions and efficient sharing of data allowing the Travel Reservation System to function dependably.

## Communications Interfaces

For the Travel Reservation System to function properly, users must have an established internet connection. This connection is necessary for accessing the system's login page and retrieving data from the PostgreSQL database. The users will access the login page of Travel Wizards Software. Here, they will enter their username and password associated with their account to access the system's functionalities. After successful authentication, they will be directed to the system's specified interface to start using it.

# System Features

## Login Feature

4.1.1 Description and Priority

To access their travel reservation system accounts, users need to verify their identity by logging in. This step is essential for improving user engagement. By requiring users to log in, the system can provide personalized services, maintain security, and track user activity effectively.

4.1.2 Stimulus/Response Sequences

When a user enters their login information the system checks and grants access if it’s correct. If there are errors the system prompts the user to reenter their login details and displays an error message. This process forms the basis of the login features interaction.

4.1.3 Functional Requirements

REQ-LF-1: Users should input their username and password on the system’s login interface.

REQ-LF-2: The system must verify user credentials against the database.

REQ-LF-3: Access to the user account will be granted if credentials are valid.

REQ-LF-4: In case of incorrect credentials, users will be prompted by the system to reenter them through a notification prompt in the form of a message box, ensuring that users can correct any mistakes in their login information.

## 4.2 Travel Agent Reservation Handling

4.2.1 Description and Priority

Travel agents can efficiently handle traveler’s reservations using this feature which's crucial for the systems core operations. It empowers travel agents to effectively assist travelers.

4.2.2 Stimulus/Response Sequences

When the travel agent logs into the software, there will be a feature to make a reservation by entering the traveler's name, departure location, and arrival destination. By pressing the search button, a trigger in the database will retrieve all routes on the screen ordered by the weighted score. This score is calculated as the price divided by the total hours in which the trip will be completed. Also, on this screen, there will be a button called 'Reserve' which, when pressed, will make a reservation, and add it to the table in the database. The travel agent has the option to search by the client's name to find all reservations booked for the traveler. On the different screen that will appear, the travel agent has the option to cancel each one of the reservations by pressing the button ‘Cancel’.

4.2.3 Functional Requirements

REQ-TA-1: The system should offer a user interface for travel agents to input traveler details and search for routes.

REQ-TA-2: The system must find the route in the database with the best weighted score and showcase it to the travel agent in the same user interface.

REQ-TA-3: Travel agents should be able to reserve which route they think is best for their client by clicking on the reserve button. This operation must trigger an insertion into the database to be displayed when the travel agent searches for it.

REQ-TA-4: The system must have the option to search by the traveler’s name to find all the reservations and to be displayed on the second interface available to the travel agent. The system will get this information from the database with the help of a trigger. The travel agent should be able to cancel any reservation at the request of the client.

## 4.3 Transport Company Routes Management

4.3.1 Description and Priority

This feature allows the transportation company to effectively manage its routes, pricing, and cancellations, which are crucial for the smooth operation, financial performance, and satisfaction of customers.

4.3.2 Stimulus/Response Sequences

When the transport company employee logs into the software, there will be the feature to add a new route into the database by pressing the way of transportation, inputting the departure location, departure time, arrival destination, arrival time, the frequency of the route and the period in which the route will be available. By pressing the ‘Add route’ button, a trigger in the database will make this insertion happen. On the second interface available to the transport company, there will be the option to search by departure location or arrival destination and on the right side of the screen all the routes will be displayed from the database. Moreover, the employee will have the opportunity to change the price of any route and to cancel them.

4.3.3 Functional Requirements

REQ-TC-1: The system must allow transport companies to introduce new routes with many specifications about them included, such as departure location and time, arrival destination and time, frequency of the route, type of transportation, and the period in which the route will be operated. The database will manage the safe storage of this information for future use.

REQ-TC-2: Provide search functionality to find a certain route and to adjust the price or to cancel that route altogether. Each operation will trigger one in the database.

## 4.4 Boarding Agent Check-In Feature

4.4.1 Description and Priority

Managing the check in process efficiently for travelers is an importnat aspect of the trip. It directly impacts the boarding of rides ensuring an experience.

4.4.2 Stimulus/Response Sequences

When a boarding agent logs into the software, there will be a list of passengers provided from the database and the option to check in every traveler by pressing the button ‘Aboard’. At the end of the boarding, the agent can generate a report of all the passengers and the status of each one of them.

4.4.3 Functional Requirements

REQ-BA-1: The system should grant access to the passenger list.

REQ-BA-2: The system should enable boarding agents to mark every passenger as aboard or missing.

REQ-BA-3: The system should be able to produce a boarding report when requested using the passenger list.

# Other Nonfunctional Requirements

## Performance Requirements

The system must effectively handle high user traffic, ensuring fast response times for booking reservations and generating reports. It just needs to scale, distribute traffic evenly and minimize database queries using caching. Regular performance testing is required to maintain optimal performance.

## Safety Requirements

The Travel Reservation System ensures the physical safety of users. The system features user-friendly interfaces and clear instructions to minimize the risk of user errors that could lead to accidents or injuries.

## Security Requirements

The main security concern of a travel reservation system is the protection of user information, especially when it comes to password management. We must ensure that passwords are strong and updated regularly to prevent unauthorized persons from accessing user accounts. This means password rules, such as their complexity. We must also constantly check for security holes in our system to fix them and keep user data protected.

## Software Quality Attributes

The Travel Booking System prioritizes robustness, usability, flexibility, and portability. Durability ensures a smooth backup process, usability ensures an intuitive user interface, flexibility enables easy upgrade and portability enables usability in different systems. These features are intended to provide users with a reliable, user-friendly, and customizable platform for efficient travel booking.

## Business Rules

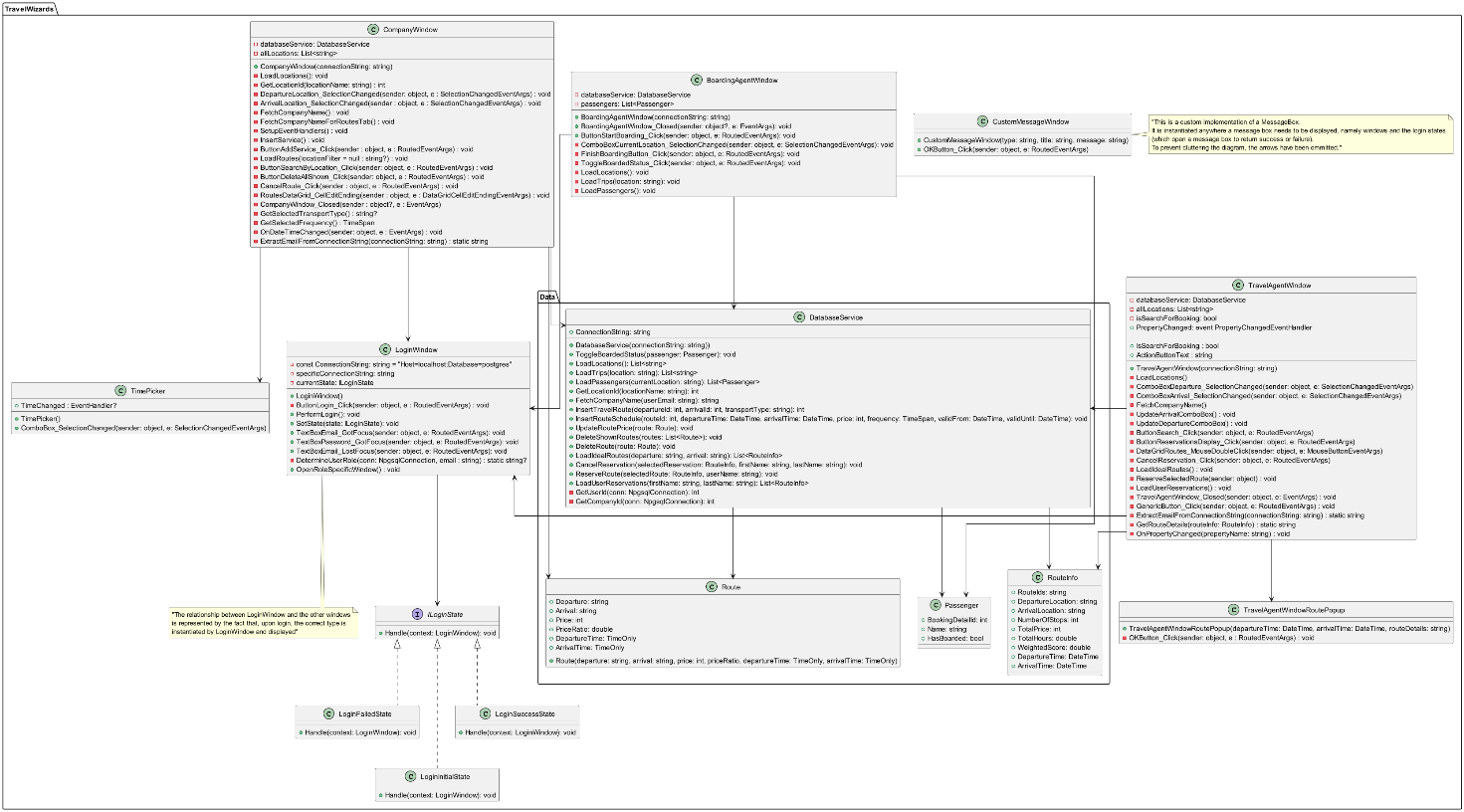
It is the policy of the Development Team to follow and ensure compliance with all Code of Conduct established by faculty. ethical guidelines and professional standards during the development process.

# Other Requirements

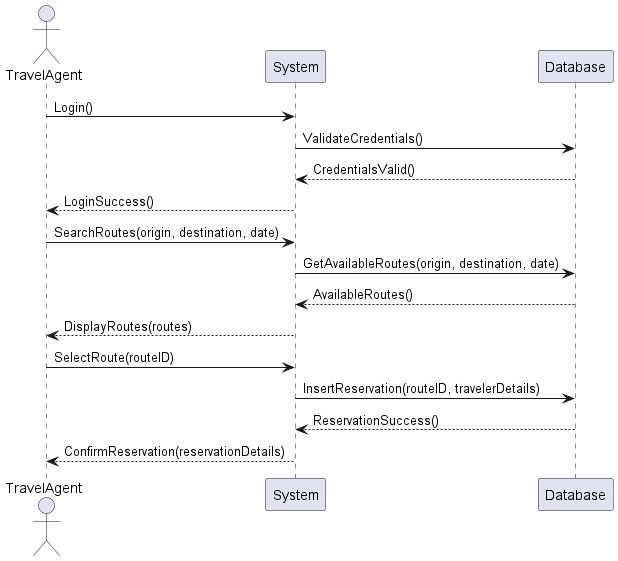
Appendix A: Glossary

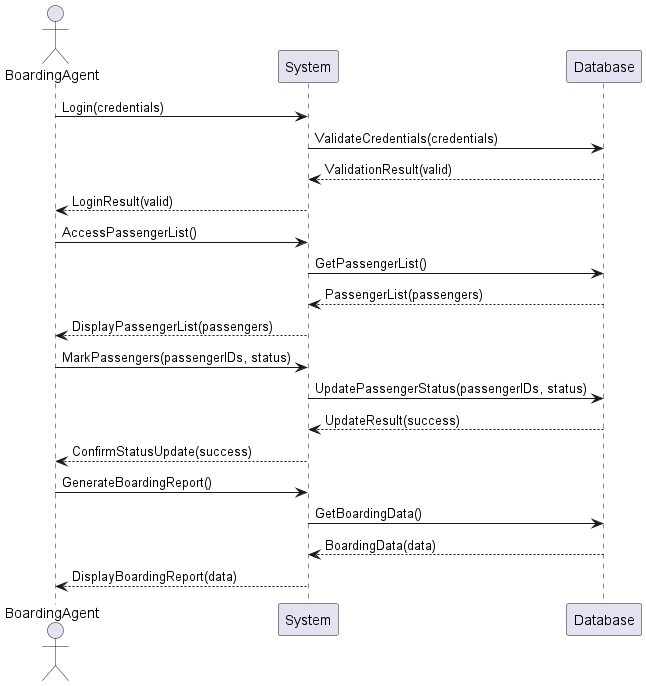
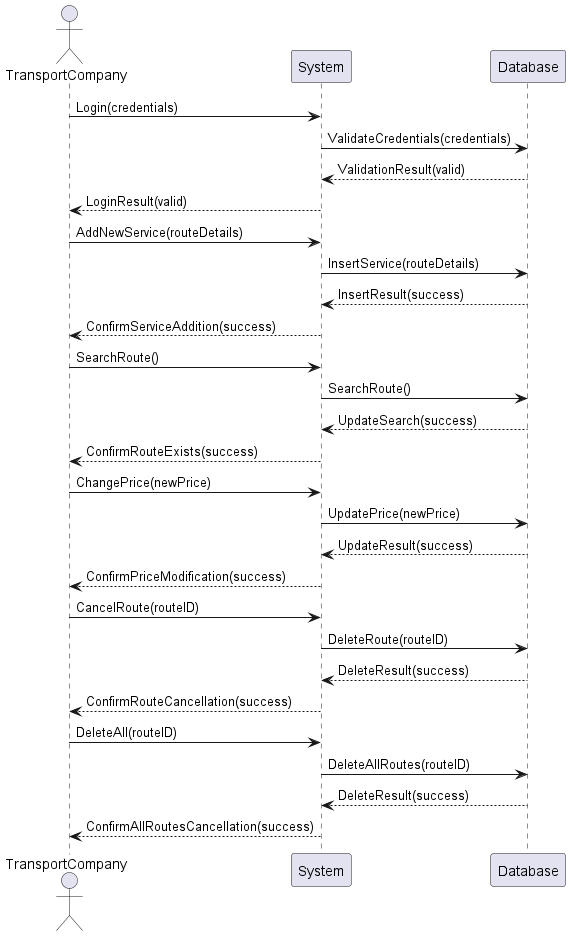
N/A

Diagrama de clase:

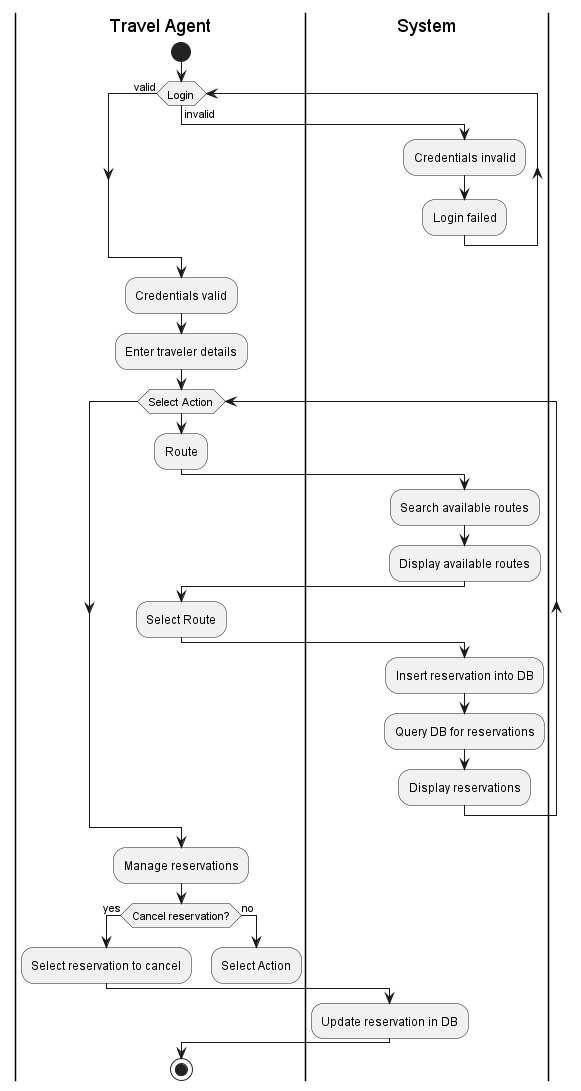
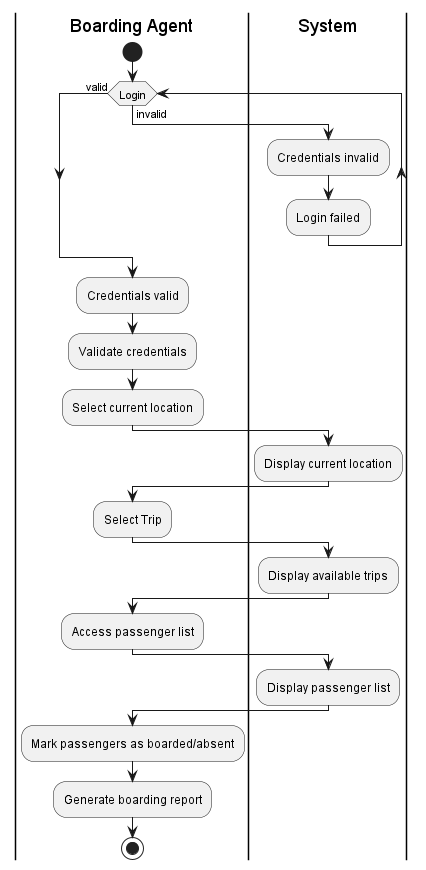


Diagrame de secvente:





Diagrame de activități:



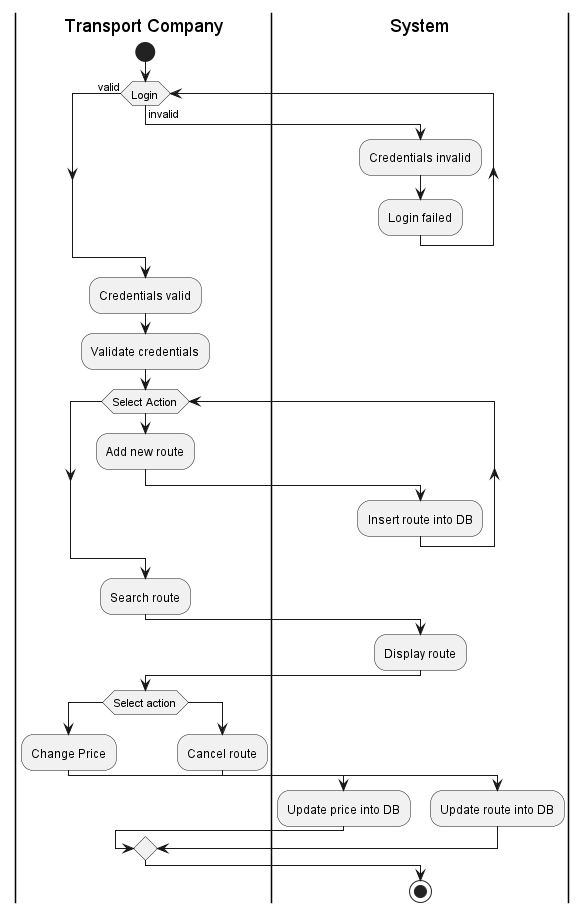
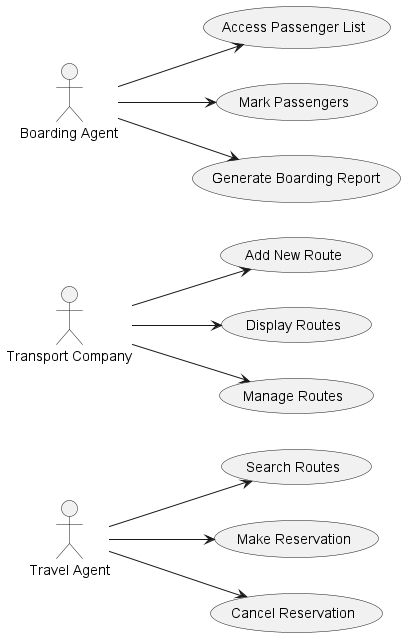


Diagrama de use-case:



Usage:

When opening the application, you will be greeted by a login screen. To use the application, use the email address and password provided to you by your company. Based on your login information, the application will open the appropriate window. As such, depending on your job, refer to the appropriate section.

If you are a travel agent:

As a travel agent, you can make a reservation by entering the traveler's name, the departure city, and the arrival location. Once you input these details, all available routes will be displayed on the right side of the interface, allowing you to select the best route. Additionally, you can click on any route to get more information about it, including stops and layover times.

A screenshot of a reservation

Description automatically generated

You can also see all the routes that the client has reserved on the right side of the screen with the button search. You have the option to cancel any reservation if needed.

A screenshot of a reservation

Description automatically generated

If you manage available transportation offers for your transport company:

As an employee, you can see two tabs on the top of the window. The first one allows you to introduce a new route.

A screenshot of a computer

Description automatically generated

On the left side of the screen you need to choose the type of transport for the route. Then you have to select the departure and arrival locations from the options displayed in the drop-down.

A screenshot of a computer

Description automatically generated

After that, you have to introduce the departure time and the arrival time. When you finish introducing both times, the trip time will be displayed in the middle of arrow. You will also have to input the frequency of the route and the day the service starts and the day the service ends. At the end, you also have to introduce the price.

A screenshot of a computer

Description automatically generated

The second tab displays all the routes that your company operates. You can view detailed information about each route, update prices, and cancel routes if necessary. This tab also allows you to search for or sort rides based on filters like departure, arrival, or price, and you can mass delete all routes you filtered by.

A screenshot of a computer

Description automatically generated

If you are a gate agent or similar:

As an agent, you will first have to select your current location and the trip that you will assist in boarding. Then, you will see a list of all the passengers scheduled to board. You have the option to press a button to mark each traveler as boarded. At the bottom of the page, there is a button to indicate when boarding is finished, allowing you to generate a report that shows the total number of passengers on board.

A screenshot of a computer

Description automatically generated

Ideal route finding algorithm:

/// <summary>

/// Loads optimal routes from the database based on the selected departure and arrival locations.

/// </summary>

/// <param name = "departure" > The departure location.</param>

/// <param name = "arrival" > The arrival location.</param>

/// <returns>A list of optimal routes.</returns>

public List<RouteInfo> LoadIdealRoutes(string departure, string arrival)

{

var routes = new List<RouteInfo>();

using var conn = new NpgsqlConnection(this.ConnectionString);

conn.Open();

const string query = @"

WITH RECURSIVE RouteConnections AS (

SELECT

route\_id,

departure\_location\_id,

arrival\_location\_id,

ARRAY[route\_id] AS route\_path,

1 AS hop\_count,

price AS total\_price,

EXTRACT(EPOCH FROM (arrival\_time - departure\_time)) / 3600 AS total\_hours,

price + (EXTRACT(EPOCH FROM (arrival\_time - departure\_time)) / 3600) AS weighted\_score

FROM travel\_routes

JOIN route\_schedules USING (route\_id)

WHERE departure\_location\_id = (SELECT location\_id FROM locations WHERE full\_name = @Departure)

UNION ALL

SELECT

r.route\_id,

rc.departure\_location\_id,

r.arrival\_location\_id,

rc.route\_path || r.route\_id,

rc.hop\_count + 1,

rc.total\_price + rs.price,

rc.total\_hours + EXTRACT(EPOCH FROM (rs.arrival\_time - rs.departure\_time)) / 3600,

rc.weighted\_score + rs.price + (EXTRACT(EPOCH FROM (rs.arrival\_time - rs.departure\_time)) / 3600)

FROM travel\_routes r

JOIN route\_schedules rs ON r.route\_id = rs.route\_id

JOIN RouteConnections rc ON rc.arrival\_location\_id = r.departure\_location\_id

WHERE NOT r.route\_id = ANY(rc.route\_path)

)

SELECT

ARRAY\_TO\_STRING(route\_path, ' -> ') AS RouteIds,

l1.full\_name AS DepartureLocation,

l2.full\_name AS ArrivalLocation,

hop\_count - 1 AS NumberOfStops,

ROUND(total\_price::numeric, 2) AS TotalPrice,

ROUND(total\_hours::numeric, 2) AS TotalHours,

ROUND(weighted\_score::numeric, 2) AS WeightedScore,

rs.departure\_time,

rs.arrival\_time

FROM RouteConnections

JOIN locations l1 ON RouteConnections.departure\_location\_id = l1.location\_id

JOIN locations l2 ON RouteConnections.arrival\_location\_id = l2.location\_id

JOIN route\_schedules rs ON rs.route\_id = RouteConnections.route\_id

WHERE arrival\_location\_id = (SELECT location\_id FROM locations WHERE full\_name = @Arrival)

ORDER BY total\_price, weighted\_score";

using var cmd = new NpgsqlCommand(query, conn);

cmd.Parameters.AddWithValue("@Departure", departure);

cmd.Parameters.AddWithValue("@Arrival", arrival);

using var reader = cmd.ExecuteReader();

while (reader.Read())

{

routes.Add(new RouteInfo

{

RouteIds = reader.GetString(0),

DepartureLocation = reader.GetString(1),

ArrivalLocation = reader.GetString(2),

NumberOfStops = reader.GetInt32(3),

TotalPrice = reader.GetInt32(4),

TotalHours = reader.GetDouble(5),

WeightedScore = reader.GetDouble(6),

DepartureTime = reader.GetDateTime(7),

ArrivalTime = reader.GetDateTime(8)

});

}

return routes;

}

Input validation for route insertion:

/// <summary>

/// Inserts a new service into the database based on user inputs.

/// </summary>

private void InsertService()

{

Window messageBox;

if (this.ComboBoxDepartureLocation.SelectedItem == null || this.ComboBoxArrivalLocation.SelectedItem == null ||

!this.DatePickerStartDate.SelectedDate.HasValue || !this.DatePickerEndDate.SelectedDate.HasValue ||

this.TimePickerDepartureTime.Time == TimeSpan.Zero || this.TimePickerArrivalTime.Time == TimeSpan.Zero)

{

messageBox = new CustomMessageWindow("info", "Input Error", "Please fill all required fields.");

\_ = messageBox.ShowDialog();

return;

}

var departureLocation = this.ComboBoxDepartureLocation.SelectedItem.ToString();

var arrivalLocation = this.ComboBoxArrivalLocation.SelectedItem.ToString();

if (departureLocation is null)

{

Window errorMessageBox = new CustomMessageWindow("error", "Error", "Please select a departure location");

\_ = errorMessageBox.ShowDialog();

return;

}

if (arrivalLocation is null)

{

Window errorMessageBox = new CustomMessageWindow("error", "Error", "Please select an arrival location");

\_ = errorMessageBox.ShowDialog();

return;

}

var departureLocationId = this.GetLocationId(departureLocation);

var arrivalLocationId = this.GetLocationId(arrivalLocation);

var transportType = this.GetSelectedTransportType();

var frequency = this.GetSelectedFrequency();

var departureDateTime = this.DatePickerStartDate.SelectedDate.Value.Date + this.TimePickerDepartureTime.Time;

var arrivalDateTime = this.DatePickerEndDate.SelectedDate.Value.Date + this.TimePickerArrivalTime.Time;

if (arrivalDateTime < departureDateTime)

{

arrivalDateTime = arrivalDateTime.AddDays(1);

}

var validFrom = this.DatePickerStartDate.SelectedDate.Value.Date;

var validUntil = this.DatePickerEndDate.SelectedDate.Value.Date;

var price = int.Parse(this.TextBoxPrice.Text, System.Globalization.CultureInfo.InvariantCulture);

try

{

using (var conn = new NpgsqlConnection(this.databaseService.ConnectionString))

{

conn.Open();

using (var tran = conn.BeginTransaction())

{

var routeId =

this.databaseService.InsertTravelRoute(departureLocationId, arrivalLocationId, transportType);

this.databaseService.InsertRouteSchedule(routeId, departureDateTime, arrivalDateTime, price,

frequency, validFrom, validUntil);

tran.Commit();

}

}

messageBox = new CustomMessageWindow("info", "Success", "Service added successfully!");

\_ = messageBox.ShowDialog();

this.LoadRoutes();

}

catch (Exception ex)

{

messageBox = new CustomMessageWindow("error", "Database Error", $"Failed to add service: {ex.Message}");

\_ = messageBox.ShowDialog();

}

}