

**Yaşar University**  
**Spring, 2023 - 2024**  
**SE 2224 - Software System Analysis**  
**Final Project Report: Software Requirements Specifications Document (SRS)**

<b>Student Name:</b>	Yağmur Sabırlı
<b>Student No:</b>	21070006017
<b>Department Name:</b>	Software Engineering
<b>Course Section No:</b>	SE2224.1

This template is prepared based on the IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998).

## Table of Contents (Do not change the Section Names!)

<b>1</b>	<b>Introduction .....</b>	<b>3</b>
1.1	Purpose .....	3
1.2	Scope .....	3
1.3	Definitions, acronyms, and abbreviations .....	3
1.4	References .....	3
1.5	Overview .....	3
<b>2</b>	<b>Design and Implementation Constraints .....</b>	<b>3</b>
<b>3</b>	<b>Specific Requirements .....</b>	<b>4</b>
3.1	Functional Requirements .....	4
3.2	Performance Requirements .....	4
3.3	Software System Attributes .....	4
3.4	Use Case Analysis .....	5
3.4.1	Actors .....	5
3.4.2	Scenarios .....	5
3.4.3	Use Case Forms .....	5
3.4.4	Relationships among Actors and Use Cases .....	8
3.4.5	Use Case Diagram .....	8
<b>4</b>	<b>Behavioral Models .....</b>	<b>9</b>
4.1	Sequence Diagram .....	9
<b>5</b>	<b>Structural Models .....</b>	<b>12</b>
5.1	Class Diagram .....	12
<b>6</b>	<b>Process Modeling .....</b>	<b>13</b>
6.1	Data Flow Diagram (DFD) .....	13
<b>7</b>	<b>Graphical User Interface(s) (GUIs) .....</b>	<b>15</b>
<b>8</b>	<b>Conclusion and Future Work .....</b>	<b>25</b>

# 1 Introduction

No explanation is needed here. Only complete the subsections.

## 1.1 Purpose

The purpose of this SRS document is to provide a detailed description of the software system being developed. It includes the functional and non-functional requirements and other specifications necessary for implementing the system successfully.

## 1.2 Scope

- a) The software product to be produced is named “Favorite Sites”.
- b) The system will allow users to manage visits, collect visited locations and share visits with other users.
- c) The goal of the application is to create a user-friendly visit management and sharing system.

## 1.3 Definitions, acronyms, and abbreviations

GUI: Graphical User Interface

SRS: Software Requirements Specifications

DFD: Data Flow Diagram

UML: Unified Modeling Language

## 1.4 References

IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications

Course materials from SE 2224 - Software System Analysis

ChatGPT

## 1.5 Overview

The rest of the SRS document includes detailed descriptions of the specific requirements, design and implementation constraints, use case analysis, behavioral and structural models (sequence and class diagram), process modeling (DFD) and the graphical user interfaces.

# 2 Design and Implementation Constraints

Development Tools: IntelliJ IDEA (version 2022.3.2), MySQL (version 8.0 CE)

Programming Language: Java

Modeling Tool: Visual Paradigm (version 17.1)

Hardware Limitations: Specific memory and timing requirements should be met as detailed in the performance requirements section.

### **3 Specific Requirements**

No explanation is needed here. Only complete the subsections.

#### **3.1 Functional Requirements**

The system shall allow users to log in with their valid username and password.

The system shall allow users to add visit with filling the necessary gaps.

The system shall enable users to delete a visit with the given visit id.

The system shall allow users to get visit details with the given visit id.

The system shall allow users to edit and update the visit details with the given visit id.

The system shall allow users to get the country whose best feature is food.

The system shall allow users to get visits by the given year.

The system shall allow users to get the most visited country.

The system shall allow users to get the visits which visited in spring.

The system shall allow users to display the image of the visit with the given visit id.

The system shall allow users to share visit information with other users.

The system shall allow users to display the shared visits with them.

#### **3.2 Performance Requirements**

The system shall support up to 100 simultaneous users.

The system shall respond to user actions within 2 seconds.

The system shall handle up to 10000 visits without performance degradation.

#### **3.3 Software System Attributes**

Reliability: The system must perform without failure in 95 percent of use cases during a month.

Availability: The system shall not be unavailable more than 1 hour per 1000 hours of operation.

Security: At least 99% of intrusions shall be detected within 10 seconds.

Maintainability: The mean time to restore the system (MTTRS) following a system failure must not be greater than 10 minutes.

Usability: The system shall be user-friendly and easy to navigate.

## 3.4 Use Case Analysis

No explanation is needed here. Only complete the subsections.

### 3.4.1 Actors

User: A person who interacts with the system.

### 3.4.2 Scenarios

**Add a Visit:** A user logs in with his/her valid username and password and adds a visit with details such as country, city, year, season, best feature, comment and rating.

**Share Favorite Visit:** A user logs in with his/her valid username and password and selects share favorite visit, enters the visit id that he/she wants and enters the username that the user wants to share the visit.

### 3.4.3 Use Case Forms

#### Complete Use Case Forms

**1. Use Case Name: Add Visit**

Actors: User

Description: User adds a new visit.

Trigger: User selects 'Add Visit' button.

Preconditions: User must be logged in.

Normal Course: 1) User clicks 'Add visit'.

2) User fills the country, city, year, season, best feature, comment and rating sections.

3) User clicks 'OK' button to save.

Post conditions: Visit is saved in the system.

Exceptions:

E1: Required fields are empty (occurs at step 2)

1) The system displays a message that says all the fields must be filled.

**2. Use Case Name: Share Favorite Visit**

Actor: User

Description: User shares a visit with a user.

Trigger: User selects 'Share Favorite Visit' button.

Preconditions: User must be logged in.

Normal Course: 1) User clicks 'Share Favorite Visit' button.

2) User enters the visit id that he/she wants to share.

3) User clicks 'OK' button

4) User enters the username whose he/she wants to share with.

5) User clicks 'OK' button.

Post conditions: Visit shared with the given username.

Exceptions:

E1: User enters an invalid visit id (occurs at step 2)

1)The system displays a message that says the visit id is invalid.

E2: Users enters an invalid username (occurs at step 4)

1)The system displays a message that says the username is not valid.

### 3. Use Case Name: Get Visits by Year

Actor: User

Trigger: User selects 'Visits by Year' button.

Precondition:

Normal Course: 1) User selects 'Visit by Year' button.

2) User enters the year that she/he wants.

3)User clicks 'OK' button.

Post conditions: Visit details displayed to the screen.

Exceptions:

E1: User enters an invalid year (occurs at step 2)

1) The system displays an empty page.

## Basic Use Case Forms

### 1. Use Case Name: Delete Visit

Actor: User

Description: User deletes an existing visit.

Trigger: User selects 'Delete Visit' button.

Preconditions: User must be logged in.

There must be an existing visit.

### 2. Use Case Name: Log in

Actor: User

Description: User opens the application and logs in.

Trigger: User wants to use the application.

Preconditions: User must have a valid account.

### 3. Use Case Name: Get Visit Details

Actor: User

Description: User displays the visit details.

Trigger: User selects 'Get Visit Details' button.

Preconditions: User must be logged in.

### 4. Use Case Name: Edit

Actor: User

Description: User edits the visit's information.

Trigger: User selects 'Edit' button.

Preconditions: User must be logged in

User must have been clicked the 'Get Visit Details' button and get the details.

5. Use Case Name: Update

Actor: User

Description: User updates the edited visit information.

Trigger: User clicks the 'Update' button.

Preconditions: User must be logged in.

User must have been clicked the 'Get Visit Details' button and get the details.

User must have been clicked the 'Edit' button and edited the visit's details.

6. Use Case Name: Best Food Locations

Actor: User

Description: The system displays the locations whose best feature is food.

Trigger: User selects 'Best Food Location' button.

Preconditions: There must be at least one visit that whose best feature is food.

7. Use Case Name: Most Visited Countries

Actor: User

Description: The system displays the names of the most visited countries.

Trigger: User clicks 'Most Visited Countries' button.

Preconditions: There must be at least one visit.

8. Use Case Name: Visited in Spring

Actor: User

Description: The system displays the visits whose season is spring.

Trigger: User selects 'Visited in Spring' button.

Preconditions:

9. Use Case Name: Display Shared Visit

Actor: User

Description: The system displays the visit ids which has been shared with the user.

Trigger: User selects 'Display Shared Visits' button.

Preconditions: User must be logged in.

10. Use Case Name: Display Image

Actor: User

Description: The system displays the image with the given visit id.

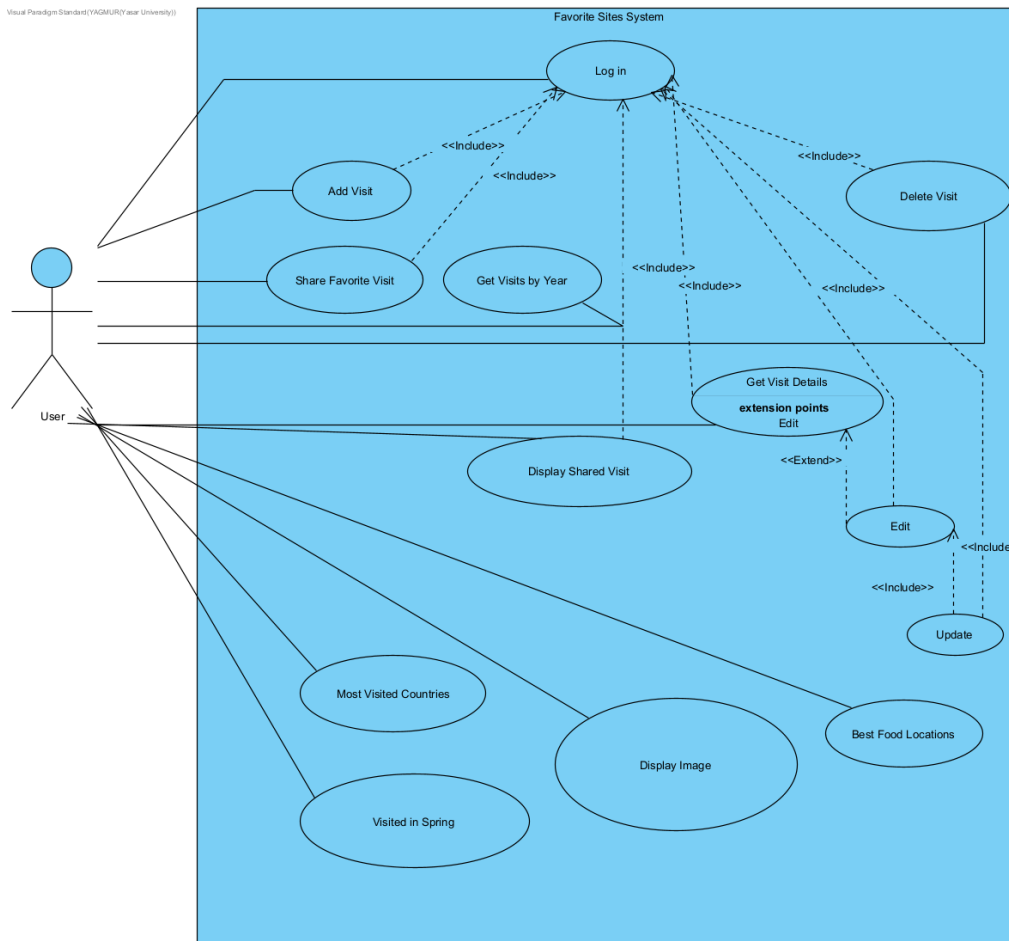
Trigger: User clicks 'Display Image' button.

Precondition:

### 3.4.4 Relationships among Actors and Use Cases

User: Interacts with use cases like Add Visit, Delete Visit, and Share Favorite Visit, Log in, Get Visits by Year, Get Visit Details, Edit, Update, Display Shared Visit, Most Visited Countries, Best Food Locations, Display Image, Visited in Spring.

### 3.4.5 Use Case Diagram



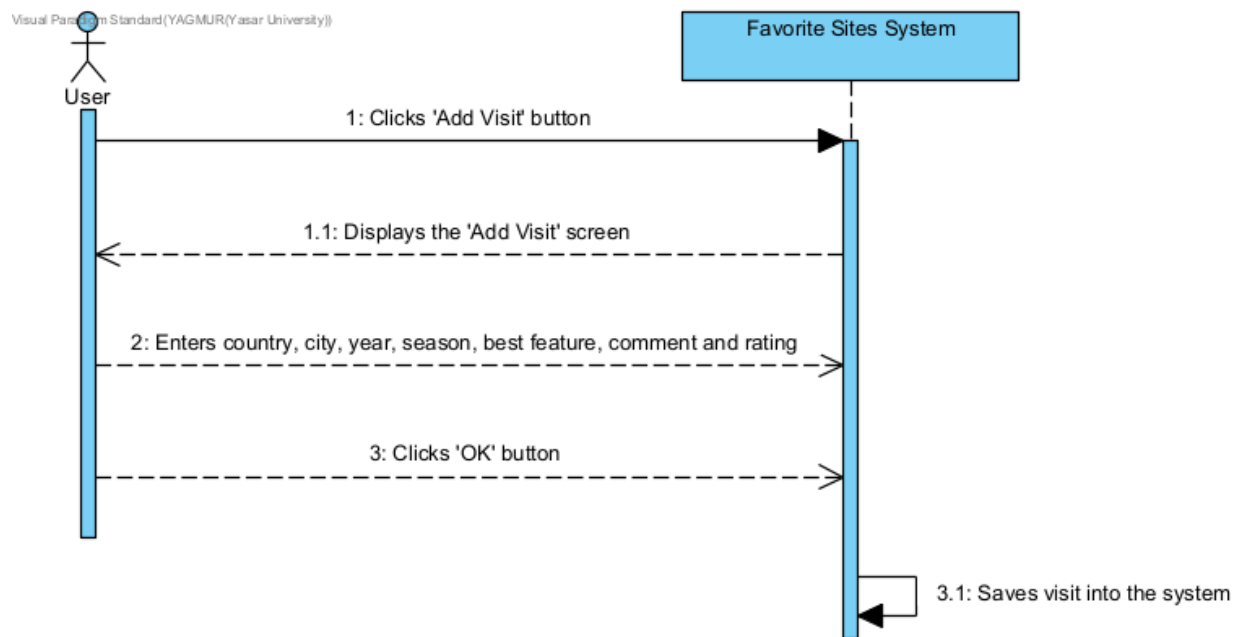
This use case diagram shows us what user can do on the 'Favorite Sites' system. User has to log in to the system for using the add visit, share favorite visit, display shared visit, get visit details, edit, update and delete visit functionalities. When user uses Get Visit Details functionality user also can use edit and update functionalities but, that is not mandatory.



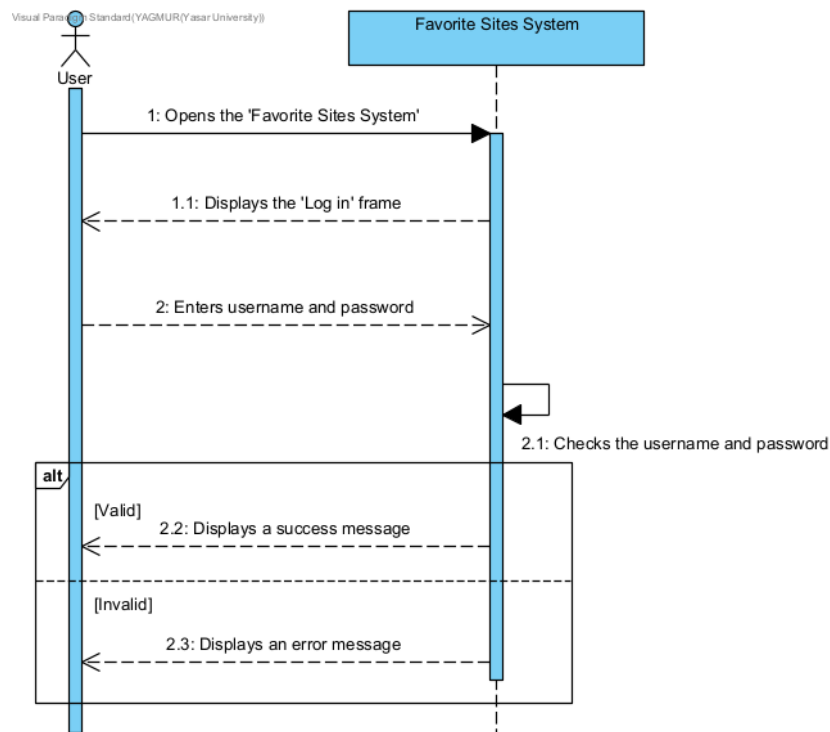
## 4 Behavioral Models

No explanation is needed here. Only complete the subsection.

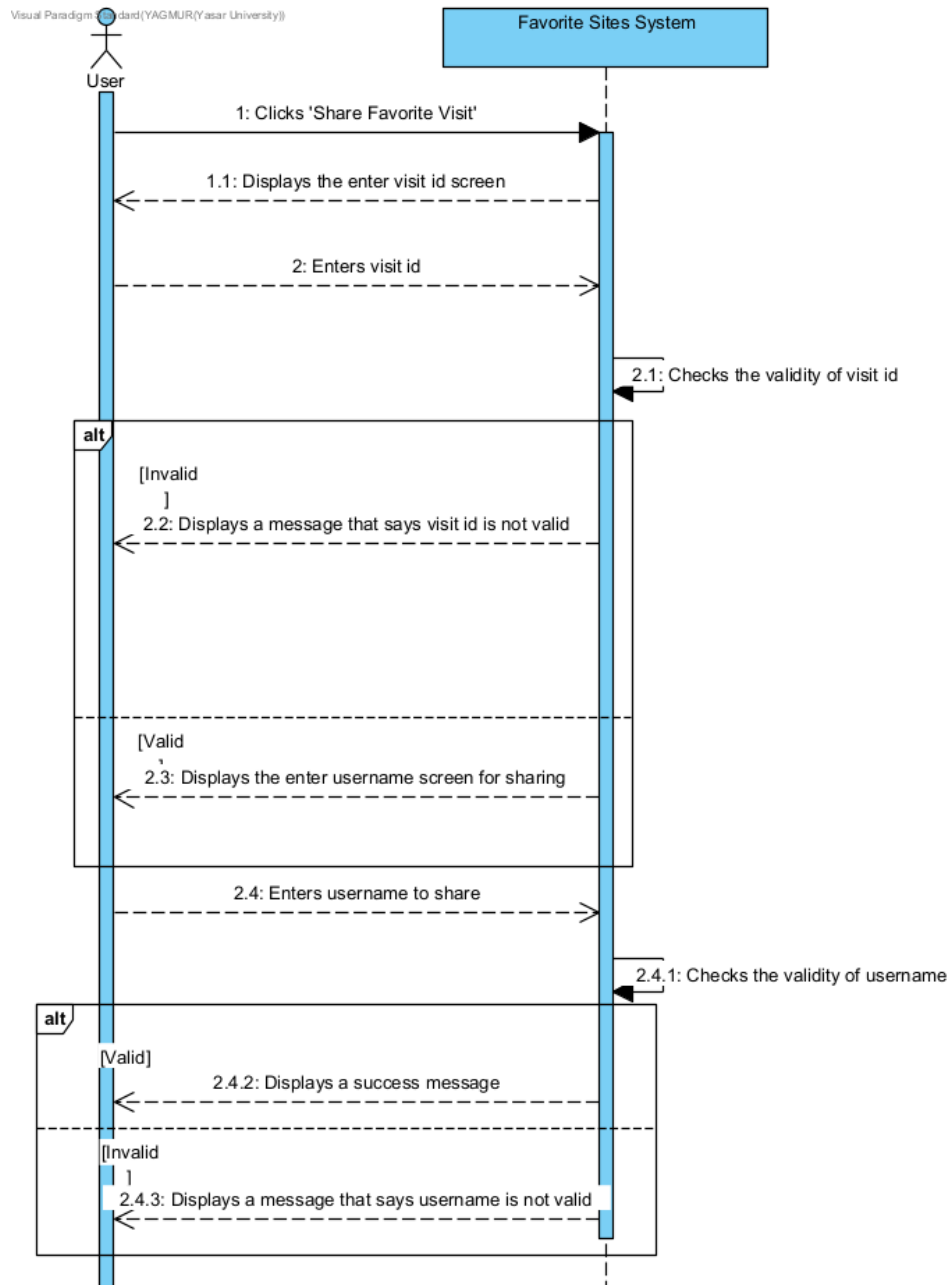
### 4.1 Sequence Diagram



This sequence diagram shows the 'Add Visit' functionality. After logging in, if a user clicks the 'Add Visit' button, the system will display the add visit frame and user fills the necessary gaps. After clicking the 'OK' button, visit will be saved in the system.



This sequence diagram is for the 'Log in' functionality. User opens the 'Favorite Sites' system and the system displays the 'Log in' frame to the user. User enters his/her username and password. If the username and password are valid user can be log in to the system and the system will display a success message. Otherwise, the system will be display an error message to the user.

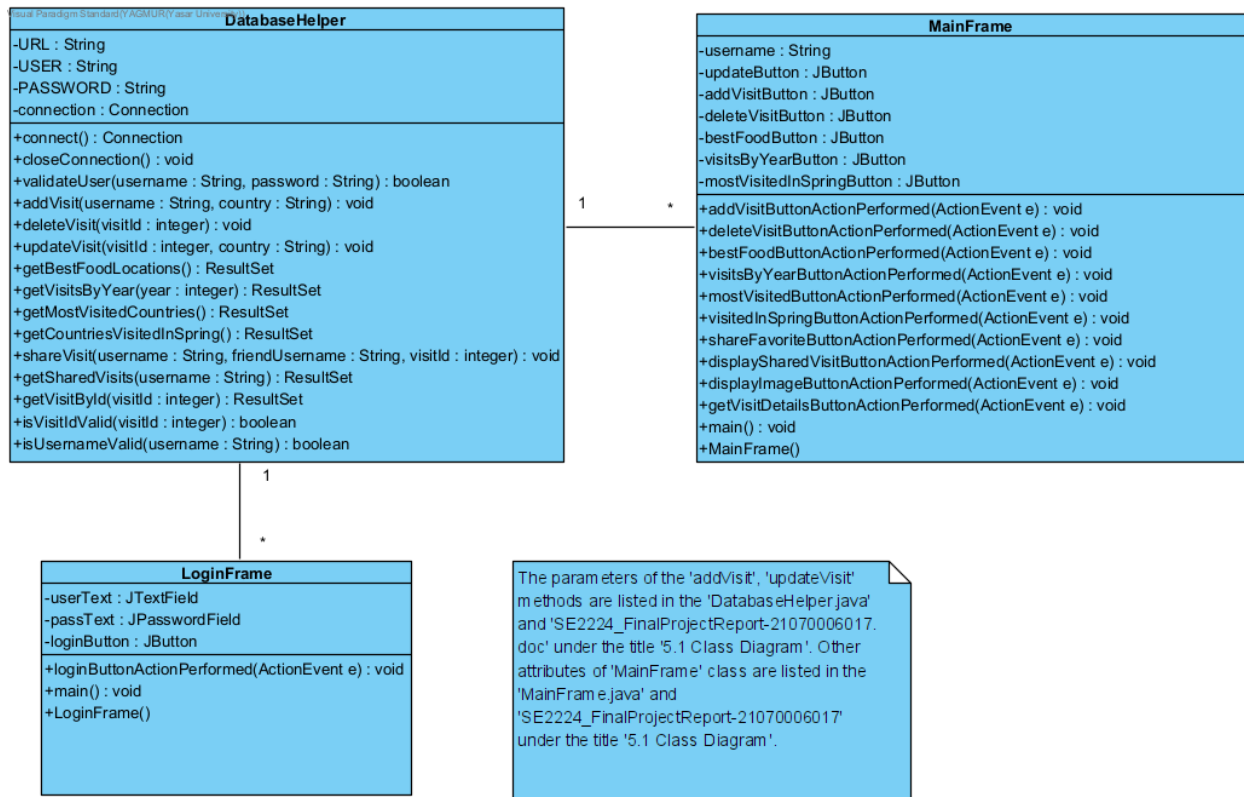


This sequence diagram is for 'Share Favorite Visit' functionality. User opens the 'Favorite Sites' system and logs in. After logging in to the system clicks 'Share Favorite Visit' button. The system displays a screen for entering the visit id. User enters the visit id and system checks its validity. If the visit id is not valid, the system shows a message that says visit id is not valid. If the visit id is valid the system displays another screen for entering the username and after the user enters the username the system checks its validity. If the username is valid the system displays a success message. Otherwise, the system displays a message that says username is not valid.

## 5 Structural Models

No explanation is needed here. Only complete the subsection.

### 5.1 Class Diagram



This is a class diagram for 'Favorite Sites' system. The parameters of the 'addVisit' and 'updateVisit' methods are:

- 1) addVisit (username: String, country: String, city: String, year: integer, season: String, feature: String, comment: String, rating: integer): void
- 2) updateVisit (visitId: integer, country: String, city: String, year: integer, season: String, feature: String, comment: String, rating: integer)

Also, you can find these parameters in 'DatabaseHelper.java'.

The attributes of 'Main Frame': username: String, updateButton: JButton, addVisitButton: JButton, deleteVisitButton: JButton, bestFoodButton: JButton, visitsByYearButton: JButton, mostVisitedButton: JButton, visitedInSpringButton: JButton, shareFavoriteButton: JButton, displaySharedVisitButton: JButton, displayImageButton: JButton, getVisitDetailsButton: JButton, editButton: JButton.

Also, you can find these attributes in 'MainFrame.java'.

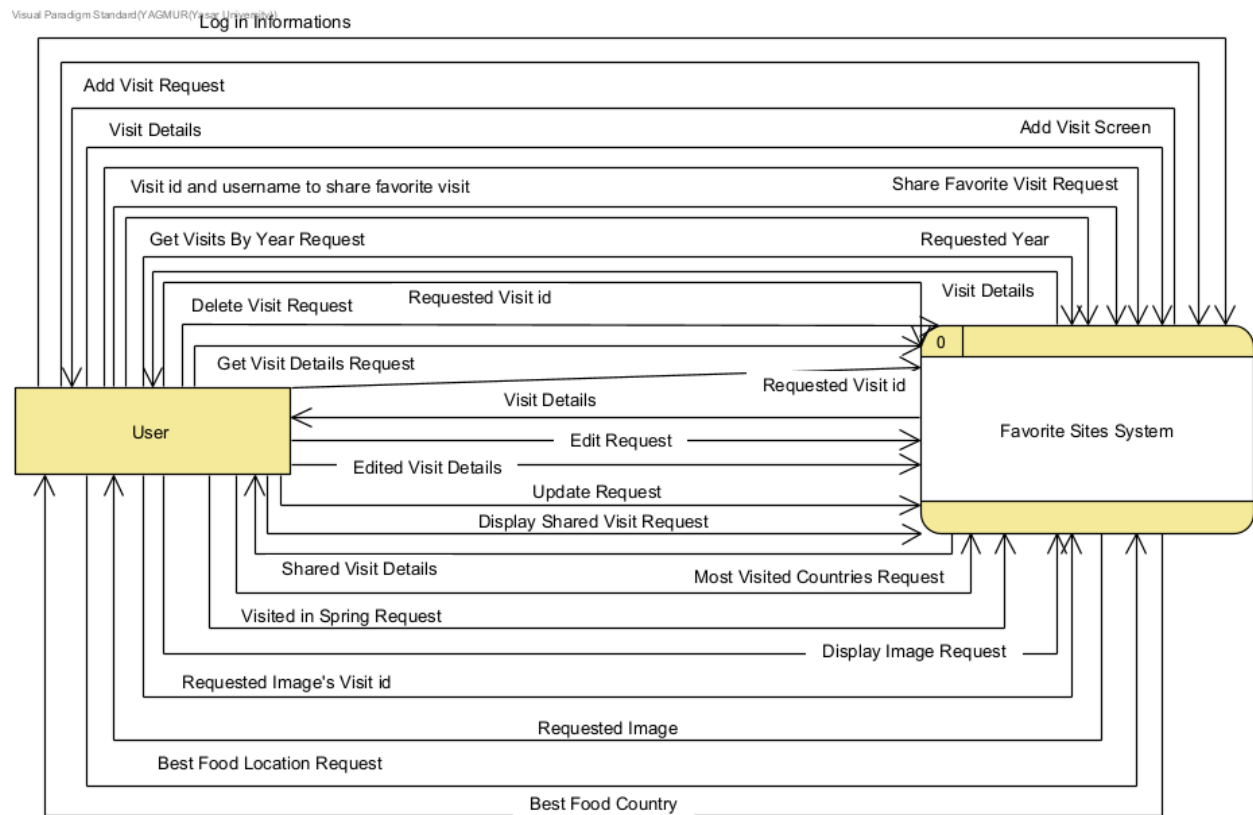
There is an association relationship between LoginFrame and DatabaseHelper. LoginFrame uses the methods of the DatabaseHelper class to perform user authentication. During user login, the LoginFrame class uses the validateUser method of the DatabaseHelper class to check the user's credentials.

There is an association relationship between MainFrame and DatabaseHelper. The MainFrame class uses the methods of the DatabaseHelper class for database operations. However, the MainFrame class can perform many different database operations, so this relationship represents a one-to-many relationship for various interconnected database operations.

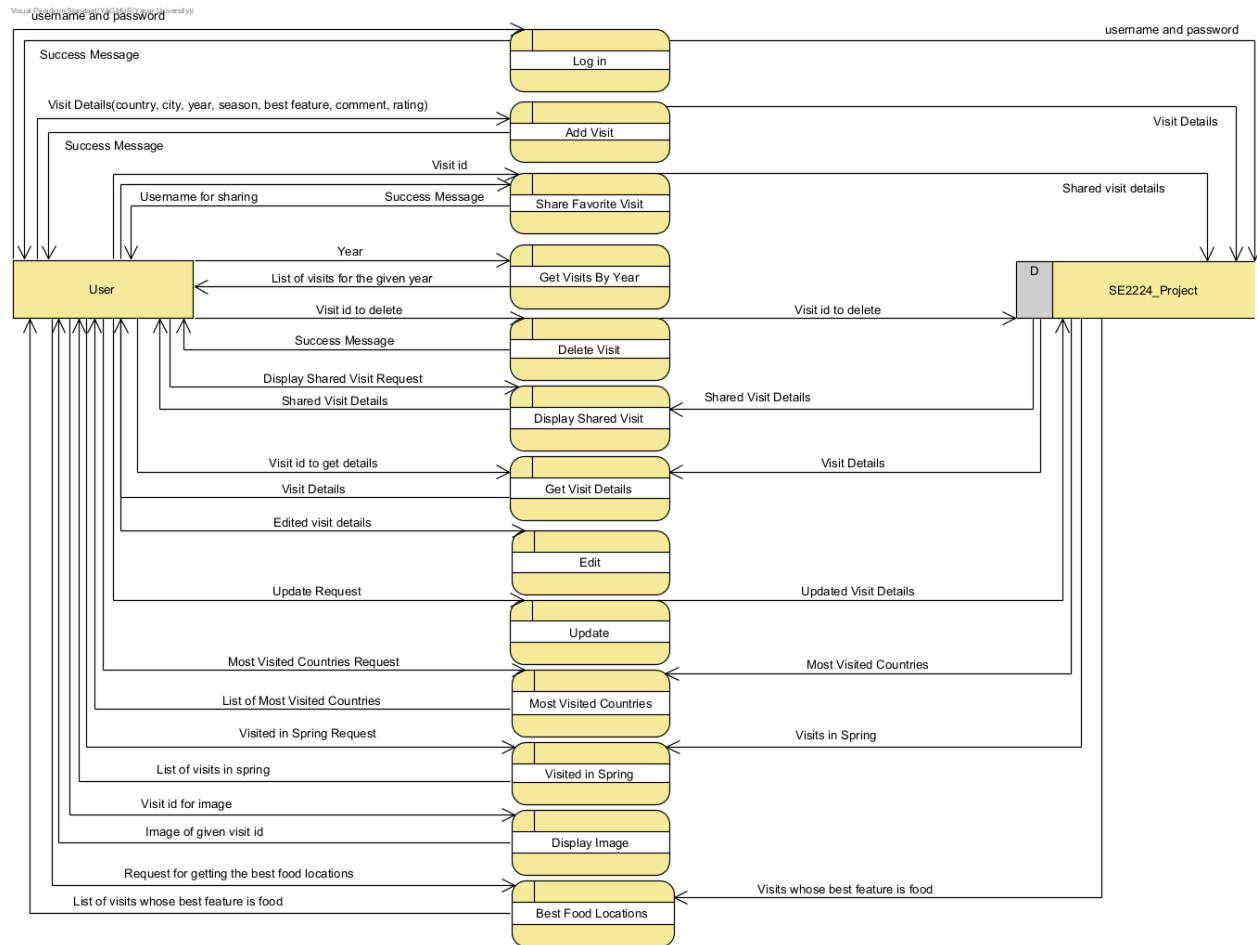
## 6 Process Modeling

No explanation is needed here. Only complete the subsection.

### 6.1 Data Flow Diagram (DFD)



This is the context diagram of the 'Favorite Sites' system. Context diagram contains the 'User' as an external entity and 'Favorite Sites' system as a process. The diagram shows the requests can be made by a user and the system's responses.

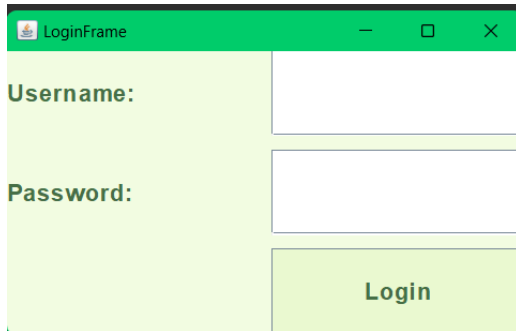


This is the level-0 diagram of the 'Favorite Sites' system. This data flow diagram shows all the use cases in the system. This diagram contains 'User' as external entity and 'SE2224\_Project' as a datastore. The diagram shows all the uses of processes.

## 7 Graphical User Interface(s) (GUIs)

### LoginFrame

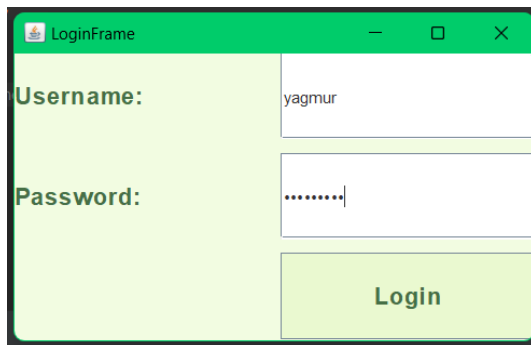
#### Login Frame



A screenshot of a Java Swing window titled 'LoginFrame'. The window has a light green background. On the left, there are two labels: 'Username:' and 'Password:'. To the right of 'Username:' is an empty text input field. To the right of 'Password:' is an empty password input field (masked with dots). Below the password field is a green button with the text 'Login'.

This is the 'LoginFrame' for users to log in.

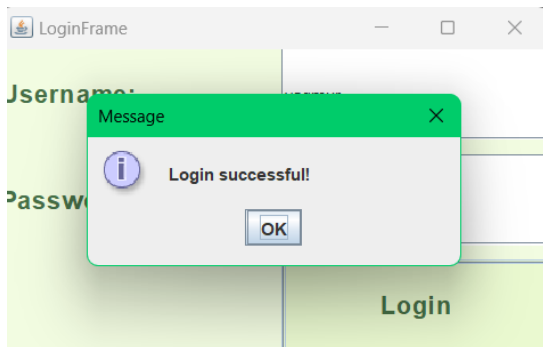
#### Login Frame with Username and Password



A screenshot of the 'LoginFrame' window. The 'Username:' field now contains the text 'yagmur'. The 'Password:' field contains seven dots, indicating a masked password. The 'Login' button remains visible at the bottom.

The user types his/her username and password.

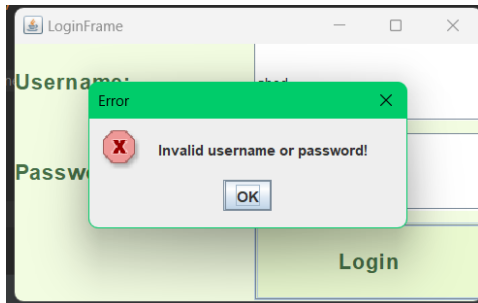
#### Login Frame Success Message



A screenshot of the 'LoginFrame' window with a modal dialog box on top. The dialog box is titled 'Message' and has a green header bar. It contains an information icon (i) and the text 'Login successful!'. There is an 'OK' button at the bottom of the dialog. The background window is partially obscured by the dialog.

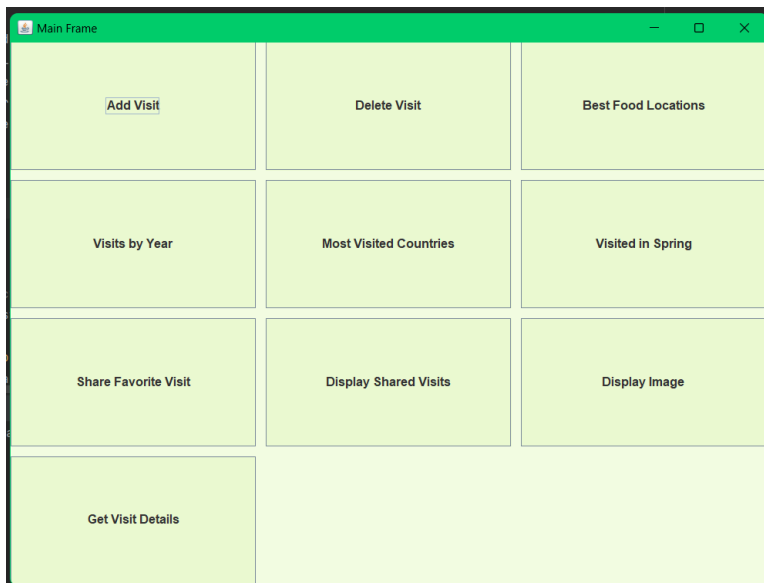
If the user types a valid username and password. The system displays a success message.

## Login Frame Error Message



The system displays an error message if the given username or password is incorrect.

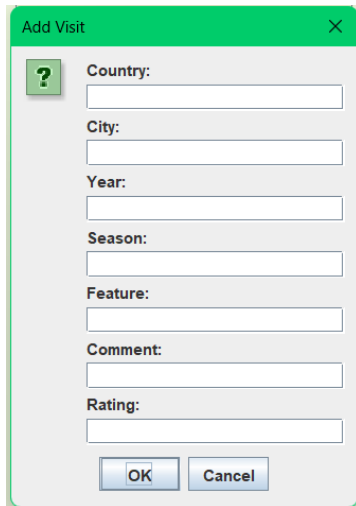
## MainFrame



This is the 'MainFrame' that user sees after logging in.

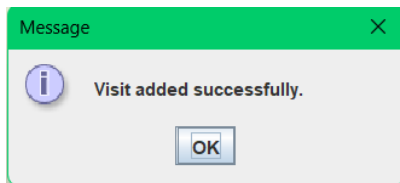


### Add Visit Functionality

A dialog box titled "Add Visit" with a green header bar and a close button (X) in the top right corner. On the left side, there is a green square icon containing a white question mark. The main area of the dialog box contains several input fields, each with a label to its left: "Country:", "City:", "Year:", "Season:", "Feature:", "Comment:", and "Rating:". Each label is followed by a white rectangular text input box. At the bottom of the dialog box, there are two buttons: "OK" and "Cancel".

User enters the details of the visit.

### Add Visit Functionality Success Message



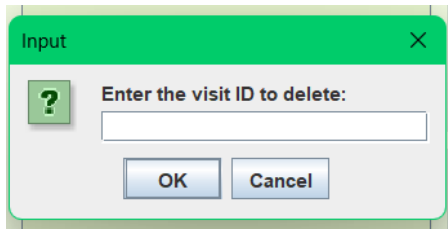
If all the necessary gaps filled by the user. The visit saved to the database and a success message displayed by the system.

### Add Visit Functionality Error Message



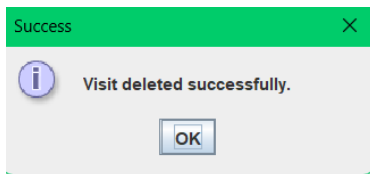
If user does not fill all the fields the system displays an error message.

### Delete Visit Functionality



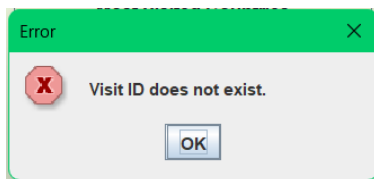
User can delete a visit by using 'Delete Visit' functionality.

### Delete Visit Functionality Success Message



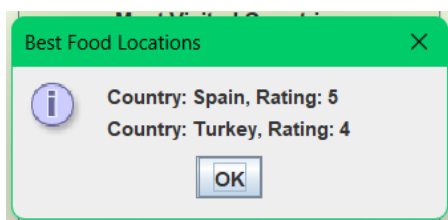
If user enters a valid visit id, the system deletes the visit and displays a success message.

### Delete Visit Functionality Error Message



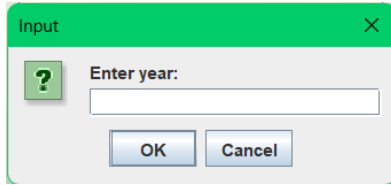
If user enters an invalid visit id, the system displays an error message.

### Best Food Locations Functionality



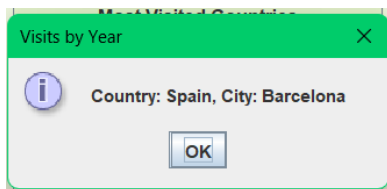
This function displays the countries whose best feature is food.

### Visits By Year Functionality



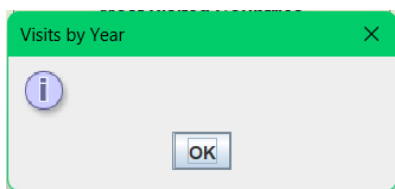
User enters the desired year for seeing the visits that in this year.

### Visits By Year Functionality with Valid Year



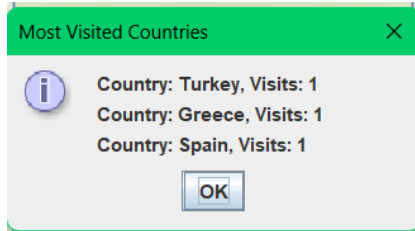
If the given year is valid, the system shows the visits that is in this year.

### Visits By Year Functionality with Invalid Year



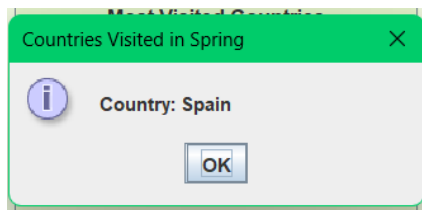
If the given year is invalid, the system displays an empty screen.

### Most Visited Countries Functionality



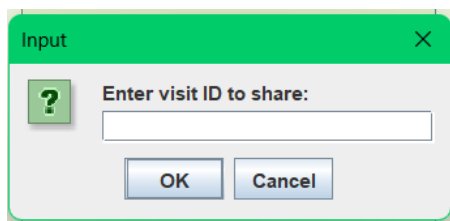
This function shows the most visited countries.

### Countries Visited in Spring Functionality



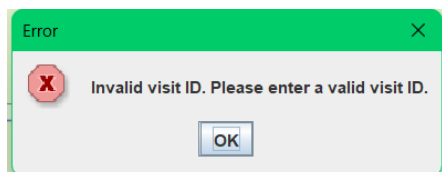
This functions shows the countries that is visited in spring.

### Share Favorite Visit Functionality



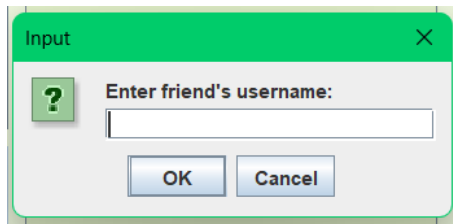
User can share his/her favorite visit with other users by using 'Share Favorite Visit' functionality.

### Share Favorite Visit Functionality Error Message Invalid Visit ID



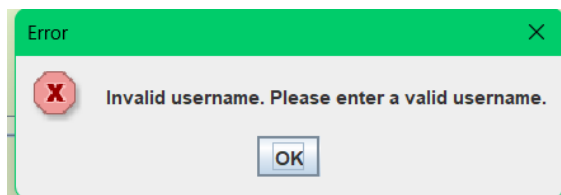
If user enters an invalid visit id to share the system displays an error message.

### Share Favorite Visits Functionality Enter Friend Username Field



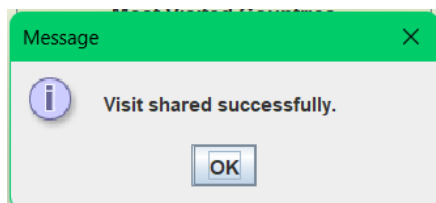
This screen displays after entering a valid visit id, user can enter the friend's username that she/he wants to share with.

### Share Favorite Visits Functionality Error Message Invalid Username



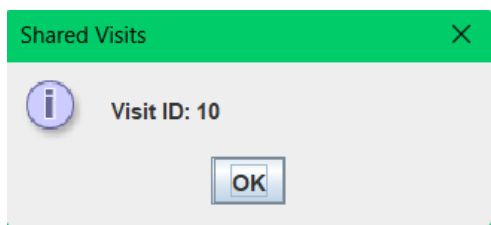
If users enters an invalid username to share, the system displays an error message.

### Share Favorite Visits Functionality Success Message



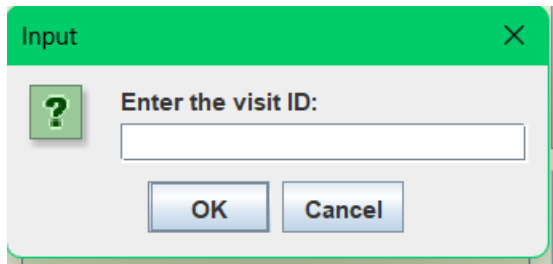
The system displays this message after entering a valid visit id and a valid username to share.

### Display Shared Visits Functionality



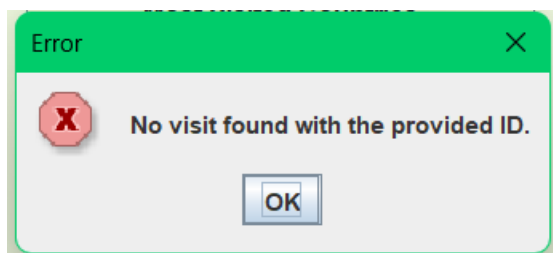
This functionality shows the shared visit ids with the user.

### Display Image Functionality



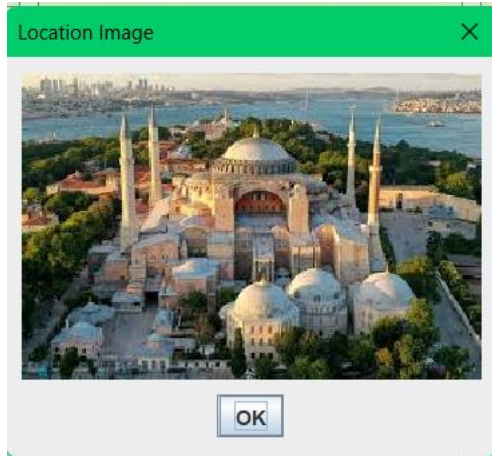
User can enter the visit id for displaying the image of the given visit id.

### Display Image Functionality Error Message



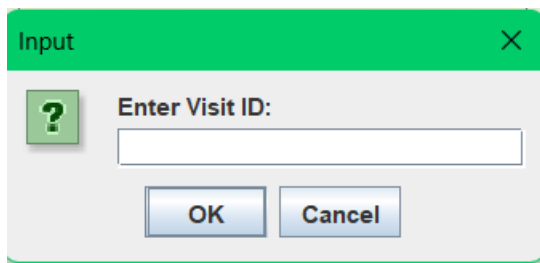
The system shows an error message if the given visit id is invalid.

### Display Image Functionality Succes



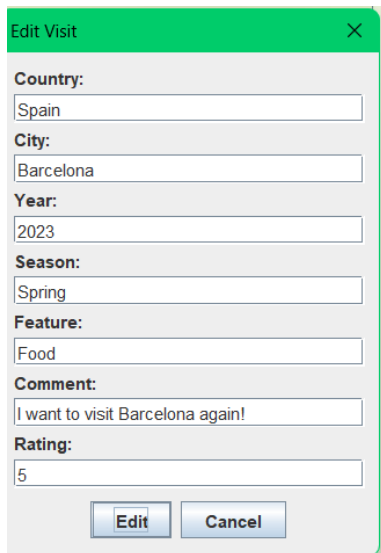
The system displays the image if the given visit id is valid.

### Get Visit Details Functionality



User can get the visit details by entering the visit id.

## Visit Details

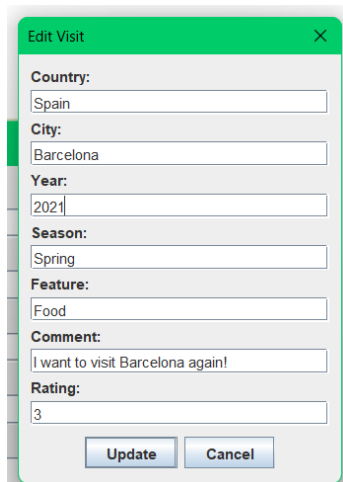


The image shows a dialog box titled "Edit Visit" with a green header bar and a close button (X) in the top right corner. The dialog contains several text input fields, each preceded by a label: "Country:" with the value "Spain", "City:" with "Barcelona", "Year:" with "2023", "Season:" with "Spring", "Feature:" with "Food", "Comment:" with "I want to visit Barcelona again!", and "Rating:" with "5". At the bottom of the dialog are two buttons: "Edit" and "Cancel".

Field	Value
Country:	Spain
City:	Barcelona
Year:	2023
Season:	Spring
Feature:	Food
Comment:	I want to visit Barcelona again!
Rating:	5

The system displays visit details after entering a valid visit id.

## Edit Visit



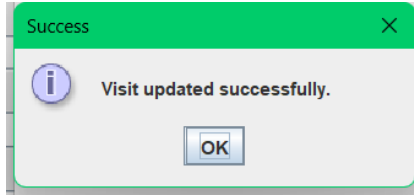
The image shows a dialog box titled "Edit Visit" with a green header bar and a close button (X) in the top right corner. The dialog contains several text input fields, each preceded by a label: "Country:" with the value "Spain", "City:" with "Barcelona", "Year:" with "2021", "Season:" with "Spring", "Feature:" with "Food", "Comment:" with "I want to visit Barcelona again!", and "Rating:" with "3". At the bottom of the dialog are two buttons: "Update" and "Cancel".

Field	Value
Country:	Spain
City:	Barcelona
Year:	2021
Season:	Spring
Feature:	Food
Comment:	I want to visit Barcelona again!
Rating:	3

User can edit visit details in this screen.



### Success Message



The system displays this success message after editing with valid inputs and clicking the update button.

## 8 Conclusion and Future Work

The 'Favorite Sites' system project has been developed for simplifying the management and sharing of visit details. The system allows users to add a new visit, delete visit and share visit information with other users, etc. . In future work, uploading an image to the system, integration with calendar applications, sending notifications to the user and mobile app development can be considered for the system.