



Student: Octavian-Mihai Matei

Group: 30431

# Travelling Agency

---

## Table of contents

---

- [Subject specification](#)
- [Use Cases](#)
- [Database](#)
- [Class Diagram](#)
- [Testing](#)
- [Conclusions](#)

# QR Code

---



# Specifications

---

The goal of this project is to design and implementation of a vacation seeking application having the following criteria: Now, the application needs to accommodate only 2 types of users (lucky you):

- Regular User/Vacay seeker
- Travelling Agency

The Travelling Agency should be able to:

1. add vacation destination
2. add vacation packages for a specific destination a. should contain information: name, price, period, extra details, number of people that can book the vacation
3. edit an existing vacation package
4. delete an existing vacation package
5. view all its listed vacation packages (with status: BOOKED, NOT\_BOOKED, IN\_PROGRESS)
6. delete vacation destination

The Regular User should be able to:

1. register on the platform using some credentials (username/email - unique & password)
2. login on the platform
3. view all available vacation packages
4. filter vacation packages by destination/price/period
5. book a vacation
6. view all its booked vacations

# Use cases

---

The project has two main users: normal user, who books packages, and the agency account, which creates and manages packages

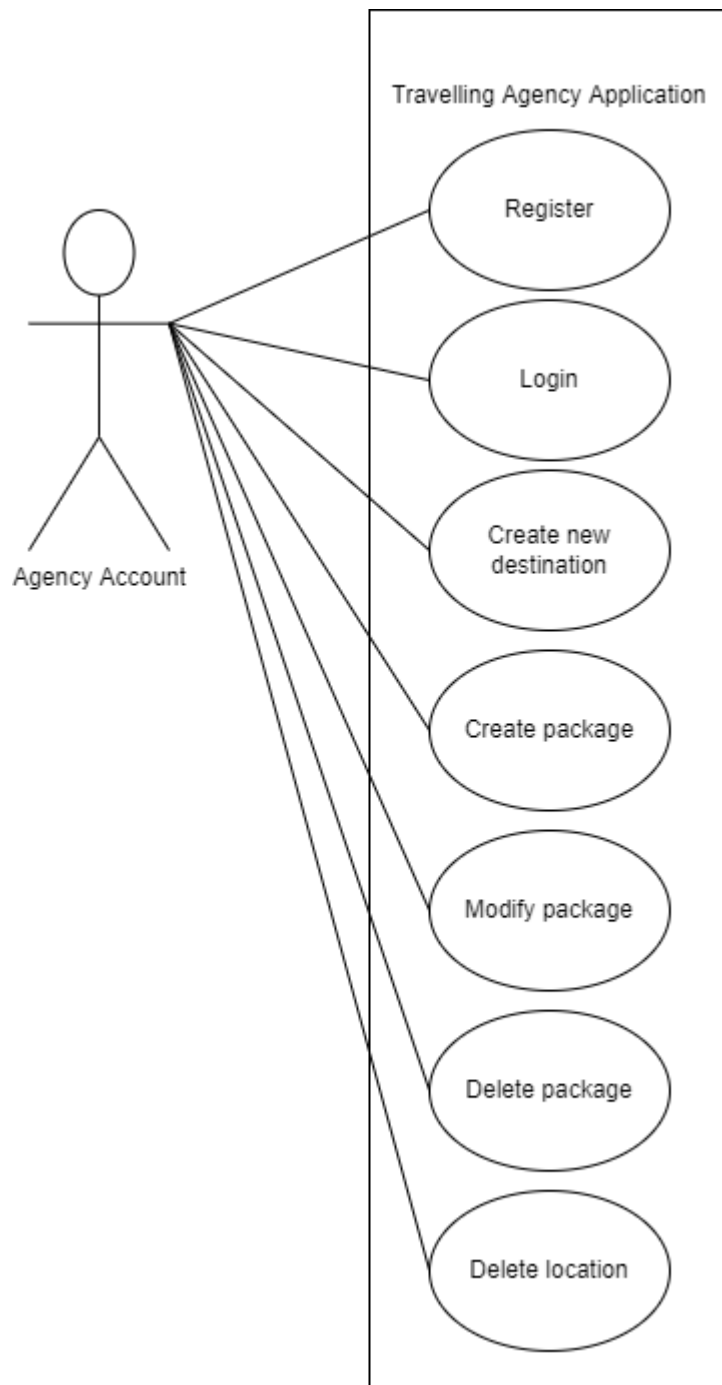
## Use-case Identification

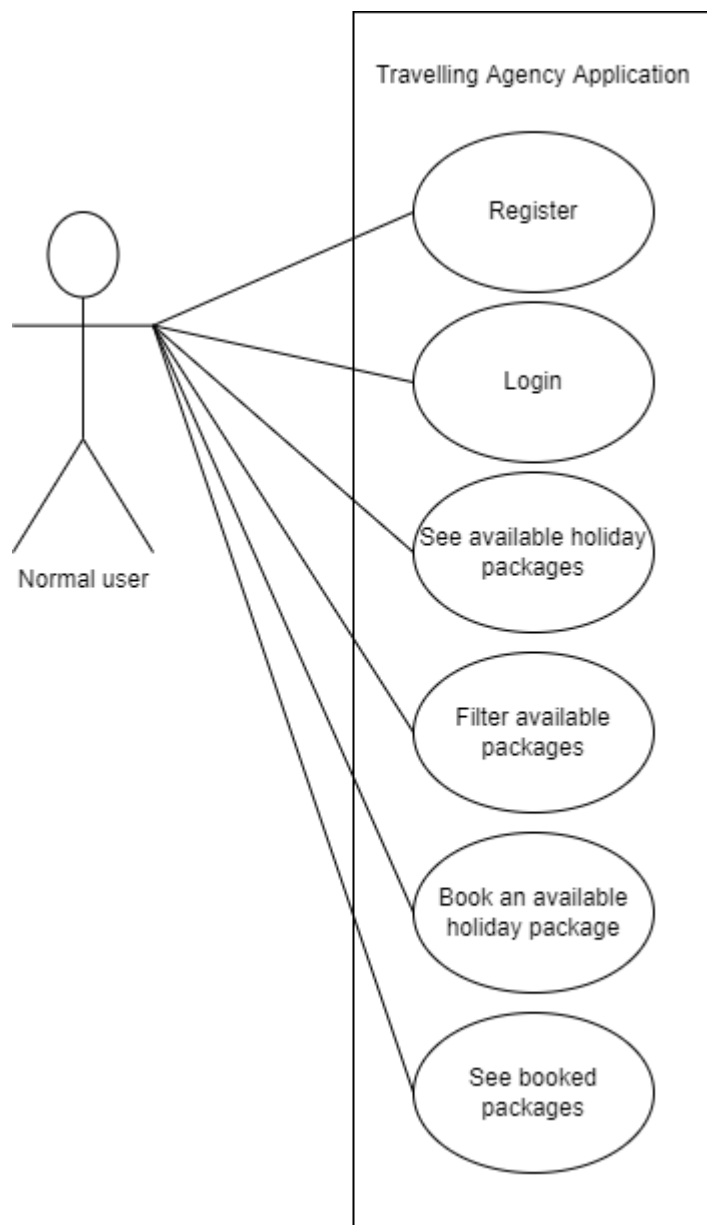
Use case name: Create vacation package Level: An agency will create a package Main actor: Agency Main success scenario: First, they will login, select the desired location, fill in the information required and execute the addition.

Use case name: Delete location Level: An agency will delete a location Main actor: Agency Main success scenario: First, they will login, select the desired location from a drop down menu and press the delete location. This action should also delete all the aparitions of the said location from database and all the packages that include this location

Use case name: Book package Level: A client will book a package Main actor: Client Main success scenario: First, they will login, select the desired filters. After this step, the client should only see the desired information and after selecting the package, they should press a submit button. Following, the user should see the package in the already booked section.

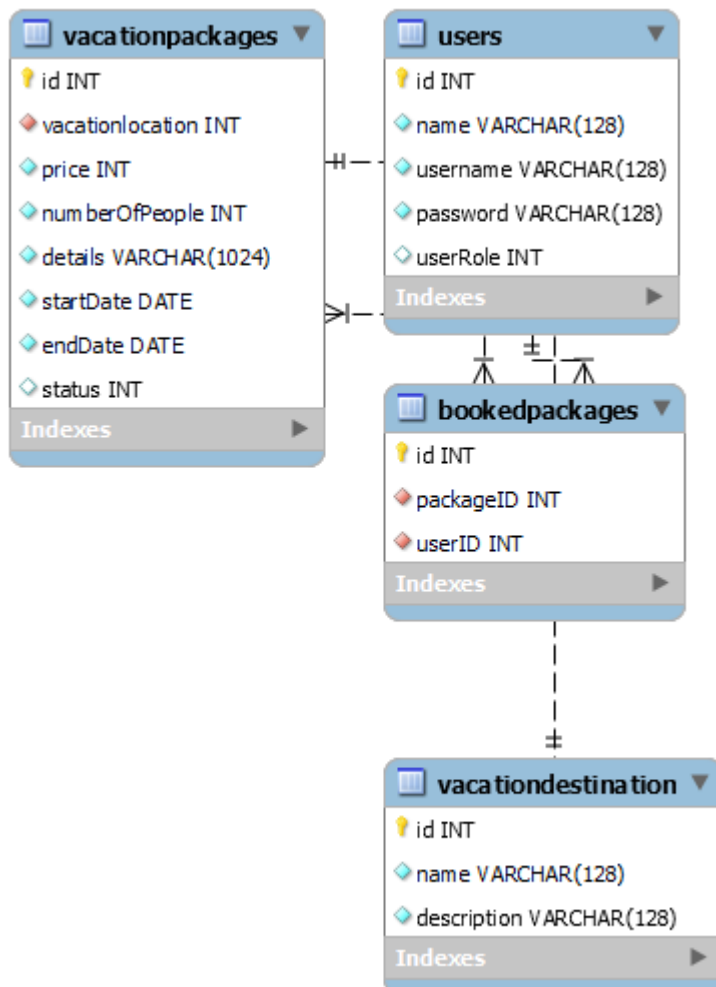
## UML Use-Case Diagram





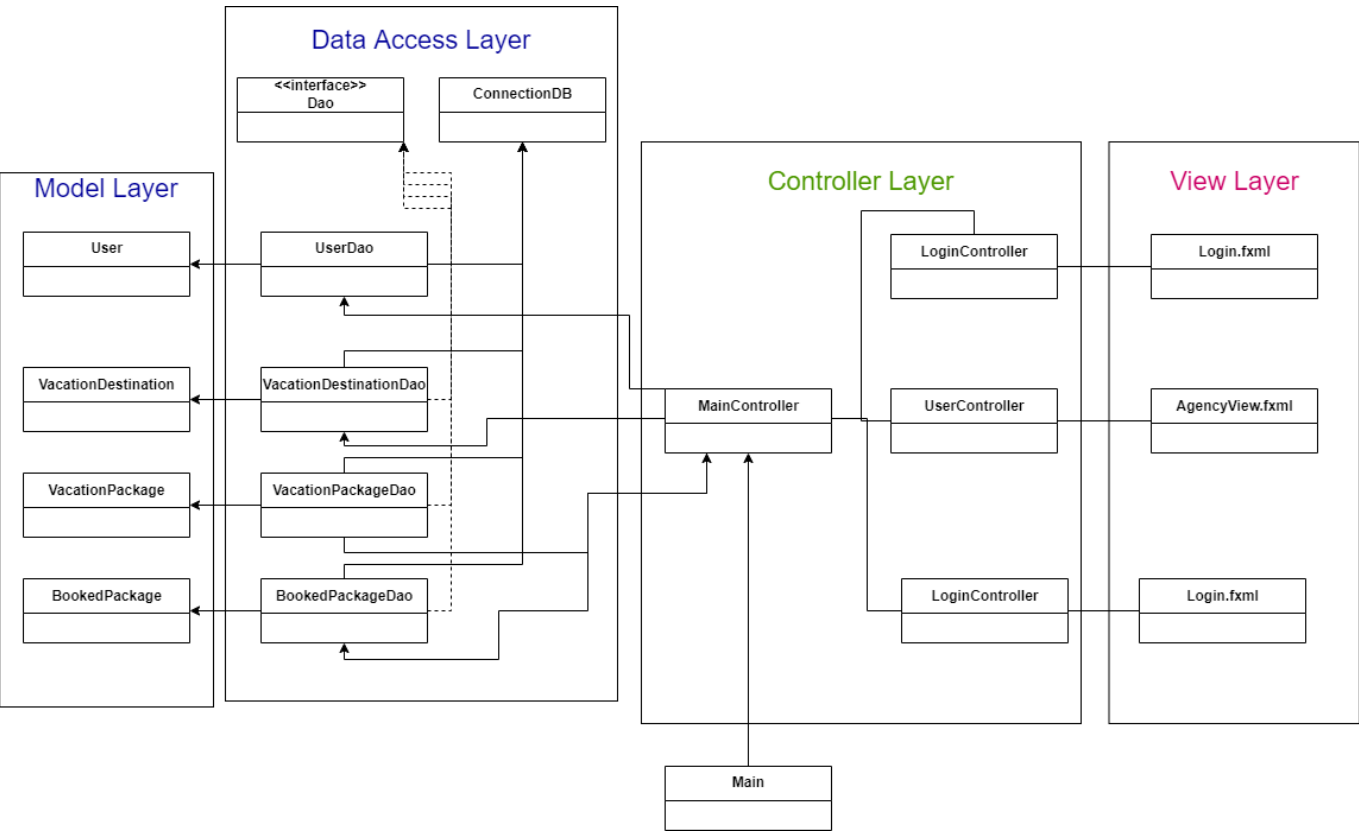
# Database

The application contains four databases that interact with each other, in order to provide cascading features in case of creation/modification/delete operations





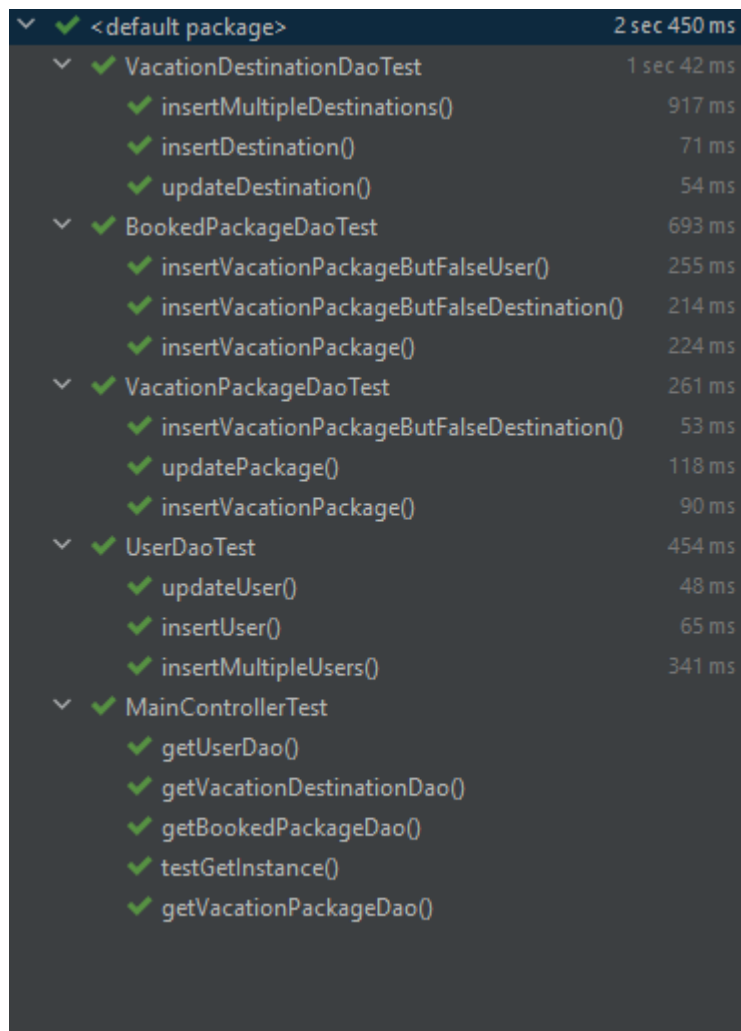
# Class Diagram



# Testing

---

In order to test the implementation of the basic operations, a suite of unit tests was created and passed as seen in the following screenshot:



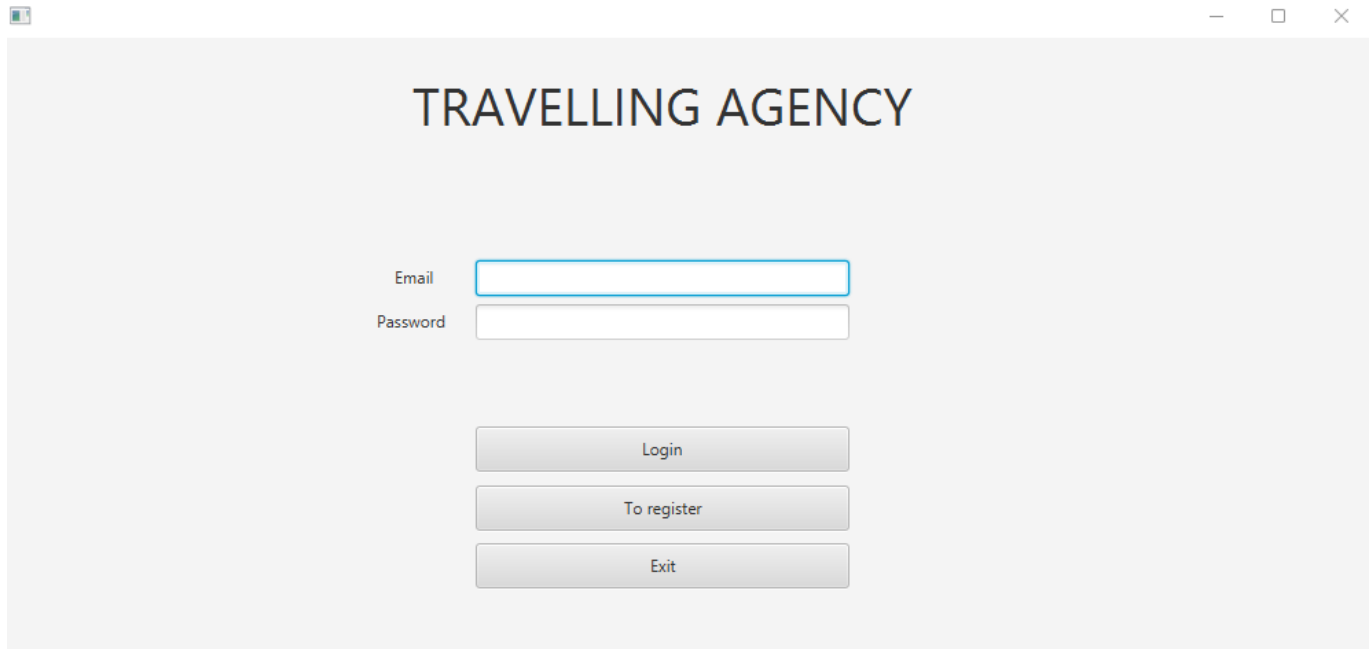
The screenshot displays a test runner interface with a dark background. It shows a hierarchical list of tests, all of which are marked with a green checkmark, indicating they passed. The tests are organized into a tree structure, with expandable/collapsible icons (downward arrows) next to each level. The total execution time for the entire suite is 2 seconds and 450 milliseconds. Individual test methods and their durations are also listed.

✓ <default package>	2 sec 450 ms
✓ VacationDestinationDaoTest	1 sec 42 ms
✓ insertMultipleDestinations()	917 ms
✓ insertDestination()	71 ms
✓ updateDestination()	54 ms
✓ BookedPackageDaoTest	693 ms
✓ insertVacationPackageButFalseUser()	255 ms
✓ insertVacationPackageButFalseDestination()	214 ms
✓ insertVacationPackage()	224 ms
✓ VacationPackageDaoTest	261 ms
✓ insertVacationPackageButFalseDestination()	53 ms
✓ updatePackage()	118 ms
✓ insertVacationPackage()	90 ms
✓ UserDaoTest	454 ms
✓ updateUser()	48 ms
✓ insertUser()	65 ms
✓ insertMultipleUsers()	341 ms
✓ MainControllerTest	
✓ getUserDao()	
✓ getVacationDestinationDao()	
✓ getBookedPackageDao()	
✓ testGetInstance()	
✓ getVacationPackageDao()	

# Conclusions

---

The application taught me how to build a layered application using databases. In the same time, it taught me how to respect stakeholder's criteria and how to manage my time. Some screenshots in the application:



Location name

Location Description

Save new location

Location name

Delete location

Location name

Barcelona

Price

Number of people

Details

Start date

End date

Save new Package

Location name

Alabala

Price

30

Number of people

10

Details

100034

Start date

1970-01-01

End date

2024-01-01

Status

In progress

Modify Package

Delete Package

Logout

Initialize

Available vacations

Price Range

None

Destination

Bucharest

Search

Location: Bucharest, Details: Da, Price: 100, From: 2022-03-09, To: 2022-03-15, Status: Fully booked

Book

My vacations

-- Location: Madrid  
Details: 10  
Price: 1000  
From: 2022-02-16  
To: 2022-03-01  
Status: Not booked

-- Location: Madrid  
Details: 10  
Price: 1000  
From: 2022-02-16  
To: 2022-03-01  
Status: Not booked

Logout