Vietnam National University, Ha Noi

University of Engineering and Technology

Object Oriented Analysis and Design (INT3110E 55)

Course Project

Parking lots Management System

Instructor: Assoc. Prof. Dr. Trương Ninh Thuận

Team members: Đỗ Việt Hưng
Nguyễn Đức Thành

HA NOI - 2024

Vietnam National University, Ha Noi

University of Engineering and Technology

Object Oriented Analysis and Design (INT3110E 55)

Course Project

Parking lots Management System

Instructor: Assoc. Prof. Dr. Trương Ninh Thuận

Team members: Đỗ Việt Hưng
Nguyễn Đức Thành

HA NOI - 2024

Table of contents

1. Requirements

1.1. Problem Statement

In a constantly advancing and growing city, there is an increased demand for space in every aspect of life. This ranges from residence houses, space for businesses and services, roads,... One of them is the increasing need for a parking lot for cars, which is of paramount importance in every city around the world. With the technological advancement and the trend of digitalization, I believe using IT to manage car parking lots will have great benefits and assist everyone in the city in their everyday lives. This system will help owners of parking areas have their business known to the people in the area, and help residents find a place for their vehicles.

Owners of parking lots can register, manage and moderate their business easily in an application and employees can work with it effortlessly with a few pushes of a button. The parking place on the system will have the total number of parking slots, slots available, details on the people using the parking space, and all the information everyone needs.

Users can look for car parks in the surrounding area, seeing the details such as address, availability status, whether it's an outdoor parking slot or an indoor one, prices.. After choosing a location, users can choose whether they want to park for the day, overnight or register a long term parking plan. Once the process has been completed, users will receive a ticket code and give it to the guard in order to check into the parking lot. The app will make sure the reservation is available until the user has checked into the parking lot.

Users who have parked at the car park can leave a review and contact the owner of the place to resolve any complaints and problems related. Owners can also use the contact system to help and guide the users on location or on the procedures of the parking place.

The only downside is that employees working at the parking lot will have to constantly update the status. The plan is to have an integrated system where the status will be updated automatically when they check cars in on their own system.

1.2. Glossary

1.2.1. Introduction

This document is used to define terminology specific to the problem domain, explaining terms, which may be unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal data dictionary, capturing data definitions so that use-case descriptions and other project documents can focus on what the system must do with the information.

1.2.2. Definitions

The glossary contains the working definitions for the key concepts in the Parking Lots Management System.

User
A car driver who uses the application to searches for or book car parks
Car Parking Lot Owner
A person who owns one or many parking lots
Administrator
A person whose job is to ensure that the site is free of spam and inappropriate behavior.
Booking
Information about a reservation made by a user to book a parking slot.
Review
A rating sent from a registered user for a parking lot.

1.3. Supplementary Specification

1.3.1. Objectives

The purpose of this document is to define requirements of the Parking lots Management System. This Supplementary Specification lists the requirements that are not readily captured in the use cases of the use-case model. The Supplementary Specifications and the use-case model together capture a complete set of requirements on the system.

1.3.2. Scope

This Supplementary Specification applies to the Parking lots Management System, which will be developed by the OOAD students.
This specification defines the non-functional requirements of the system; such as reliability, usability, performance, and supportability, as well as functional requirements that are common across a number of use cases. (The functional requirements are defined in the Use Case Specifications.)

1.3.3. References

None.

1.3.4. Functionality

Multiple users must be able to perform their work concurrently.

1.3.5. Usability

The system must be user-friendly, allowing a new user to learn and navigate the system within 1 hour.
The user interface should be intuitive, providing easy access to features such as booking, viewing parking availability, and managing parking lot details.

1.3.6. Reliability

The system must be available 24 hours a day, 7 days a week, with a downtime of no more than 5%.
The system should be able to recover from unexpected shutdowns without data loss.

1.3.7. Performance

The system shall support up to 10,000 simultaneous users.
The system must process 90% of all user transactions, including searches and bookings, within 5 seconds.
The system should be able to handle heavy traffic during peak times without significant performance degradation.

1.3.8. Supportability

None.

1.3.9. Security

Users must authenticate with a valid username and password to access their accounts.
Only parking lot owners and administrators can modify parking lot information.
User data, including reservations and payment details, must be protected against unauthorized access.

1.3.10. Design Constraints

The system must provide a responsive web-based interface usable on computers and mobile devices.

1.4. Use-Case Model

1.4.1. Parking lots Management System Use-Case Model Main Diagram

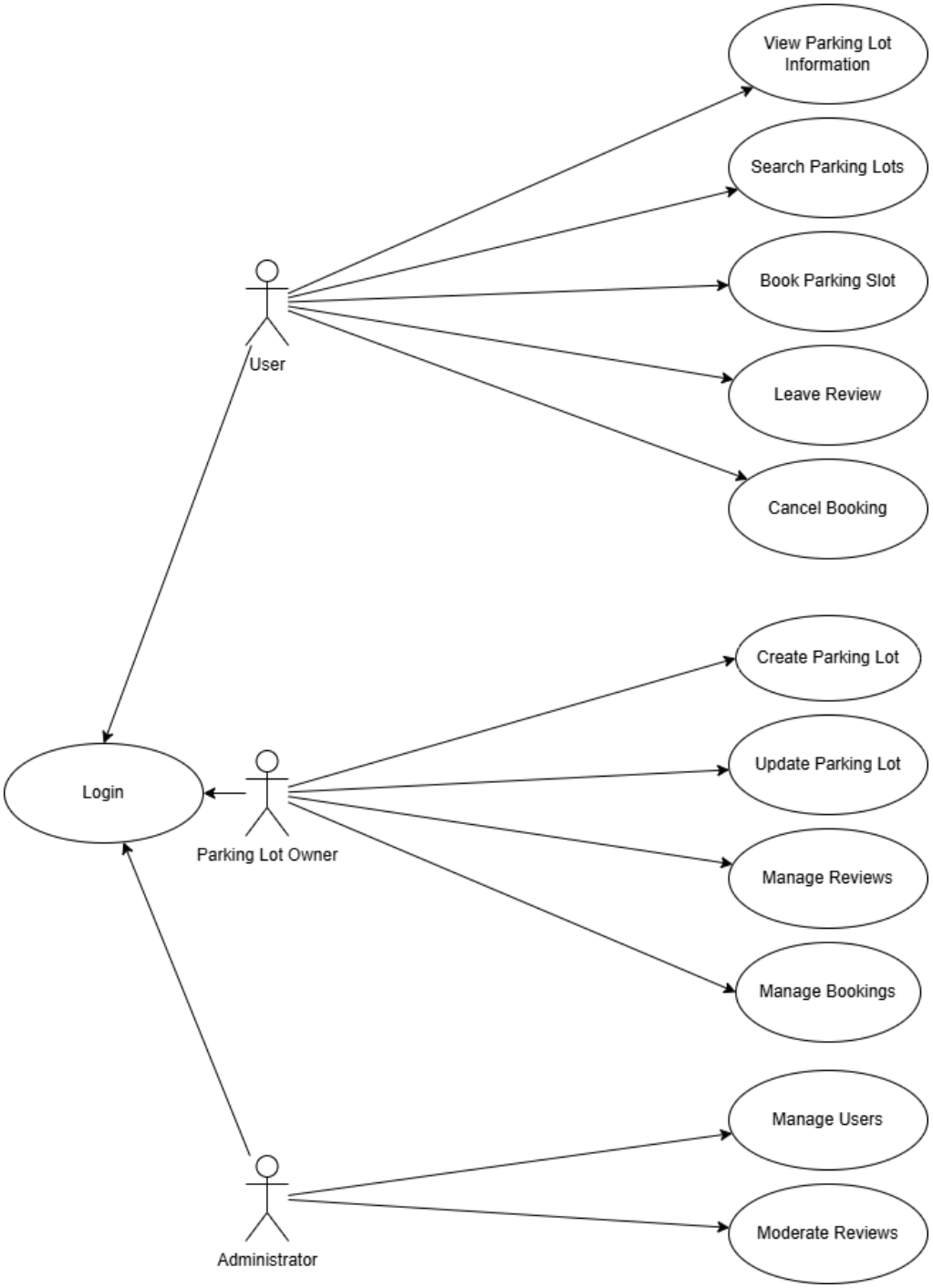

Figure 1-1. Use-case model for the Parking lots Management System

1.4.2. View Parking Lot Information

Brief Description
This use case allows a user to view the information of a parking lot.

Flow of Events
Basic Flow

- This use case starts when a user selects a parking lot from the list.
- The system displays information about the selected parking lot, including the address, contact information, prices, total parking slots, available parking slots, reviews, and other relevant details.

Special Requirements
None.

Pre-conditions

None.

Post-conditions

The system state remains unchanged after displaying the information.

Extension Points
None.

1.4.3. Search Parking Lots

Brief Description
This use case allows the user to search for available parking lots.

Flow of Events
Basic Flow

- The user initiates a search by providing location details (e.g., current location or desired area).
- The system displays a list of parking lots in the specified area, including availability status, pricing, and types of slots (indoor/outdoor).

Special Requirements

Location services might be required for accurate search results.

Pre-conditions

None.

Post-conditions

The system may store the user's recent search for future reference.

Extension Points
None.

1.4.4. Book Parking Slot

Brief Description
This use case allows the user to reserve a parking slot in the system. The user is on a parking lot page on the website.

Flow of Events
Basic Flow

- The user selects a parking lot and specifies the desired parking duration (e.g., short-term or long-term).
- The system checks the availability of parking slots in the selected lot.
- If available, the user confirms the booking.
- The system generates a booking confirmation, including a unique code for parking.

Alternative Flow

No Slots Available

- If no parking slots are available:
    - The system informs the user that no slots are available in the selected lot.
    - The user may choose to search for another parking lot.

Special Requirements
None.

Pre-conditions

- The user must be logged in.
- The user is on a parking lot page on the website.
- There must be available slots in the parking lot.

Post-conditions

- The system updates the availability of parking slots in the selected lot.
- The user receives a unique booking code for parking.

Extension Points
None.

1.4.5. Leave Review

Brief Description
This use case allows the user to leave a review for a parking lot after using it.

Flow of Events
Basic Flow

- The user navigates to the parking lot's page and selects the "Leave Review" option.
- The user inputs their feedback and rating.
- The system stores the review and displays it publicly.

Special Requirements
None.

Pre-conditions

The user must have previously booked and used a parking slot.

Post-conditions

The system stores the user's review and updates the parking lot's review section.

Extension Points
None.

1.4.6. Cancel Booking

Brief Description
This use case allows a user to cancel a previously made parking reservation.

Flow of Events
Basic Flow

- The user navigates to their booking history and selects the reservation to cancel.
- The system confirms the cancellation and releases the parking slot back to the available pool.
- The system notifies the user that the booking has been successfully canceled.

Special Requirements
None.

Pre-conditions

The user must have an existing booking to cancel.

Post-conditions

The system updates the availability of the parking slots in the parking lot.

Extension Points
None.

1.4.7. Login

Brief Description
This use case allows users, owners and admins to log into the system.

Flow of Events
Basic Flow

- The actor enters their credentials (username and password).
- The system verifies the credentials and grants access to the appropriate interface (User, Owner, Admin).

Alternative Flow

Invalid Information

If, in the **Basic Flow**, the actor enters an invalid ID and/or password, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the login, at which point the use case ends.

Missing Required Information

If any fields in the form are left empty, the system displays an error message. The actor can continue modifying the form or cancel the operation, at which point the use case ends.

Register Account

The actor can choose to register a new account:

- The actor selects the "Register" option.
- The system redirects the actor to the registration interface.
- The actor completes the registration process.
- The system creates a new account and redirects the actor to the login form to proceed with login.

Special Requirements
None.

Pre-conditions

The actor must have valid login credentials (if they intend to log in).

Post-conditions

The actor gains access to the system's features.

Extension Points
None.

1.4.8. Create Parking Lot

Brief Description
This use case allows a parking lot owner to add a new parking lot to the system.

Flow of Events
Basic Flow

- The owner logs into the system and navigates to the "Create Parking Lot" section.
- The owner enters details for the new parking lot, including name, location, contact information, total parking slots, pricing, and type (indoor/outdoor).
- The system validates the information and saves the new parking lot record.
- The system confirms the creation of the new parking lot and displays it in the owner's list of managed lots.

Alternative Flows

Cancel Operation

- The owner cancels the creation process at any point.
- The system discards any entered information and returns to the parking lot management interface.

Missing or Invalid Information

- The owner submits the form, but some required fields are missing or contain invalid data.
- The system notifies the owner of the specific errors.
- The owner corrects the errors and resubmits the form.

Duplicate Parking Lot

- The owner tries to create a parking lot that already exists (e.g., same name and location).
- The system alerts the owner of the duplicate and prevents the creation.

- The owner either cancels the operation or modifies the details to proceed.

Special Requirements
None.

Pre-conditions

The owner must be logged in.

Post-conditions

The new parking lot is available in the system and visible to users.

Extension Points
None.

1.4.9. Update Parking Lot

Brief Description
This use case allows the parking lot owner to update details of the parking lot.

Flow of Events
Basic Flow

- The owner logs in and navigates to the parking lot management interface.
- The system displays the current parking lot details.
- The owner updates information such as available slots, prices, and contact details.
- The system saves the changes and updates the displayed information.

Alternative Flows

Delete Parking Lot

- The owner chooses to delete the parking lot.
- The system prompts for confirmation.
- Upon confirmation, the system deletes the parking lot and removes it from public view.

Cancel Operation

- The owner cancels the operation at any point.
- The system discards any unsaved changes and returns to the main management interface.

Missing Information

- The owner attempts to save changes but leaves required fields empty.

- The system notifies the owner of the missing information.
- The owner must complete the required fields to proceed.

Special Requirements
None.

Pre-conditions

- The owner must be logged in.
- The parking lot must already be registered.

Post-conditions

- For updates, the system reflects the updated details on the parking lot's public page.
- For deletions, the parking lot is removed from the system.

Extension Points
None.

1.4.10. Manage Reviews

Brief Description
This use case allows the parking lot owner to view and manage user-submitted reviews related to their parking lot. The owner can respond to reviews, add comments, and address complaints.

Flow of Events
Basic Flow

- The owner logs into the system and navigates to the "Manage Reviews" section.
- The system displays a list of user reviews for the owner’s parking lot.
- The owner can view each review in detail, including user comments and ratings.
- The owner can respond to a user review by adding a comment or addressing a complaint.
- The owner’s response is saved and displayed with the review.

Special Requirements
None.

Pre-conditions

- The owner must be logged in.
- The parking lot must have existing user reviews.

Post-conditions

The system updates the review section with the owner’s responses.

Extension Points
None.

1.4.11. Manage Bookings

Brief Description
This use case allows the parking lot owner to view and manage all bookings related to their parking lot, including the option to close bookings when the period has ended or the user has left the parking lot.

Flow of Events
Basic Flow

- The owner logs into the system and navigates to the "Manage Bookings" section.
- The system displays a list of current bookings for the owner's parking lot.
- The owner can view booking details, including user information and booking status.

  Close Booking

  The owner selects a booking to close.

  - The system confirms the action and updates the booking status to "Closed" or "Completed."
  - The system releases the parking slot back into availability for new bookings.

Special Requirements
None.

Pre-conditions

- The owner must be logged in.
- The booking must be active and related to the owner's parking lot.

Post-conditions

- The system reflects any changes made to the booking status.
- Closed bookings release the associated parking slots for new bookings.

Extension Points
None.

1.4.12. Manage Users

Brief Description
This use case allows the admin to manage user accounts, including suspending and deleting user accounts.

Flow of Events
Basic Flow

The use case begins when the admin navigates to the user management section.

- The system displays a list of user accounts.
- The admin selects a user account to manage.
- The admin can choose one of the following actions:

  Suspend User

  - The admin selects the **Suspend** option for a user account.
  - The system marks the account as suspended, restricting the user's access.
  - The system updates the user's status in the database and logs the suspension action.

  Delete User

  - The admin selects the **Delete** option for a user account.
  - The system prompts the admin to confirm the deletion.
  - If confirmed, the system permanently removes the user account from the database.
  - The system logs the deletion action and updates the user database.

Alternative Flows

  Cancel Operation

  - At any point, the admin can choose to cancel the user management action.
  - The system discards any pending changes and returns to the list of user accounts.

Special Requirements
None.

Pre-conditions

The admin must be logged in.

Post-conditions

The system reflects any changes made by the admin to user accounts, updating statuses or removing accounts as applicable.

Extension Points
None.

1.4.13. Moderate Reviews

Brief Description
This use case allows the admin to review and moderate user-submitted reviews for inappropriate content.

Flow of Events
Basic Flow

This use case begins when the admin requests to view a list of user reviews.

- The system displays a list of user reviews.
- The admin selects a review to moderate.
- The admin can choose one of the following actions:

  Flag Review

  - The admin selects the **Flag** option.
  - The system marks the review as flagged for inappropriate content.
  - The system updates the review status to "Flagged" and logs the action.

  Delete Review

  - The admin selects the **Delete** option.
  - The system prompts the admin to confirm the deletion.
  - If confirmed, the system removes the review from the database.
  - The system updates the review section and logs the deletion action.

Alternative Flows

Cancel Operation

- At any point, the admin can choose to cancel the moderation action.
- The system discards any pending moderation actions and returns to the list of reviews.

Special Requirements
None.

Pre-conditions

The admin must be logged in.

Post-conditions

The system reflects any moderation actions taken by the admin on the reviews, with updated statuses or removal if applicable.

Extension Points
None.

2. Use-case Analysis

2.1. Architectural Analysis

2.1.1. High-level organization of the model

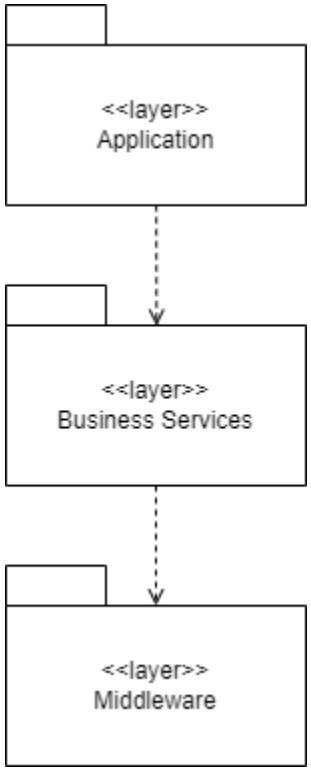

Figure 2-1. Layering approach

The above figure describes the high-level organization of the software system. The system consists of three layers:

- The *Application* layer contains the design elements that are specific to each use case of the system.
- The *Business Services* layer encapsulates some key abstractions and services common to all use cases. It is accessible from the Application layer.
- The *Middleware* layer offers services to enable data communication and management on distributed systems.

2.1.2. Key abstractions

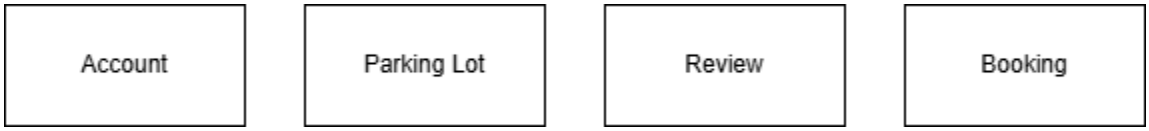

Figure 2-2. Key abstractions used in the application

- *Account:* A record about a user/parking lot owner/administrator. Each account has a unique user ID and a password, which is used to identify the actor and grant them access to secure parts of the system.
- *Parking Lot:* An announcement posted by a registered parking lot owner about the information of a parking lot offered by that actor.
- *Review:* A rating sent from a registered user for a parking lot.
- *Booking:* Information about a reservation made by a user to book a parking slot.

2.2. Use-case realizations

2.2.1. Use-case realizations: sequence diagrams

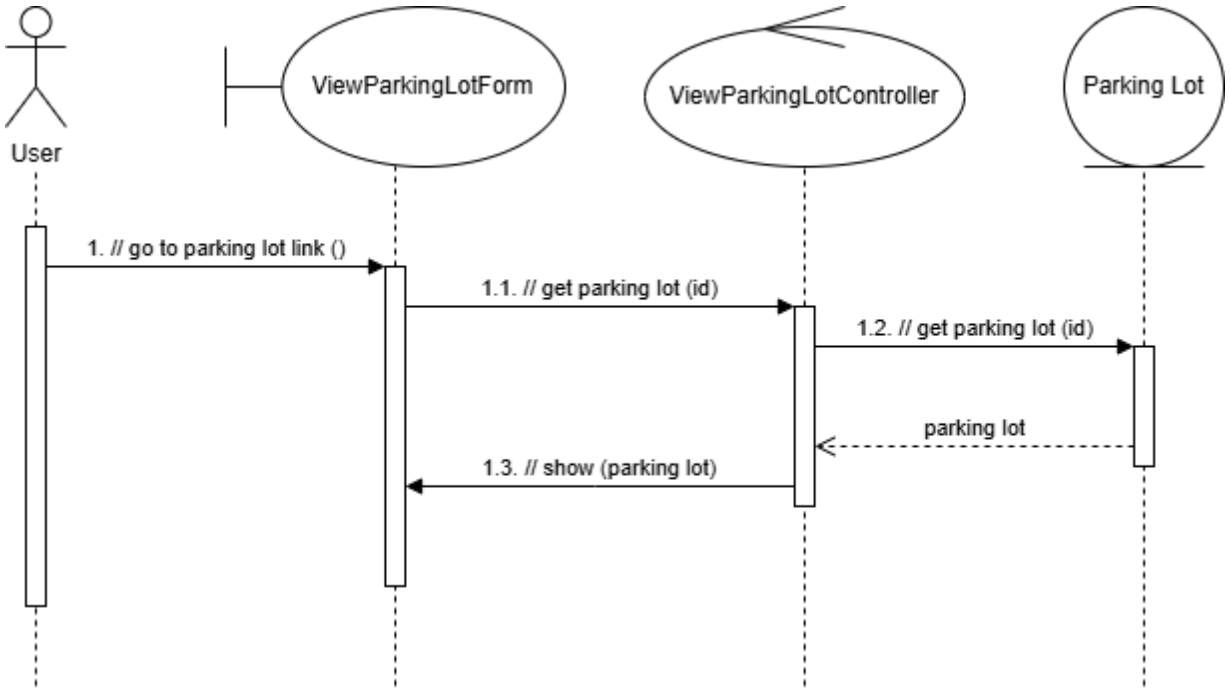

Figure 2-3. Sequence Diagram for the View Parking Lot Information *use case*

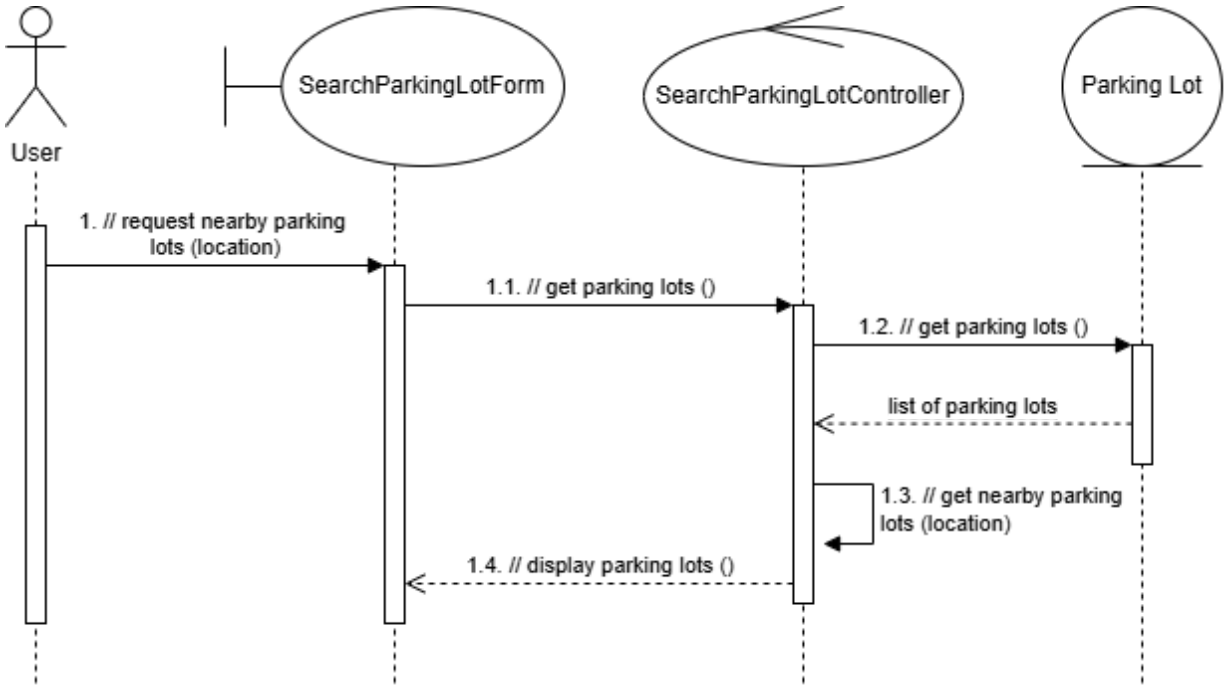

Figure 2-4. Sequence Diagram for the Search Parking Lots *use case*

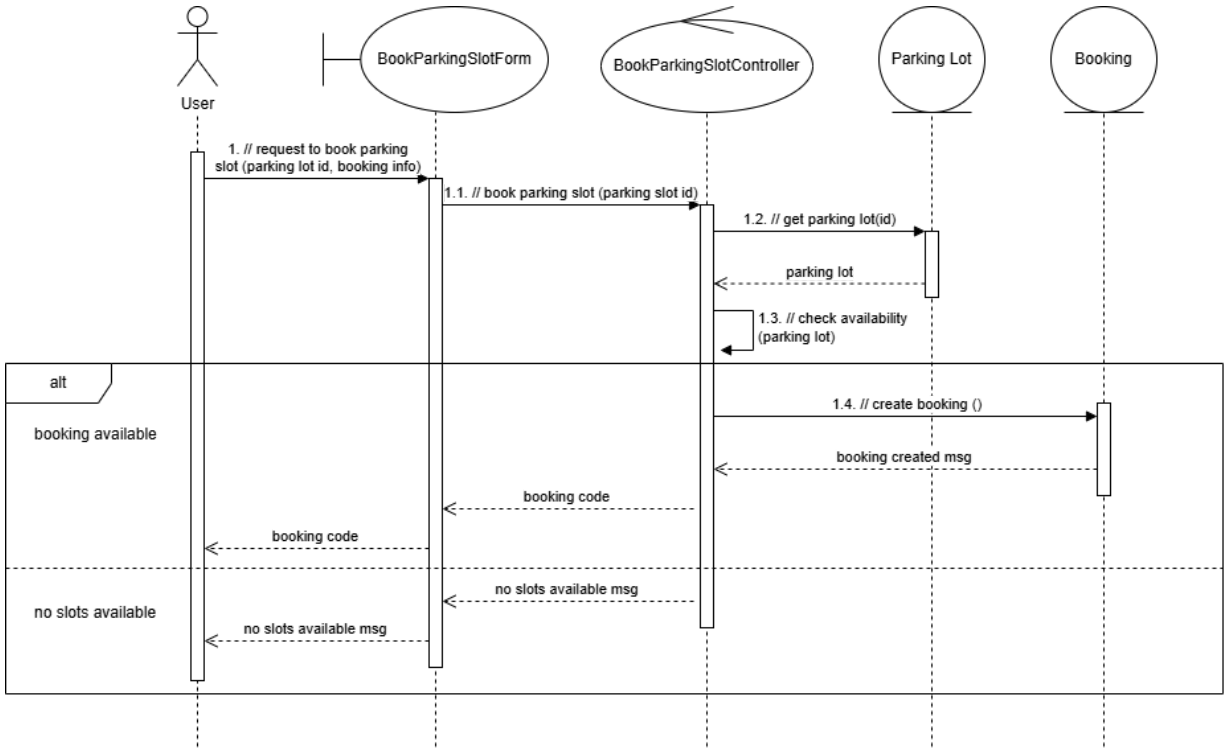

Figure 2-5. Sequence Diagram for the Book Parking Slot *use case*

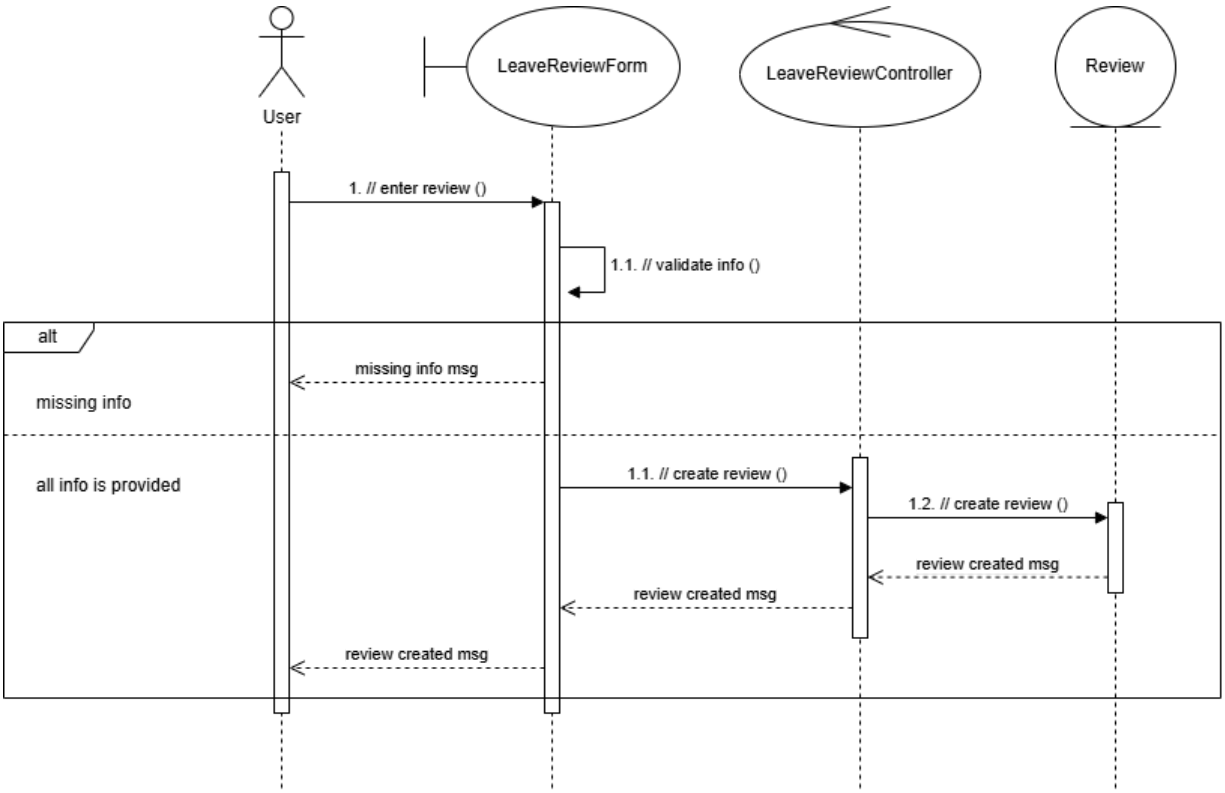

Figure 2-6. Sequence Diagram for the Leave Review *use case*

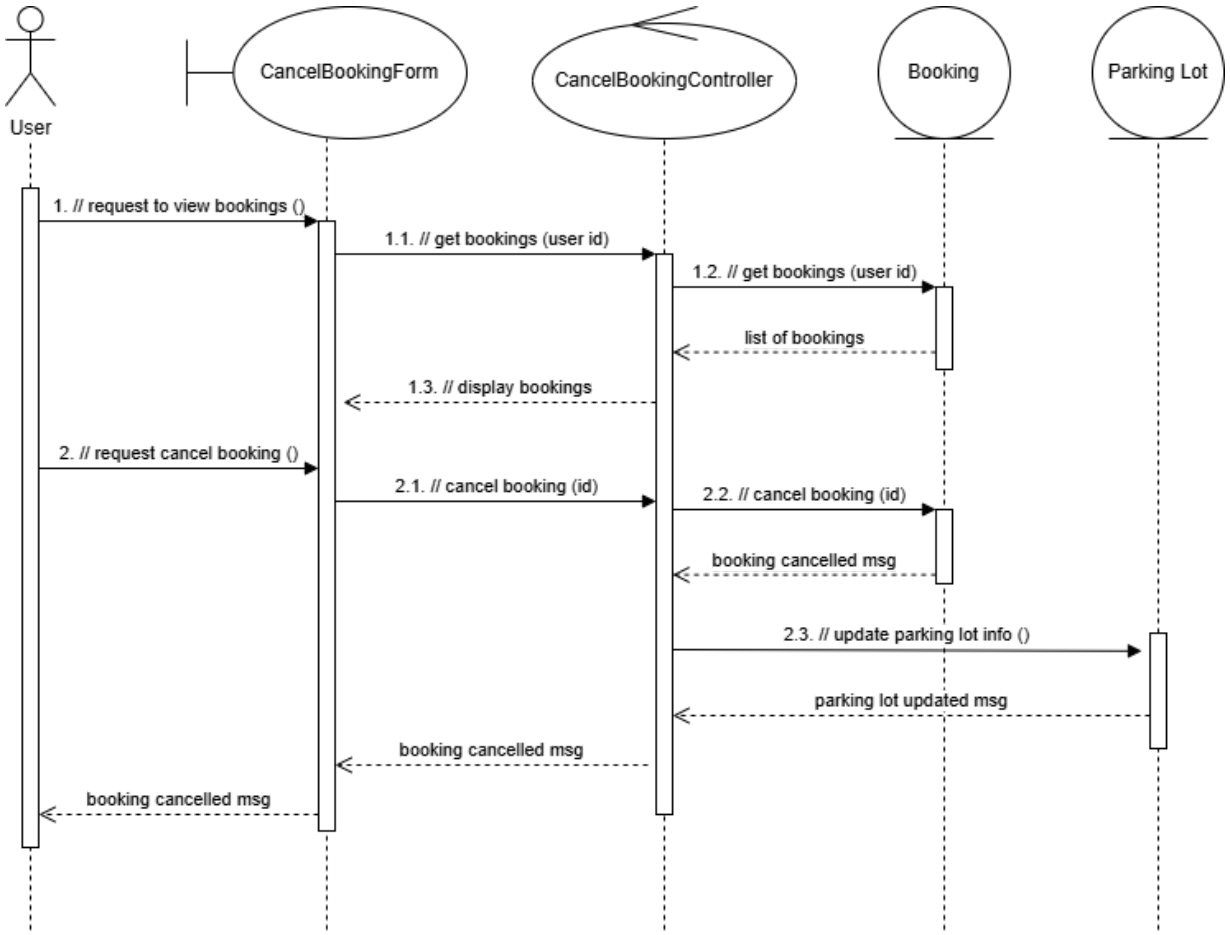

Figure 2-7. Sequence Diagram for the Cancel Booking *use case*

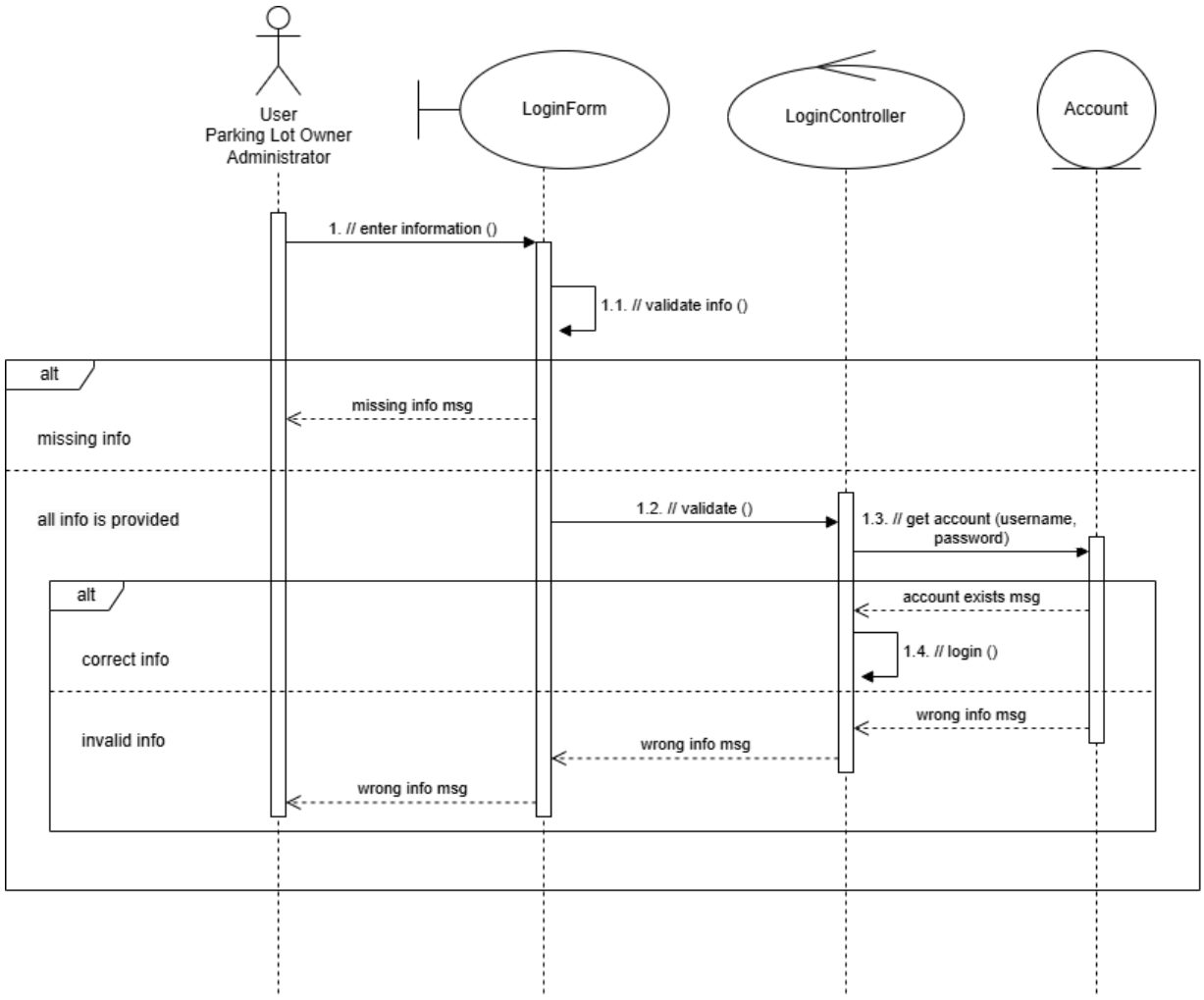

Figure 2-8. Sequence Diagram for the Login *use case*

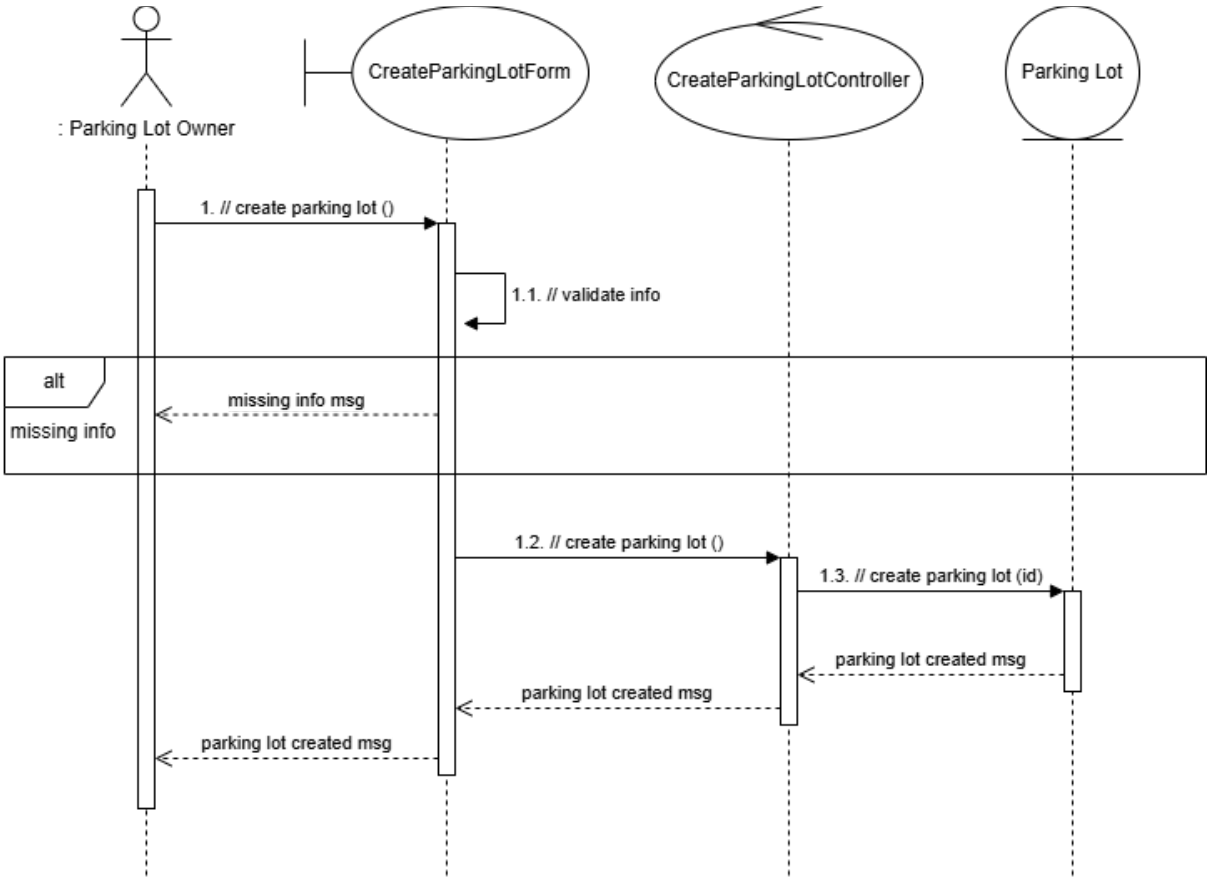

Figure 2-9. Sequence Diagram for the Create Parking Lot *use case*

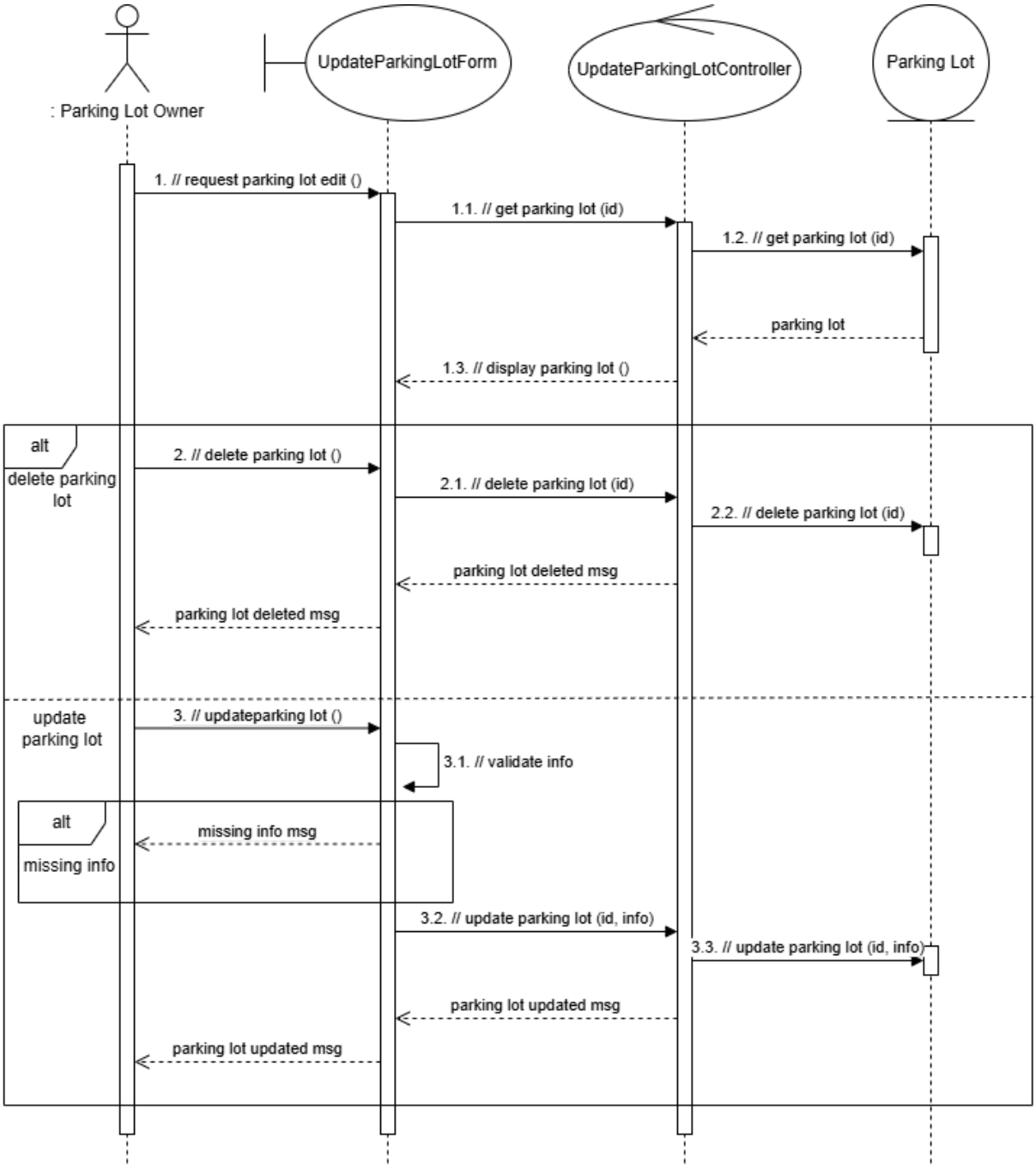

Figure 2-10. Sequence Diagram for the Update Parking Lot *use case*

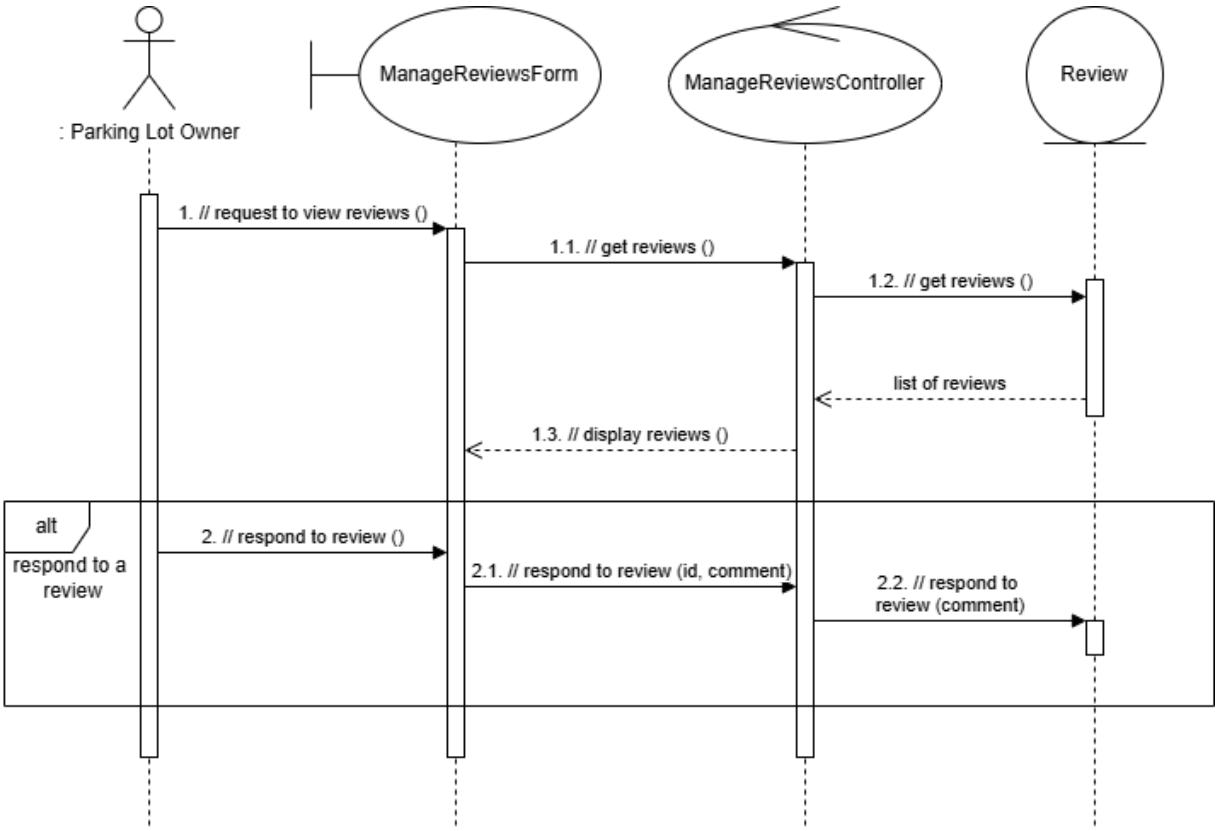

Figure 2-11. Sequence Diagram for the Manage Reviews *use case*

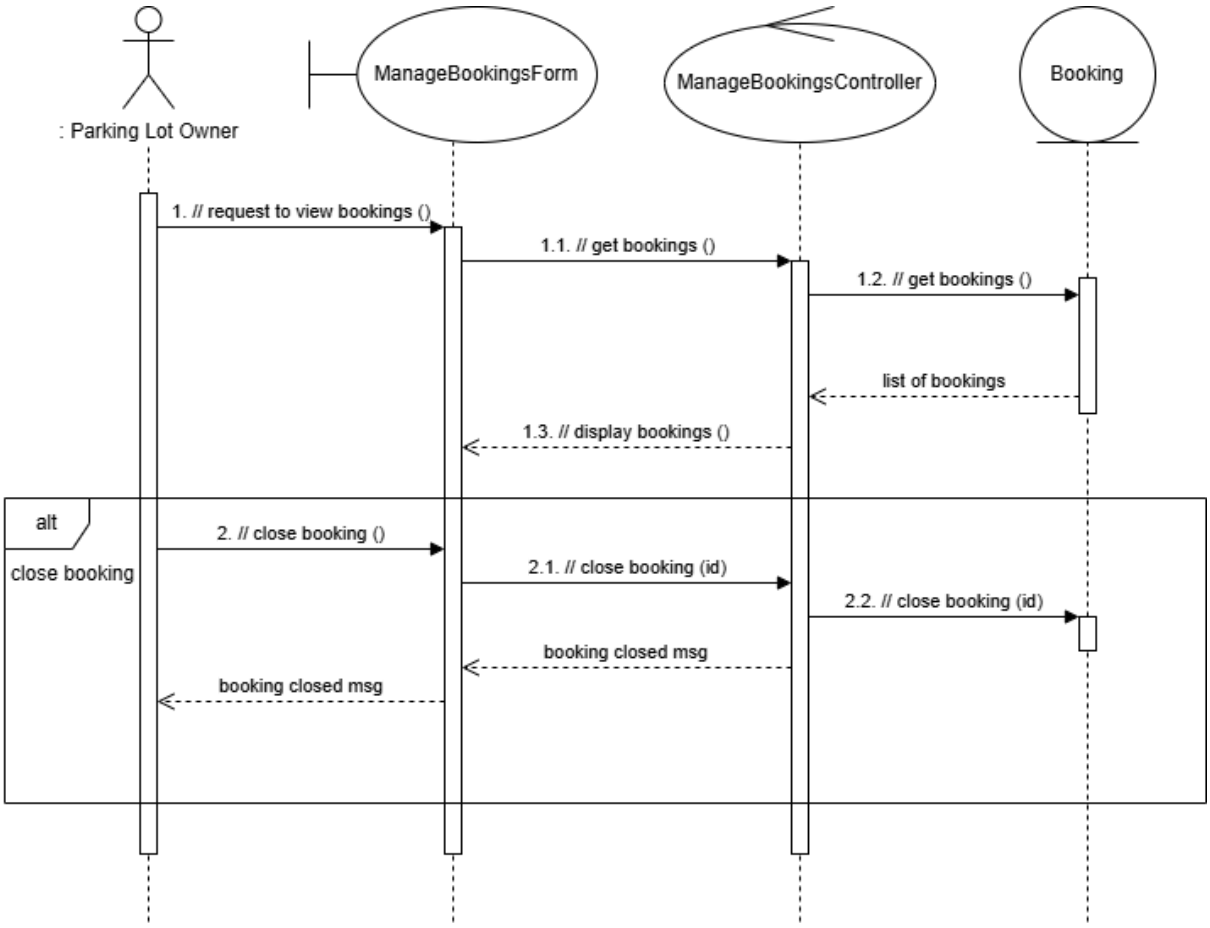

Figure 2-12. Sequence Diagram for the Manage Bookings *use case*

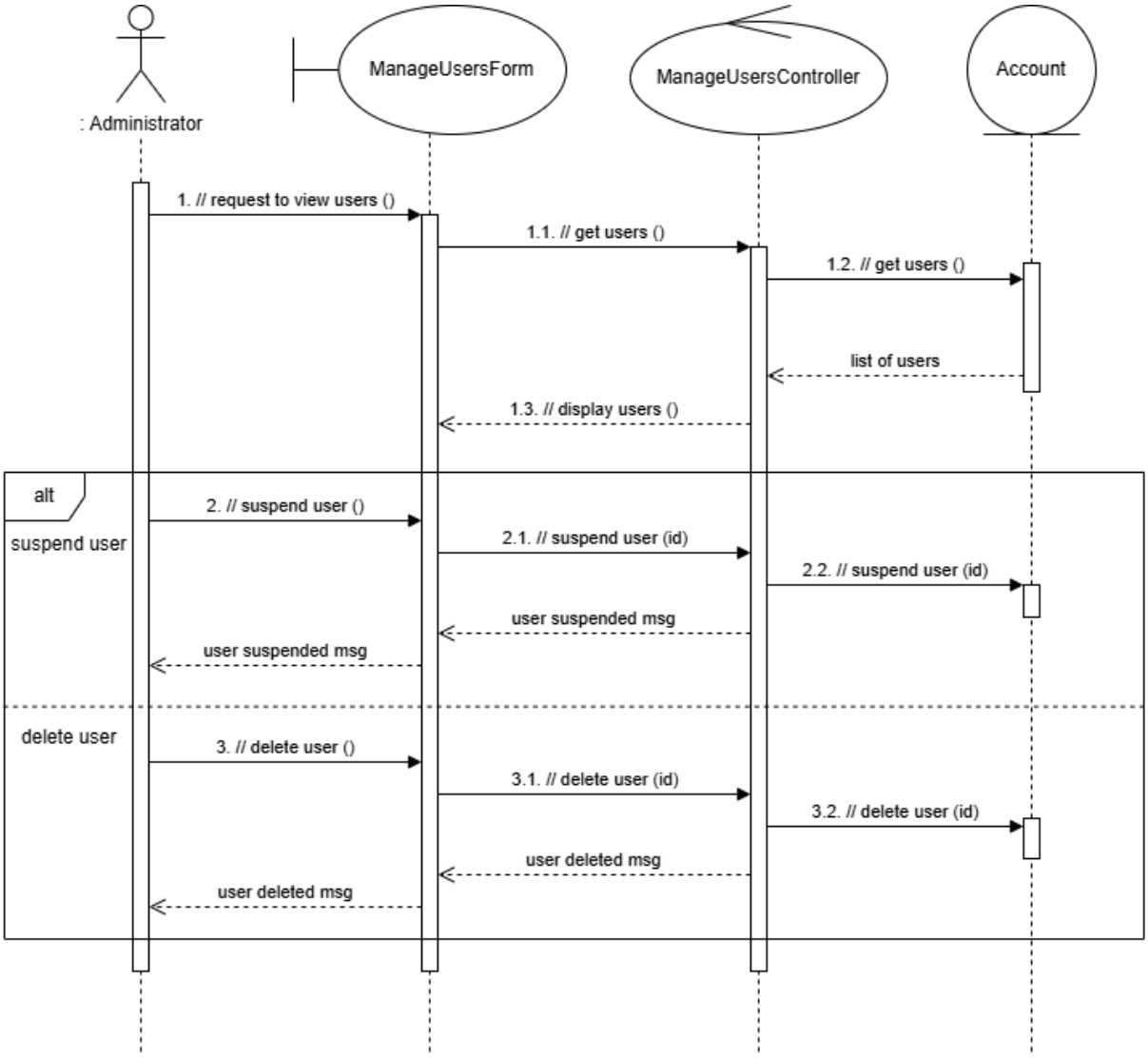

Figure 2-13. Sequence Diagram for the Manage Users *use case*

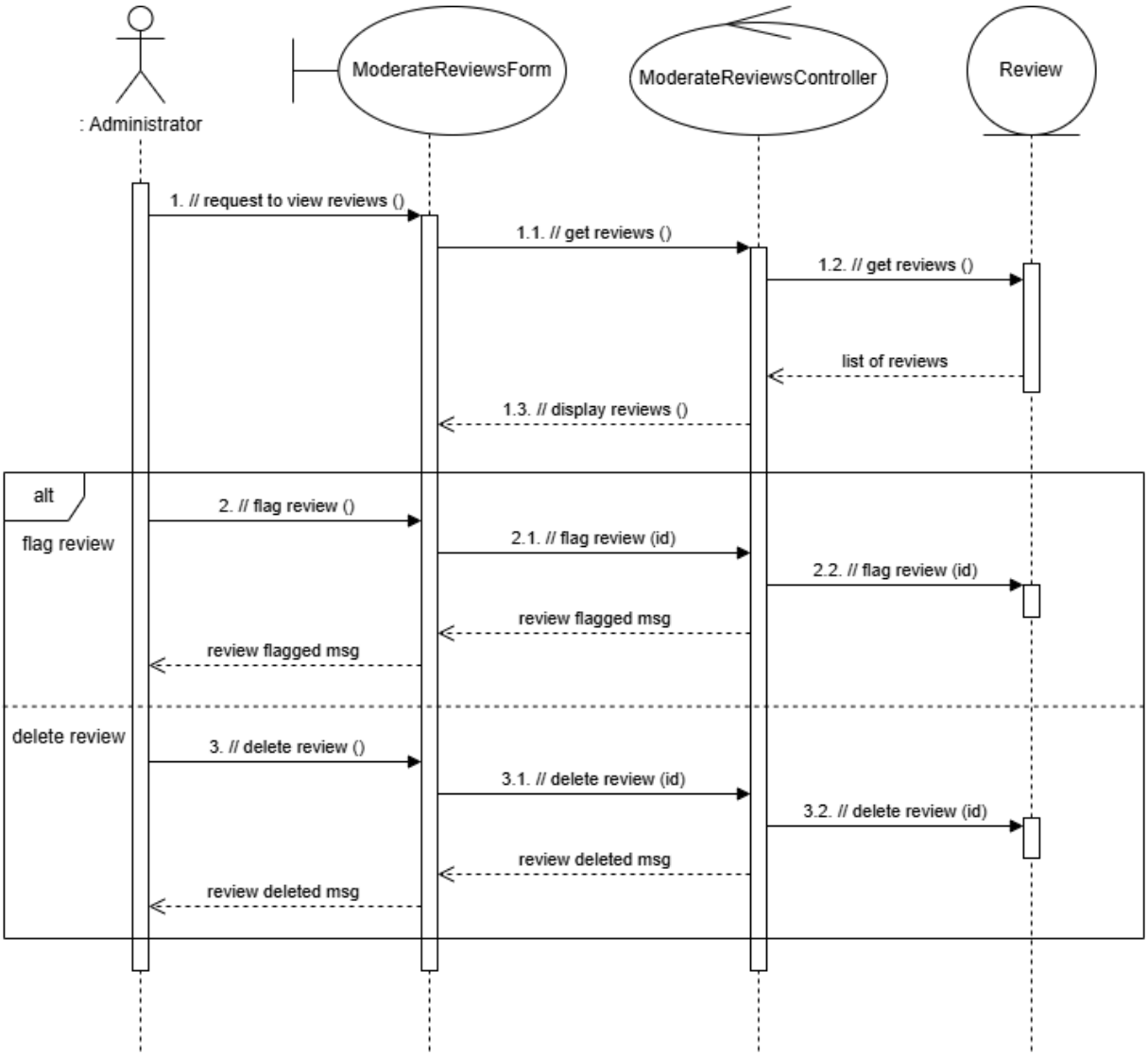

Figure 2-14. Sequence Diagram for the Moderate Reviews *use case*

2.2.2. Use-case realizations: views of participation classes

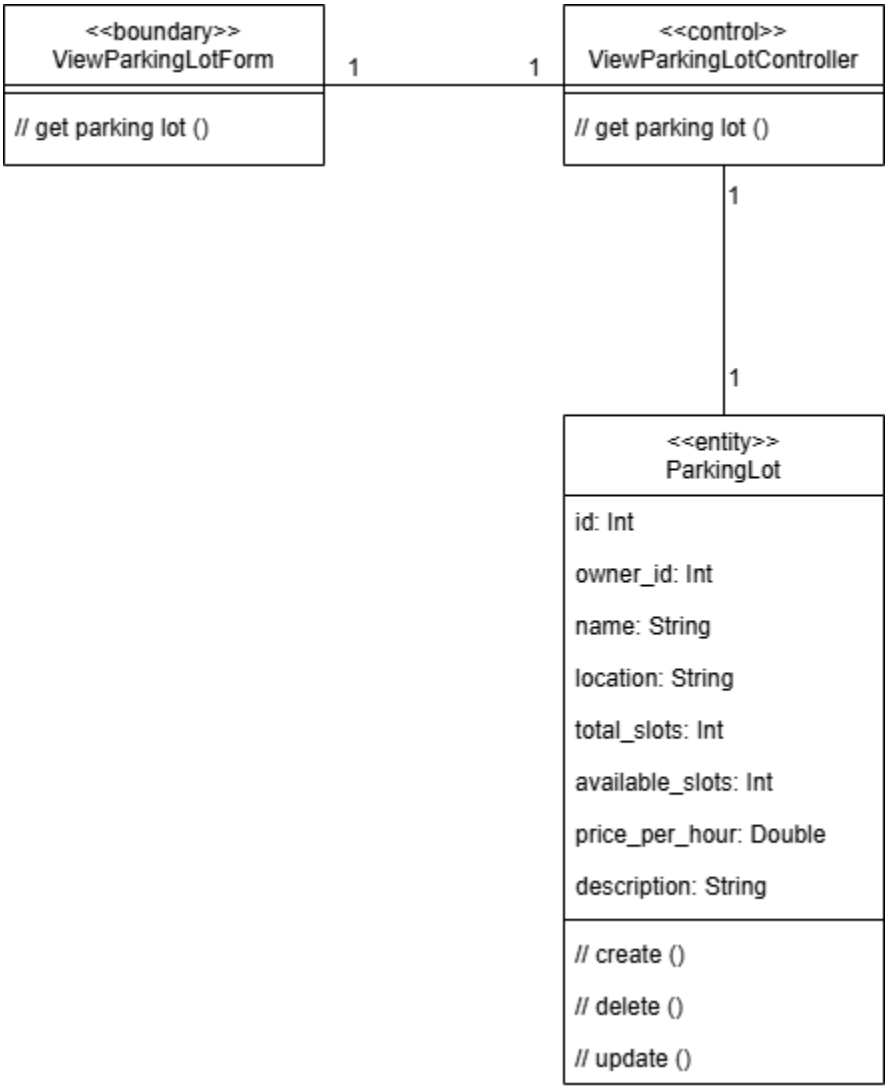

Figure 2-15. VOPC for the View Parking Lot Information *use case*

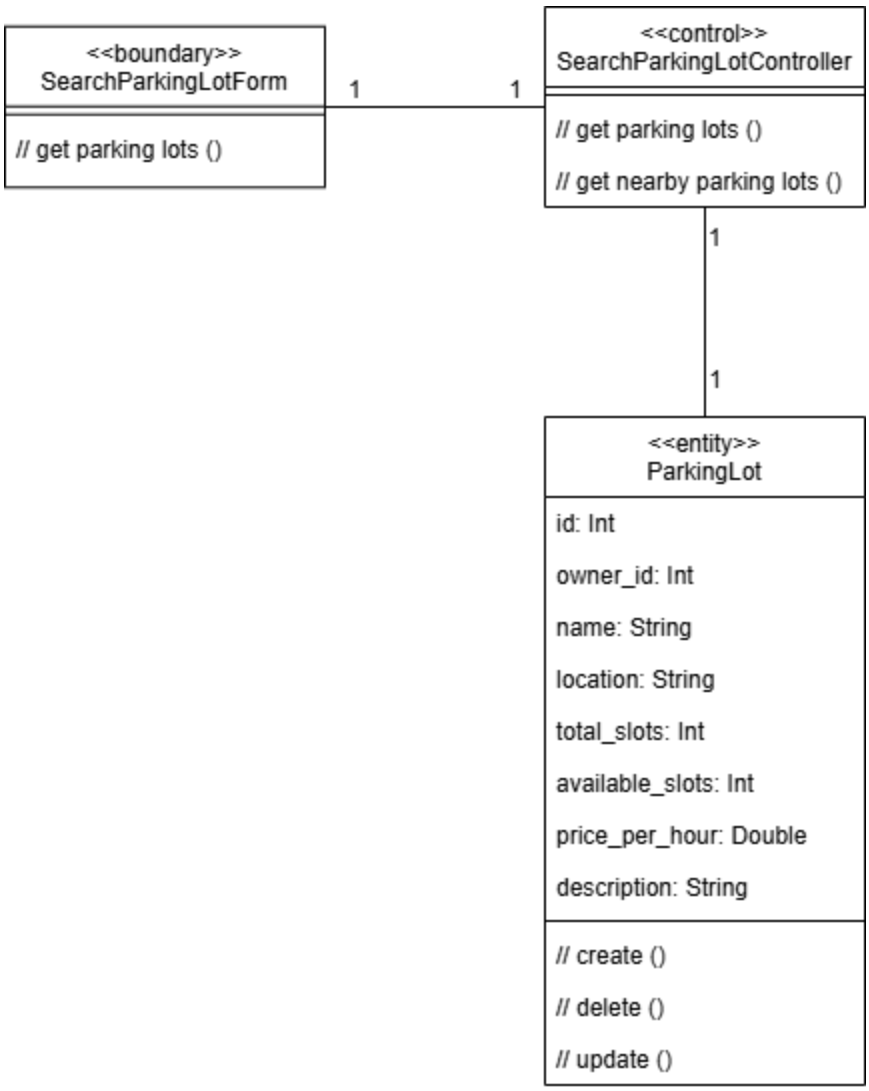

Figure 2-16. VOPC for the Search Parking Lots *use case*

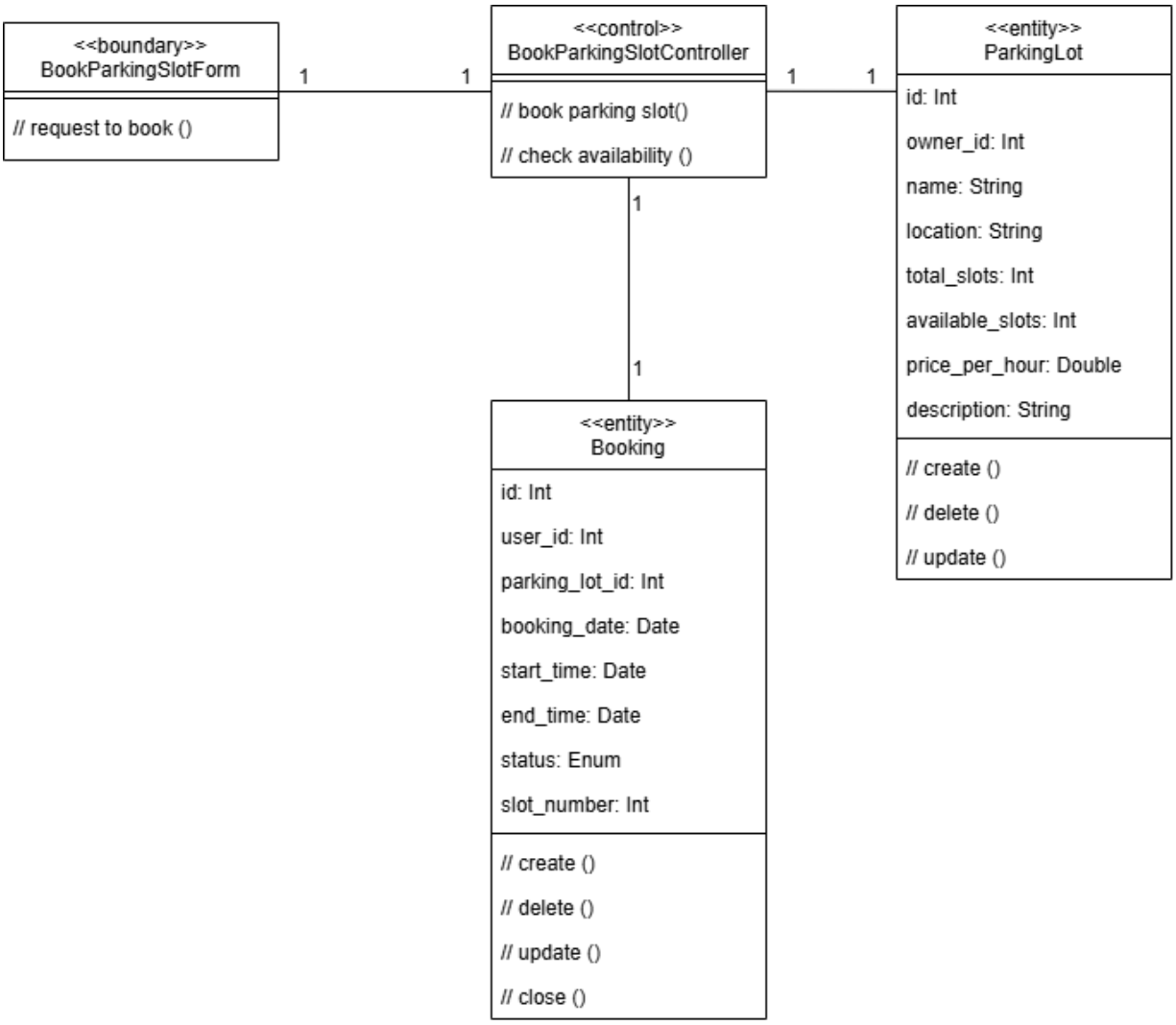

Figure 2-17. VOPC for the Book Parking Slot *use case*

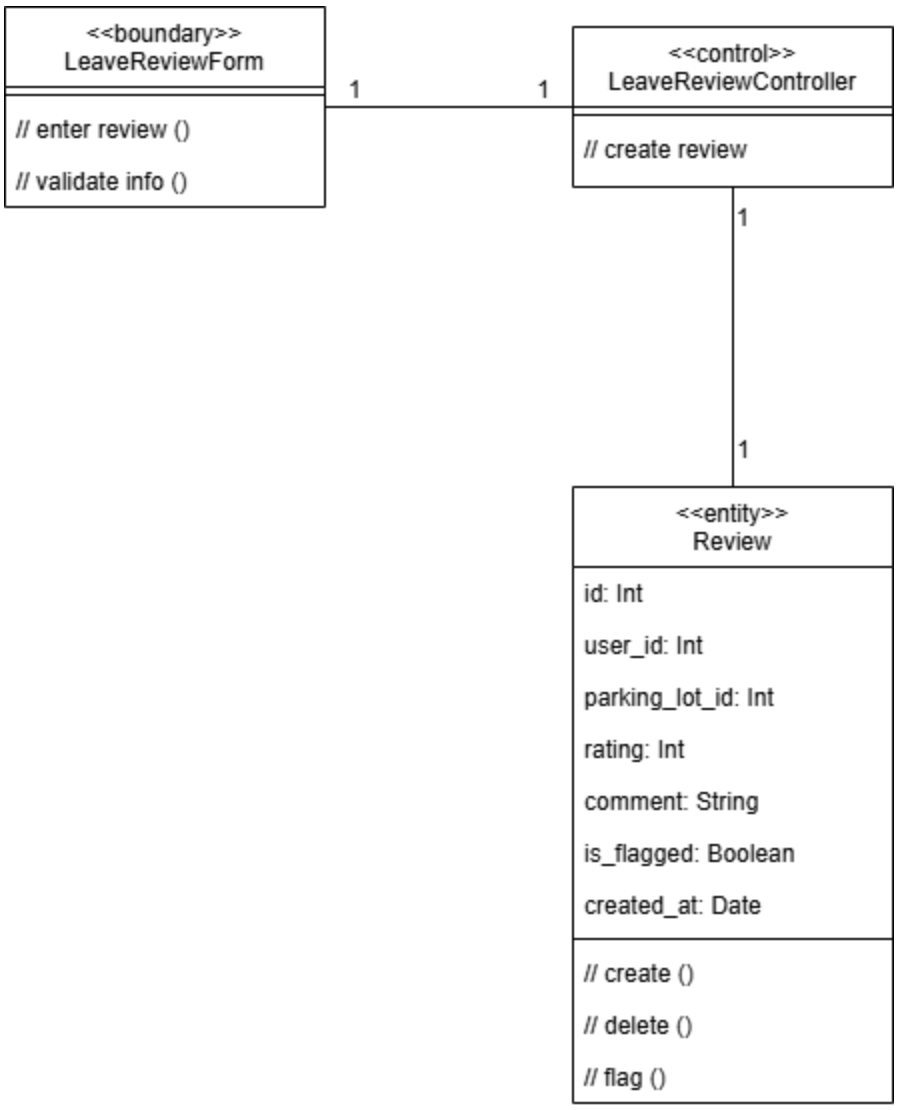

Figure 2-18. VOPC for the Leave Review *use case*

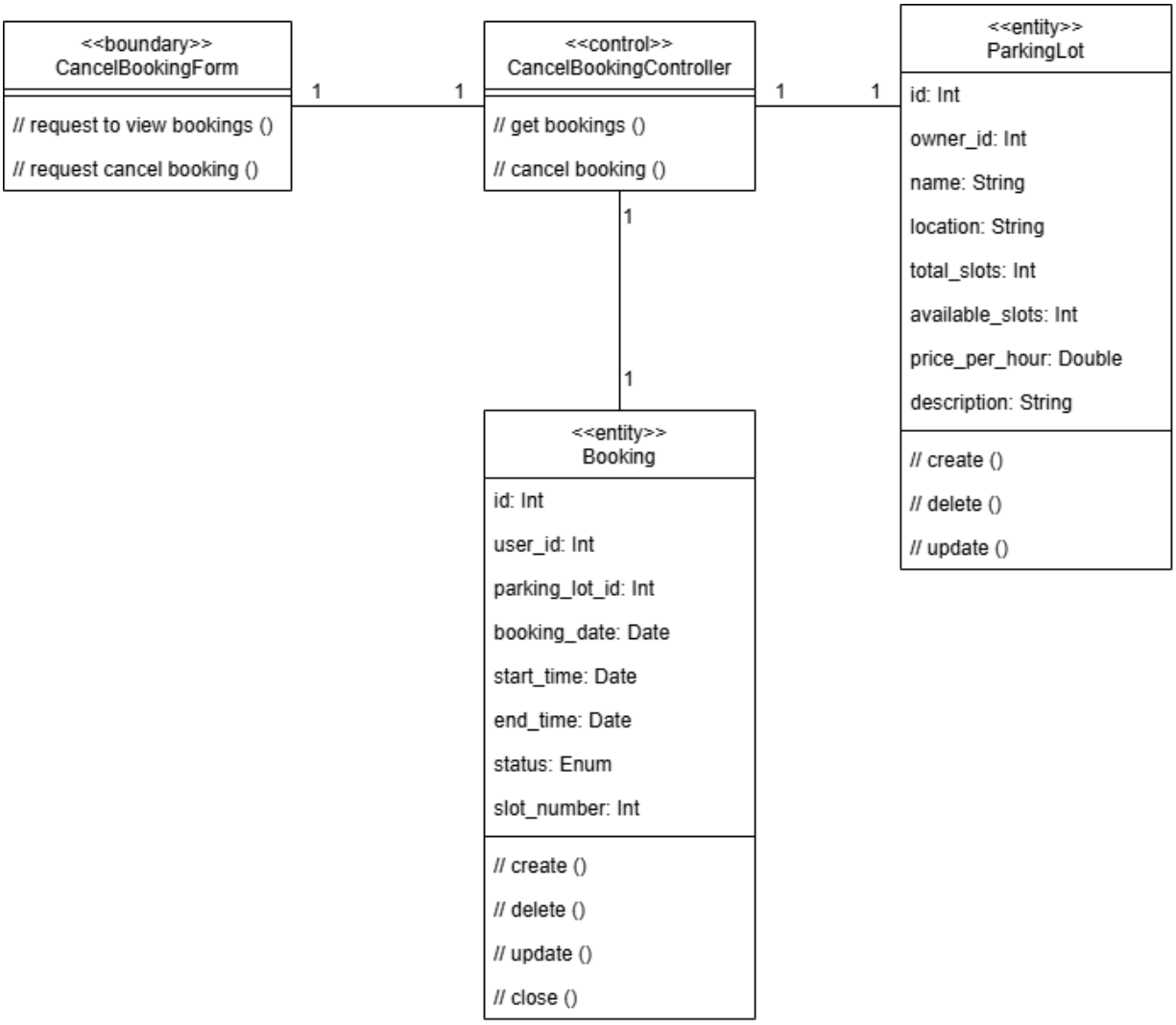

Figure 2-19. VOPC for the Cancel Booking *use case*

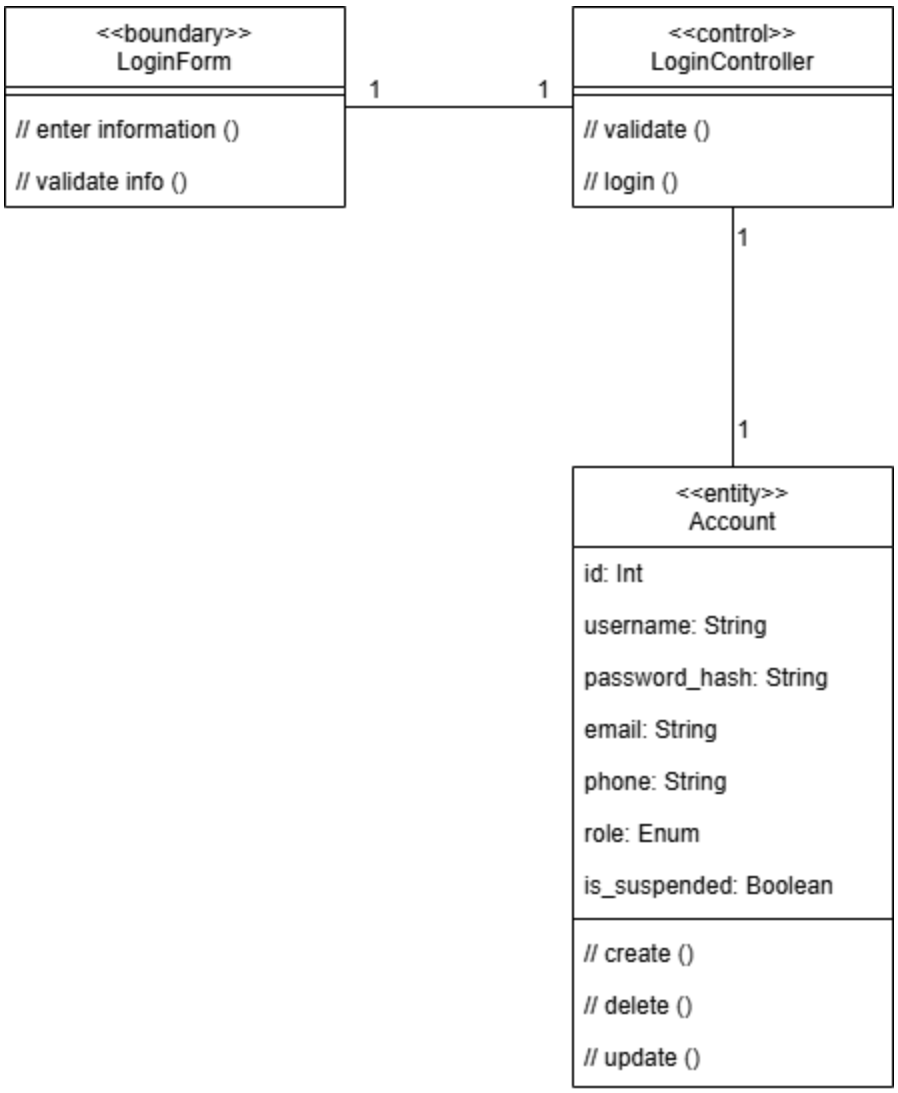

Figure 2-20. VOPC for the Login *use case*

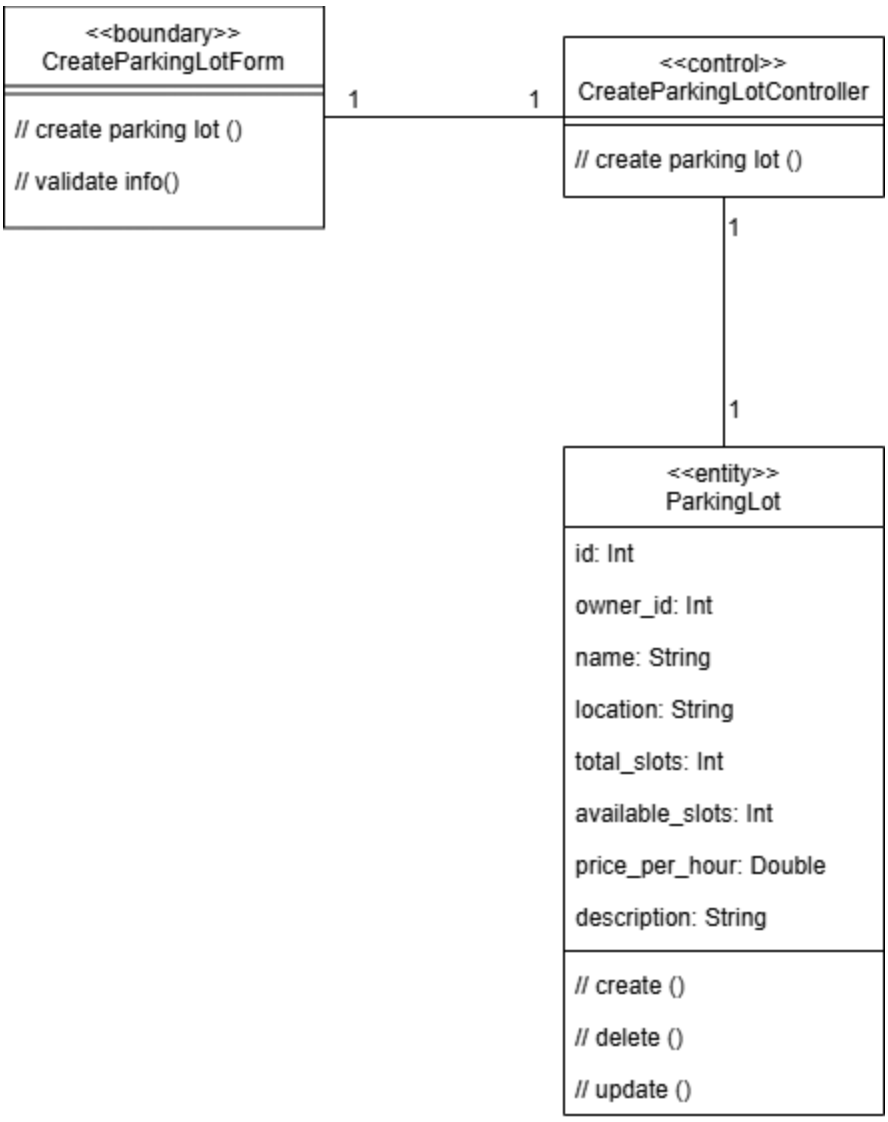

Figure 2-21. VOPC for the Create Parking Lot *use case*

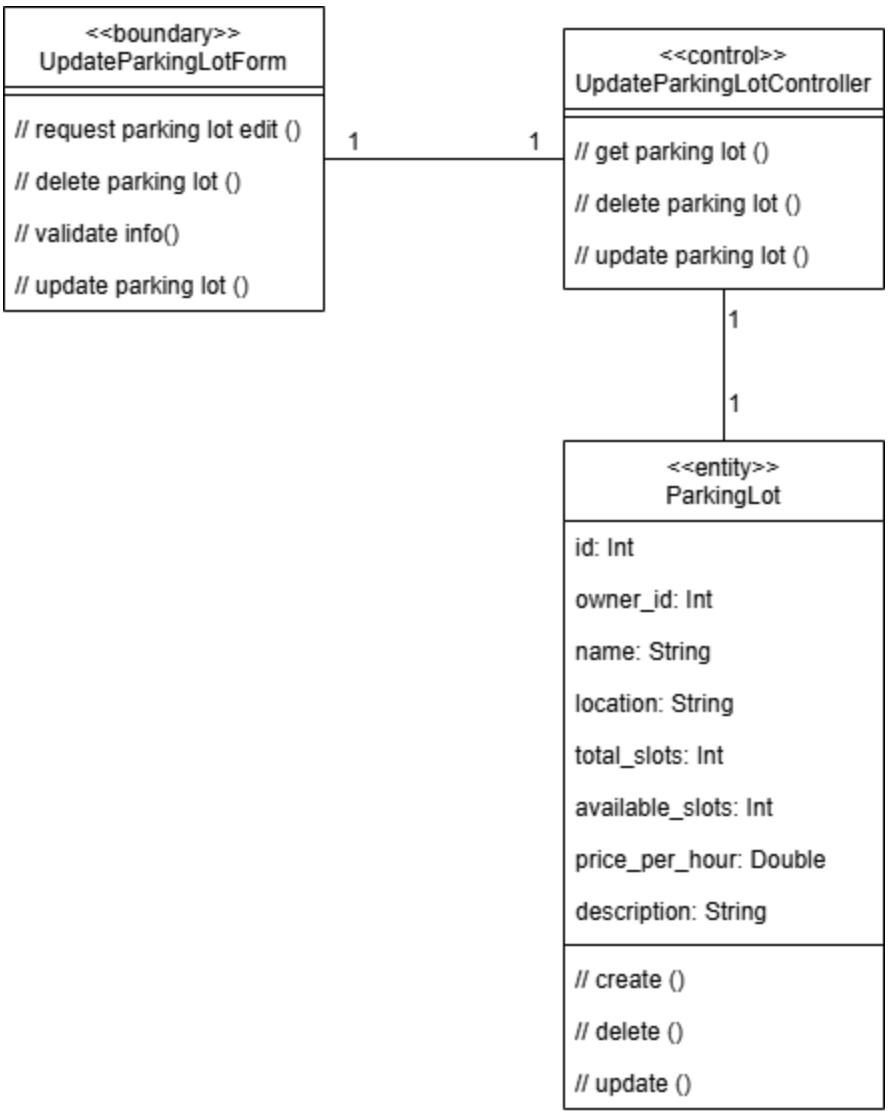

Figure 2-22. VOPC for the Update Parking Lot *use case*

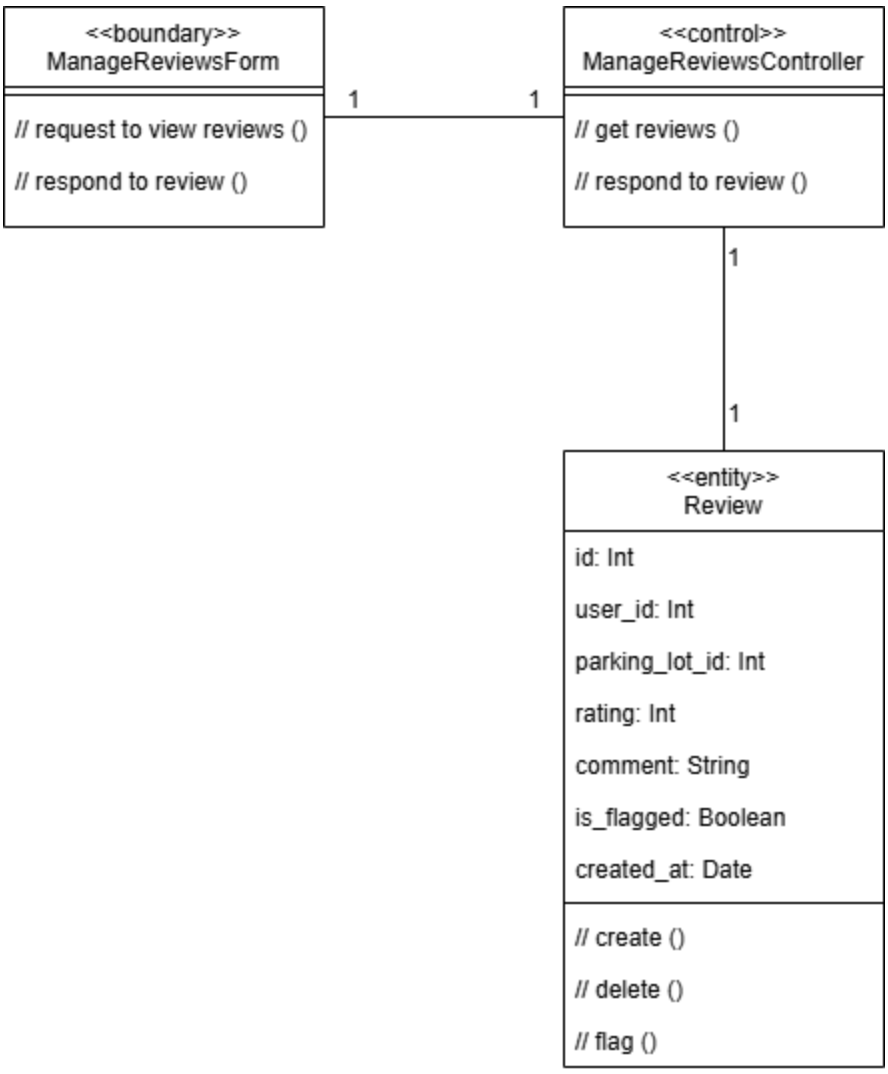

Figure 2-23. VOPC for the Manage Reviews *use case*

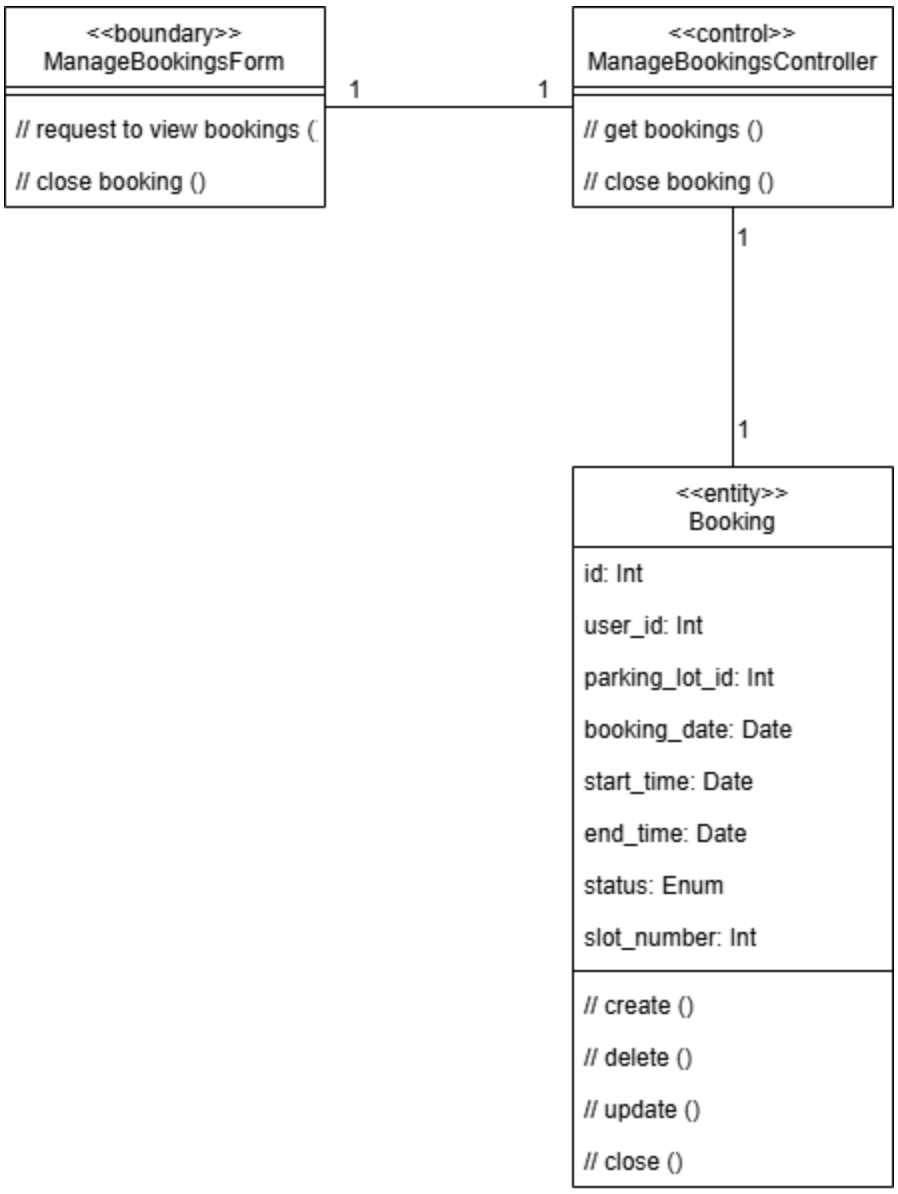

Figure 2-24. VOPC for the Manage Bookings *use case*

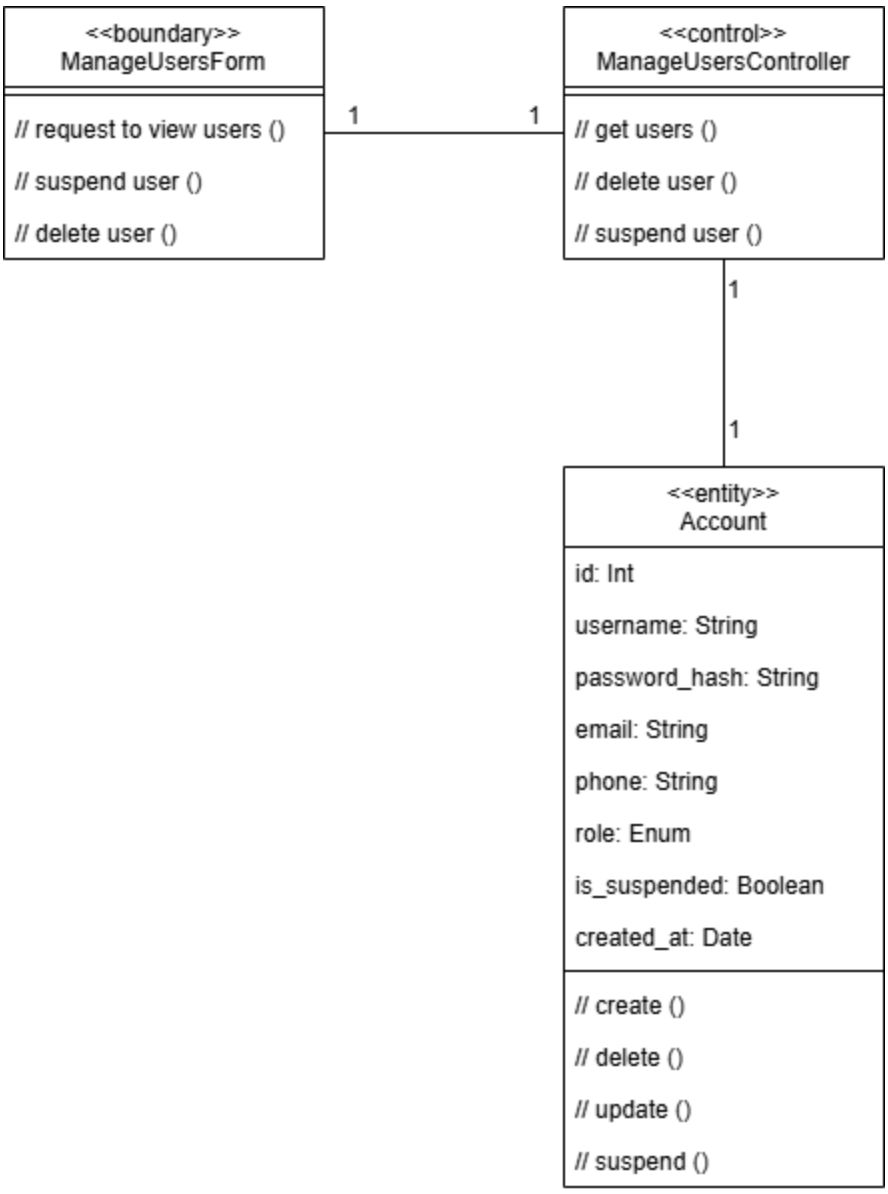

Figure 2-25. VOPC for the Manage Users *use case*

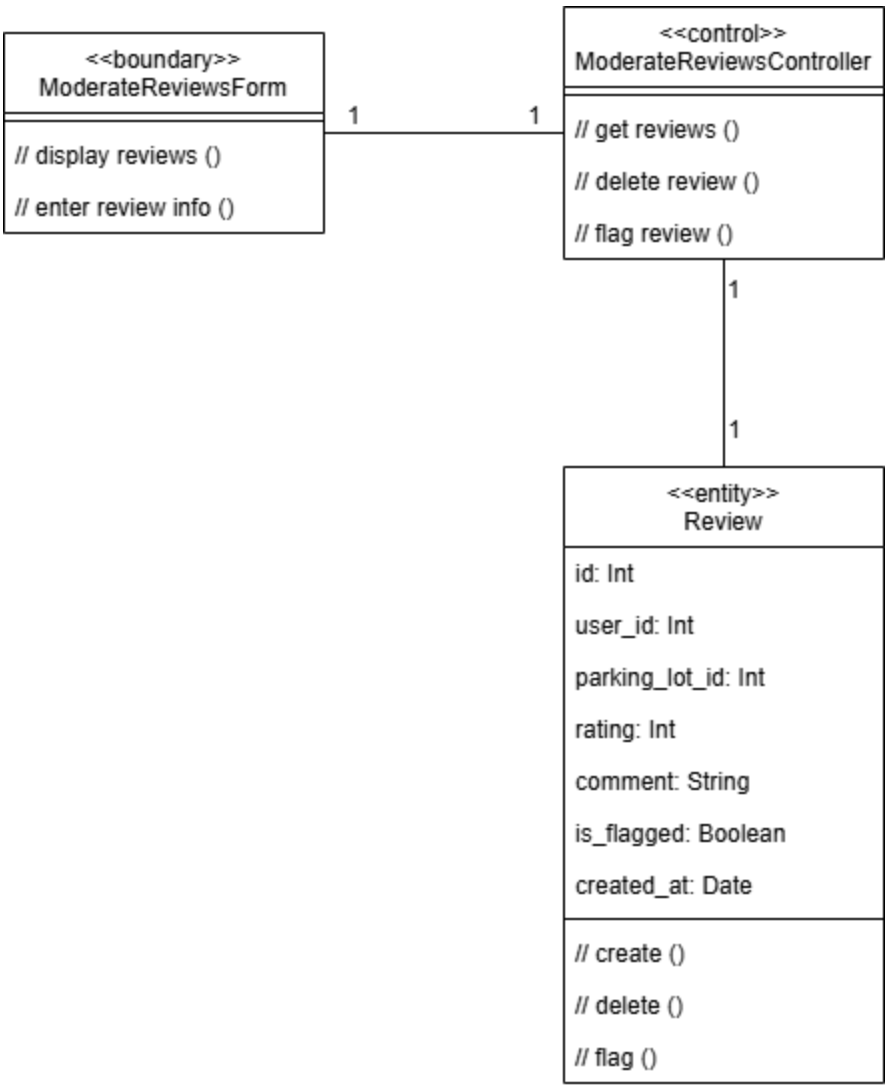

Figure 2-26. VOPC for the Moderate Reviews *use case*

2.2.3. Describe analysis mechanism

Analysis class	Analysis mechanism
Account	Persistency, Security
ParkingLot	
Booking	
Review	
ViewParkingLotInfoController	Distribution
SearchParkingLotsController	
BookParkingSlotController	
LeaveReviewController	
CancelBookingController	
LoginController	
CreateParkingLotController	
UpdateParkingLotController	
ManageReviewsController	
ManageBookingsController	
ManageUsersController	
ModerateReviewsController	

Table 2-1. Analysis-Class-to-Analysis-Mechanism map

Analysis mechanism characteristics

Security

- **Data Granularity:** Attribute level, ensuring field-level protection for sensitive information such as passwords, payment details, and user profiles.
- **User Granularity**:
    - **Registered Users**: Can view and edit their profiles, search for parking lots, book slots, and leave reviews.
    - **Parking Lot Owners**: Can manage their parking lots, bookings, and respond to reviews.
    - **Administrators**: Have full access, including user management, parking lot approval, and system moderation.
- **Security Rules**:

- Only registered users, parking lot owners, and administrators may log into the system.
- Only logged-in users can:
  - View and edit their own account profiles.
  - Search for parking lots and make bookings.
  - Leave reviews on parking lots they have booked.
- Parking lot details can only be edited by their respective owners.
- Bookings can only be closed or updated by the associated parking lot owner.
- Administrators may perform the following actions:
  - Moderate and delete reviews flagged as inappropriate.
  - Suspend or delete user accounts for policy violations.
  - View system-wide reports and logs for audit purposes.
  - Manage critical system settings to maintain integrity and performance.

Persistency

Class		Account	Parking Lot	Booking	Review
Granularity		50 KB per account	10 to 100 KB per lot	2 to 5 KB per booking	1 to 3 KB per review
Volume		Up to 100,000	Up to 50,000	Up to 500,000	Up to 1,000,000
Access frequency	**Create**	500 per day	50 per day	10,000 per day	2,000 per day
	Read	Rarely accessed	5,000 per day	20,000 per day	50,000 per day
	Update	100 per day	100 per day	5,000 per day	1,000 per day
	Delete	10 per day	10 per day	1,000 per day	500 per day

3. Use-case Design

3.1. Architectural refinements

3.1.1. Identify design elements

3.1.1.1. Identifying classes

Analysis class	Design element
Account	Account, Database subsystem
ParkingLot	ParkingLot, Database subsystem
Booking	Booking, Database subsystem
Review	Review, Database subsystem
ViewParkingLotInfoController	Map directly to design classes
SearchParkingLotsController	
BookParkingSlotController	
LeaveReviewController	
CancelBookingController	
LoginController	
CreateParkingLotController	
UpdateParkingLotController	
ManageReviewsController	
ManageBookingsController	
ManageUsersController	
ModerateReviewsController	
ViewParkingLotInfoForm	
SearchParkingLotsForm	
BookParkingSlotForm	
LeaveReviewForm	
CancelBookingForm	
LoginForm	

CreateParkingLotForm	
UpdateParkingLotForm	
ManageReviewsForm	
ManageBookingsForm	
ManageUsersForm	
ModerateReviewsForm	

Table 3-1. Analysis-Class-to-Design-Element map

3.1.1.2. Identifying subsystems and interfaces

The Database subsystem provides support for relational databases written in the SQL language. The subsystem is designed as follows:

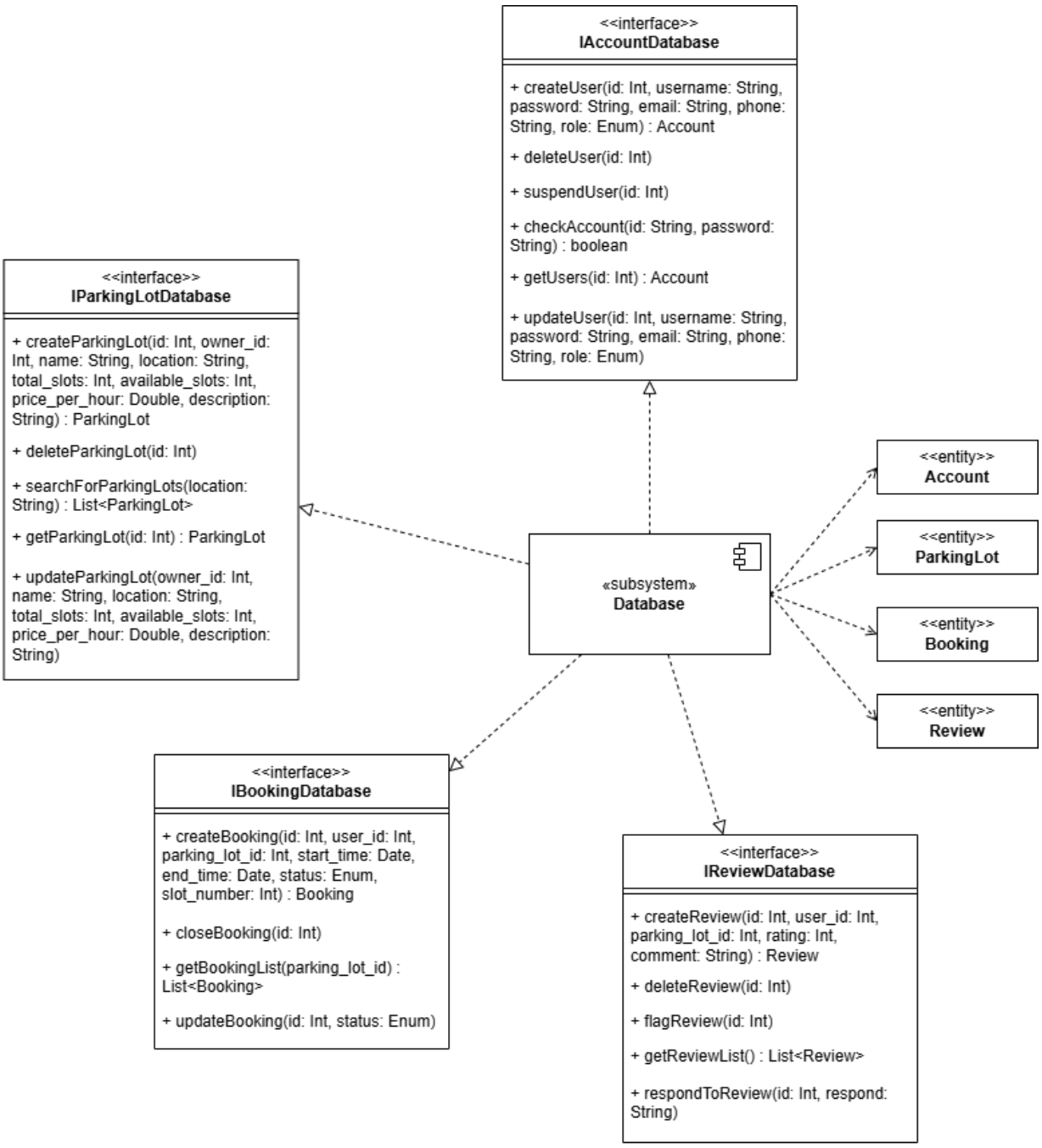

Figure 3-1. The Database *subsystem and its interfaces*

3.1.1.3. Identifying packages

Each layer in the analysis corresponds to a high-level package in the system.

The *Application* package

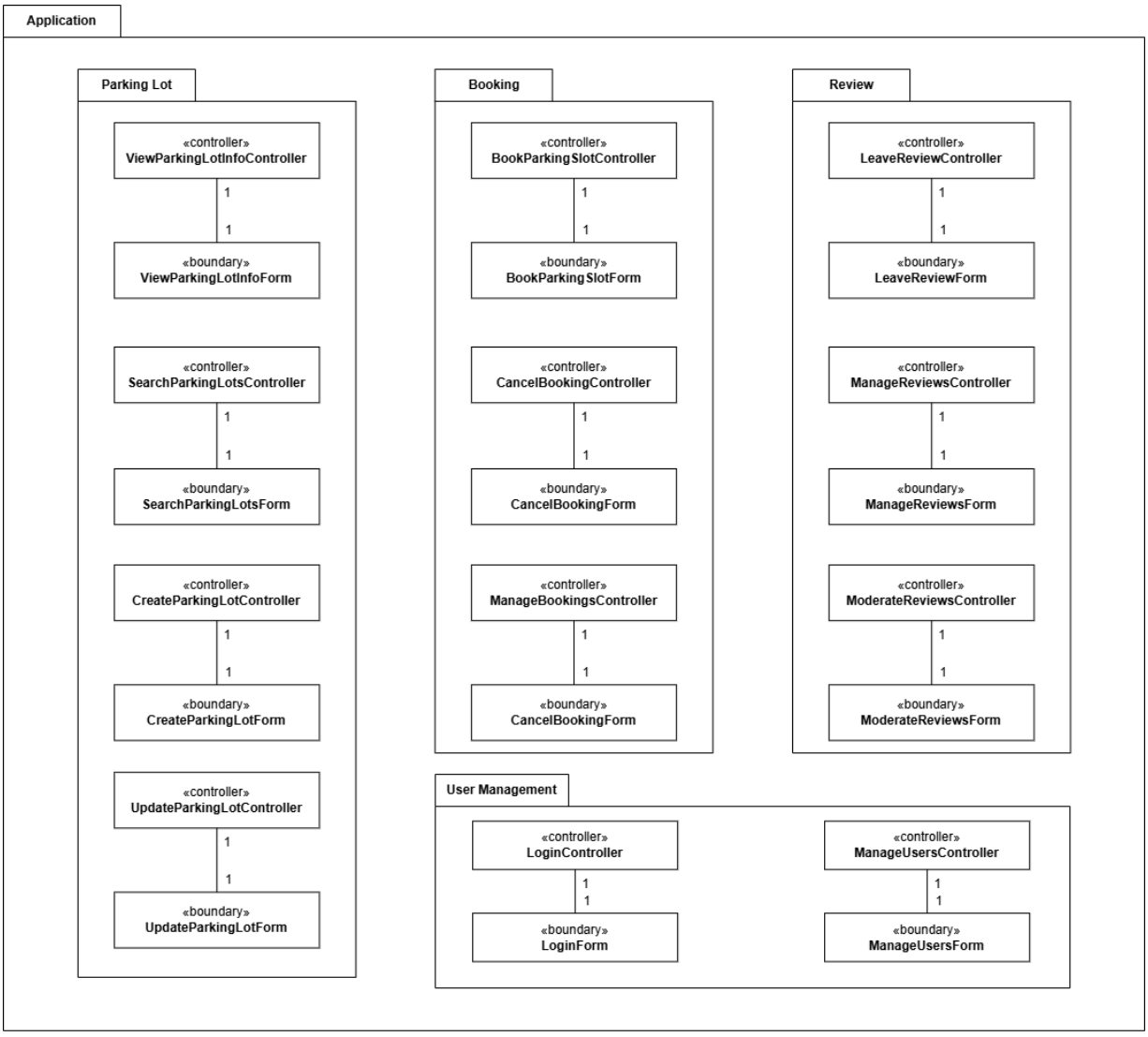

Figure 3-2. The Application *package and its sub-packages*

The Application package contains the boundary and control classes, which are present in the client application. It is further divided into four sub-packages, based on their feature or functionality:

- The *Parking Lot* sub-package handles functionalities related to parking lots (viewing, searching, creating, updating).
- The *Booking* sub-package manages booking-related operations.
- The Review sub-package is responsible for managing user reviews.
- The User Management sub-package manages user accounts and admin functionalities.

The *Business Services* package

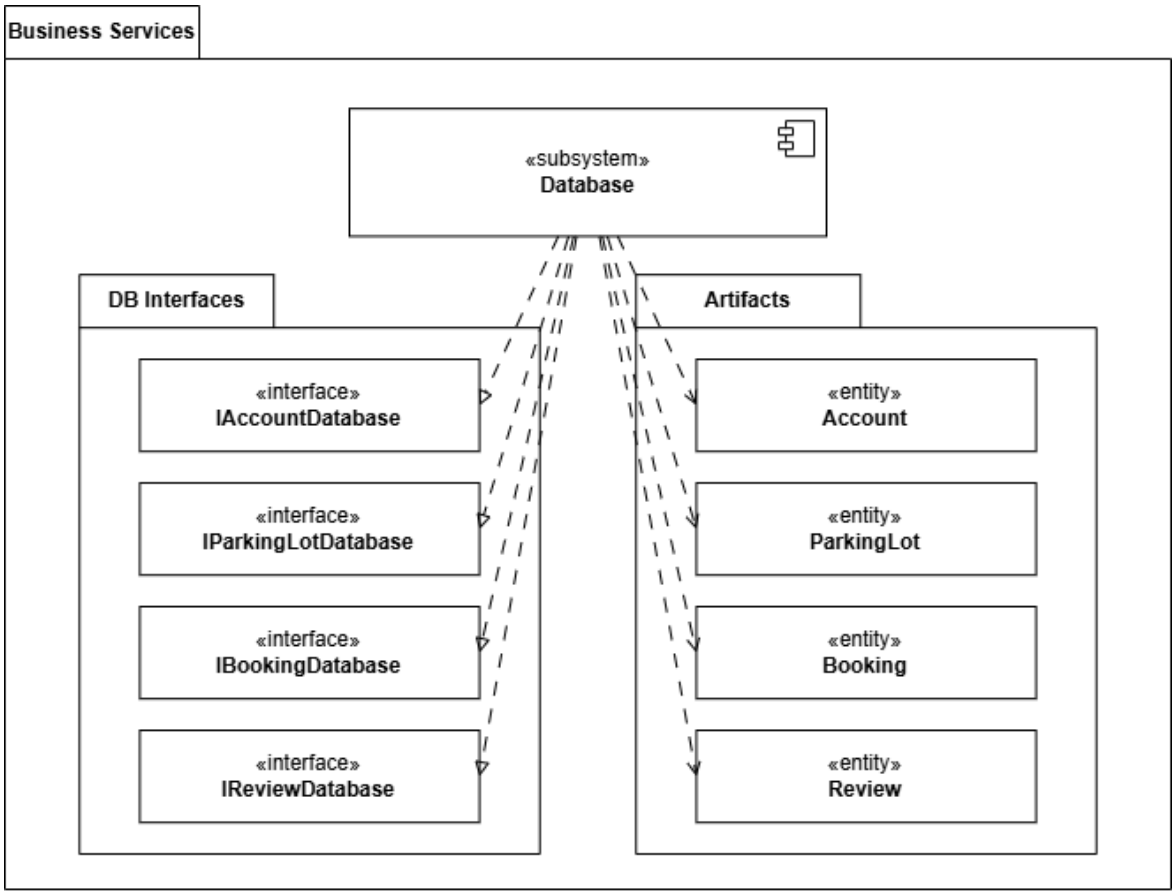

Figure 3-3. The Business Services *package*

The *Business Services* package contains the *Database* subsystem and its interfaces, as well as the entity classes. These elements are common to all use cases.

The *Middleware* package

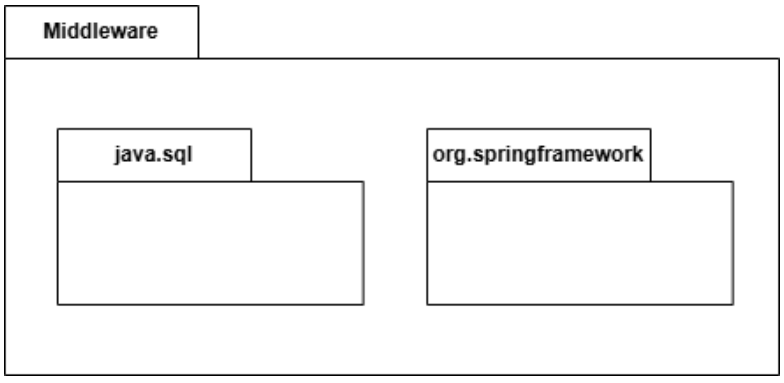

Figure 3-4. The Middleware *package*

The *Middleware* package includes Java's SQL package, which provides access to databases and the Java Spring framework, which provides network services.

Packages and their dependencies

As already stated, the *Application* package depends on the *Business Services* package, which in turn depends on the *Middleware* package.

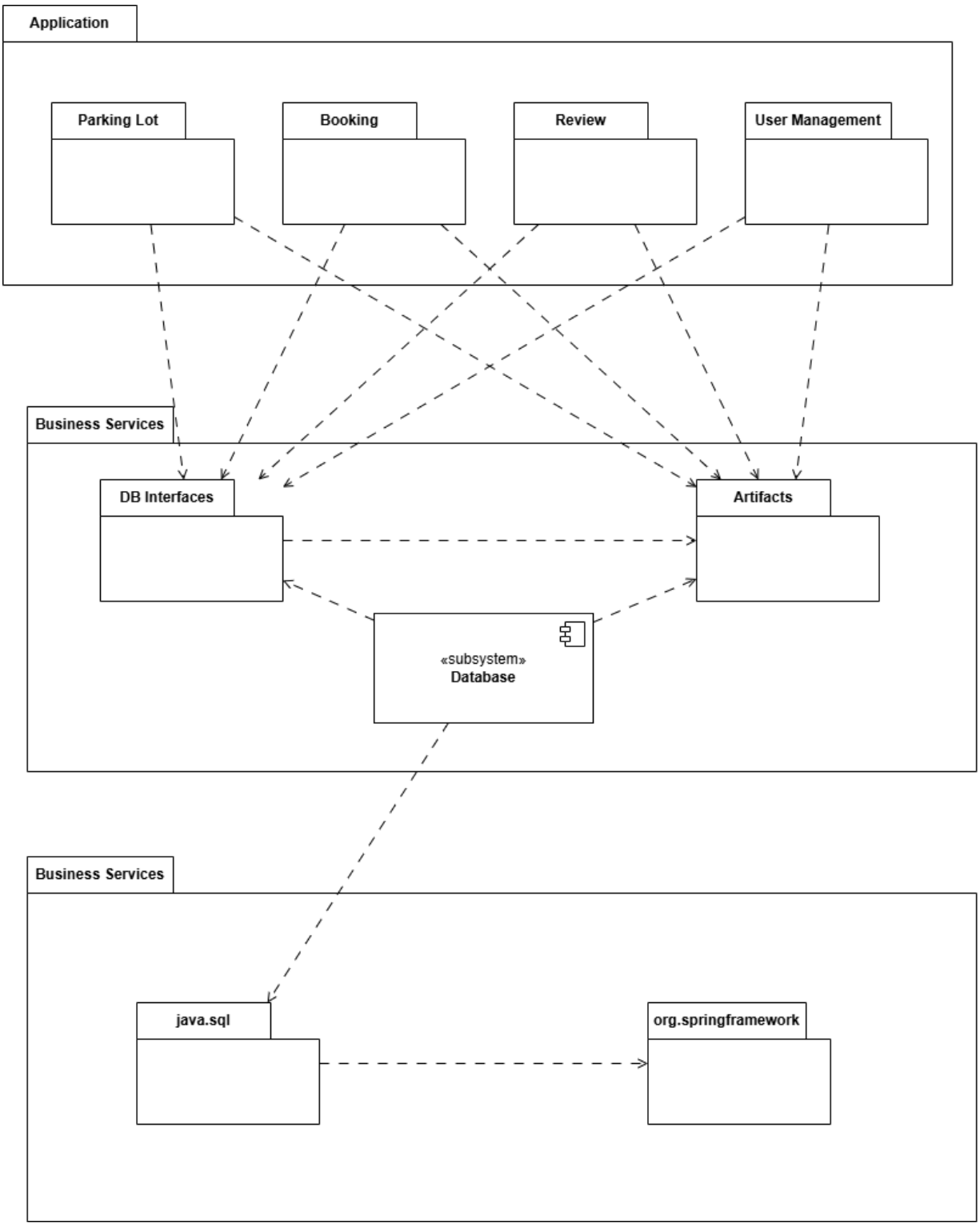

Figure 3-5. Package dependencies diagram

3.1.2. Identify design mechanisms

Analysis mechanism	Design mechanism	Implementation mechanism
Persistency	RDBMS	JDBC
Security	Web tokens	Java Spring framework
Distribution	REST API	Java Spring framework

Table 3-2. Design and implementation mechanisms

3.2. Describe the run-time architecture

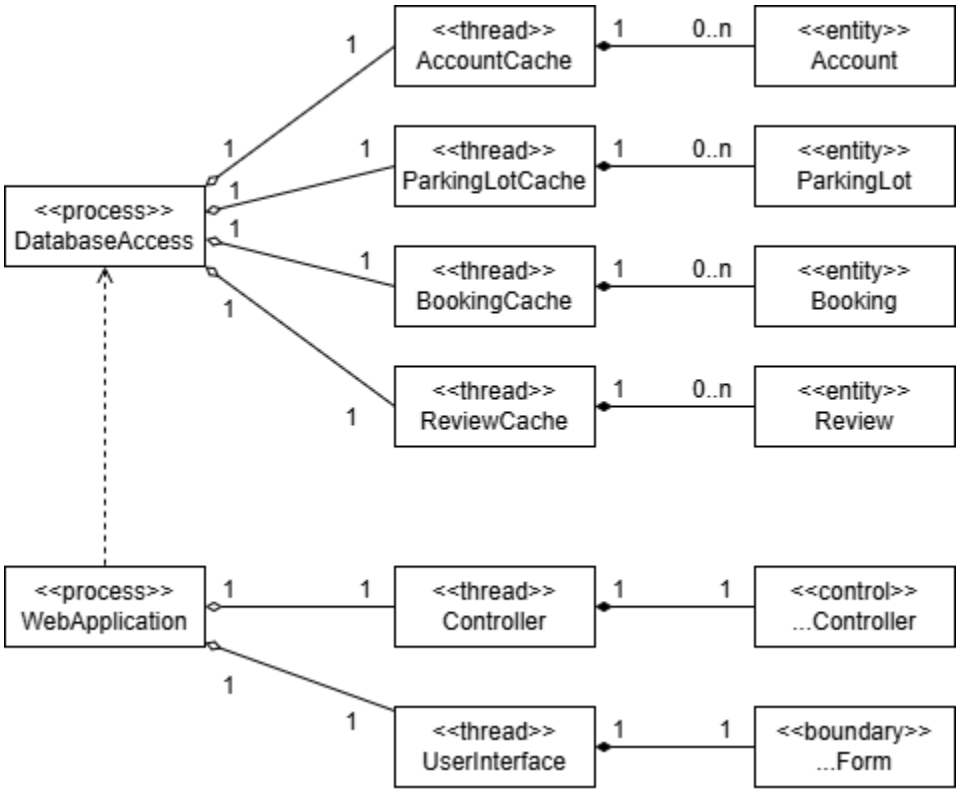

Figure 3-6. The system's process model

3.3. Describe distribution

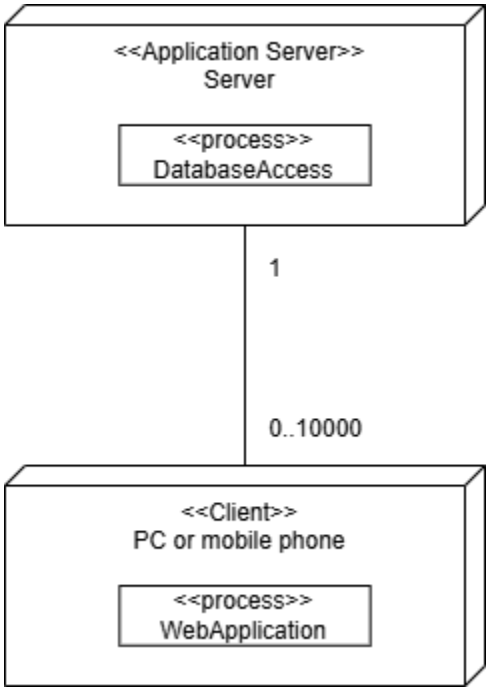

Figure 3-7. The deployment view of the system

3.4. Use-case design

3.4.1. Design sequence diagrams

After incorporating the Database subsystem, the model's sequence diagrams are updated as follows. Some method parameters are elided for conciseness and legibility – they are shown in full in the Class Design section.

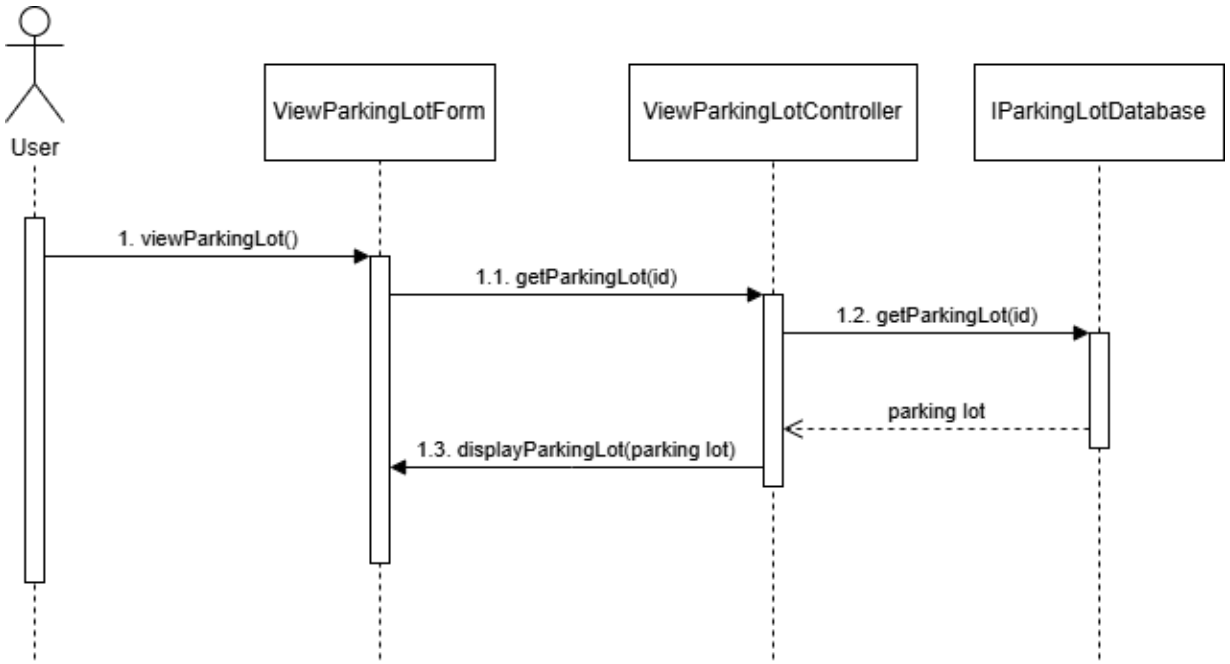

Figure 3-8. Design sequence diagram for the View Parking Lot Information *use case*

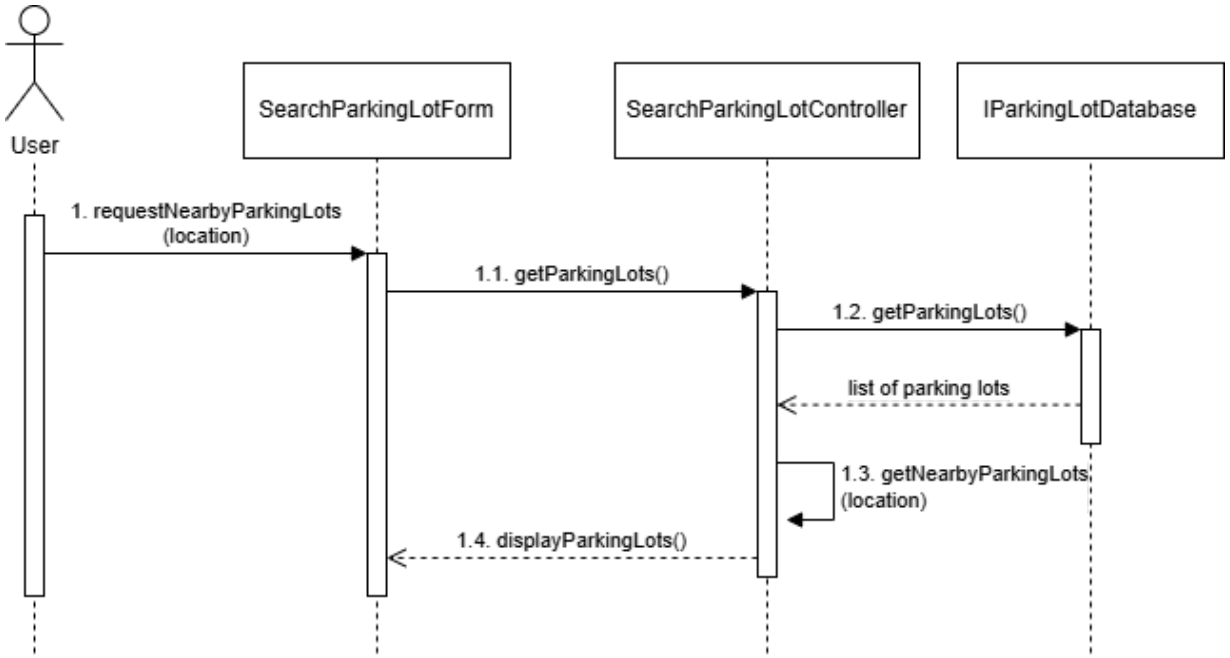

Figure 3-9. Design sequence diagram for the Search Parking Lots *use case*

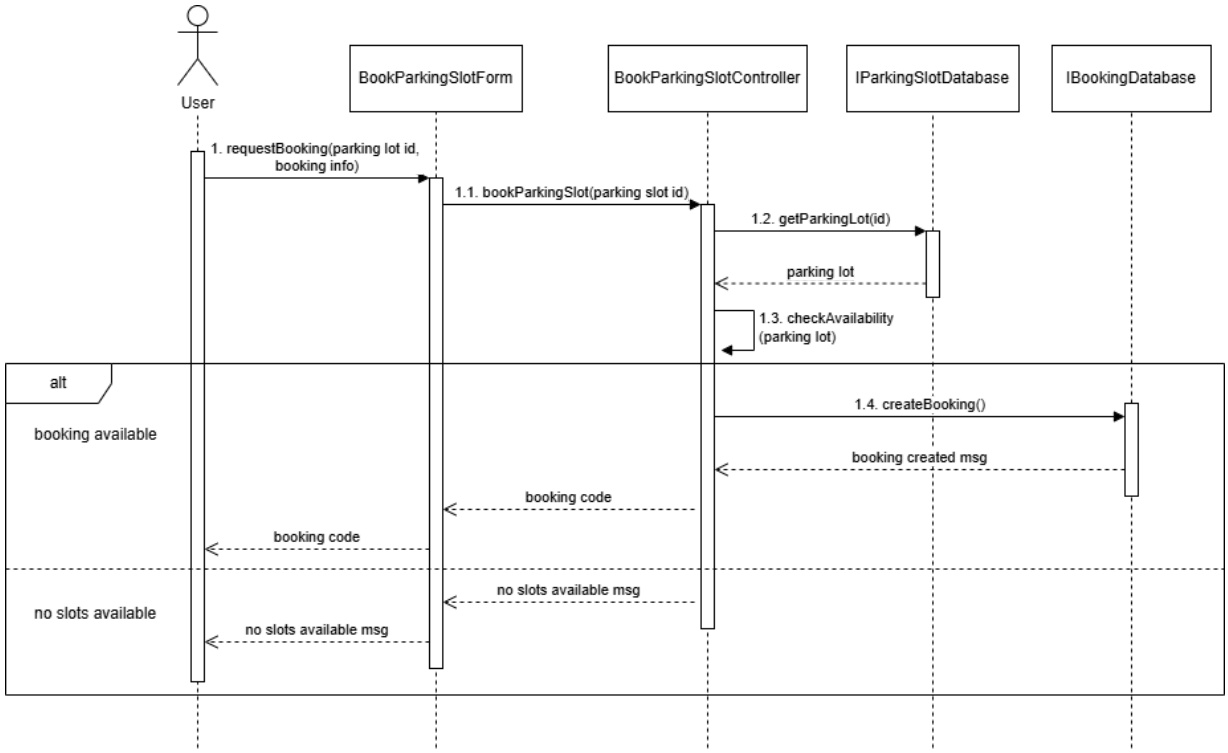

Figure 3-10. Design sequence diagram for the Book Parking Slot *use case*

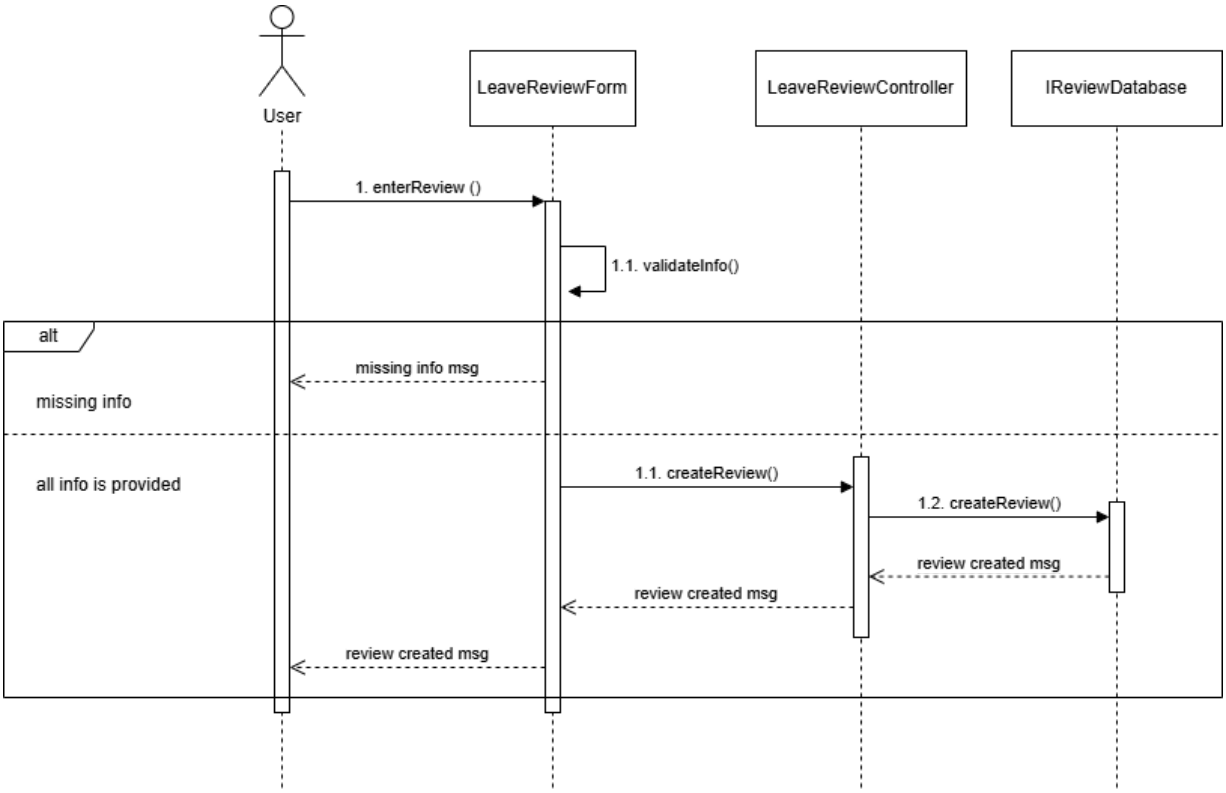

Figure 3-11. Design sequence diagram for the Leave Review *use case*

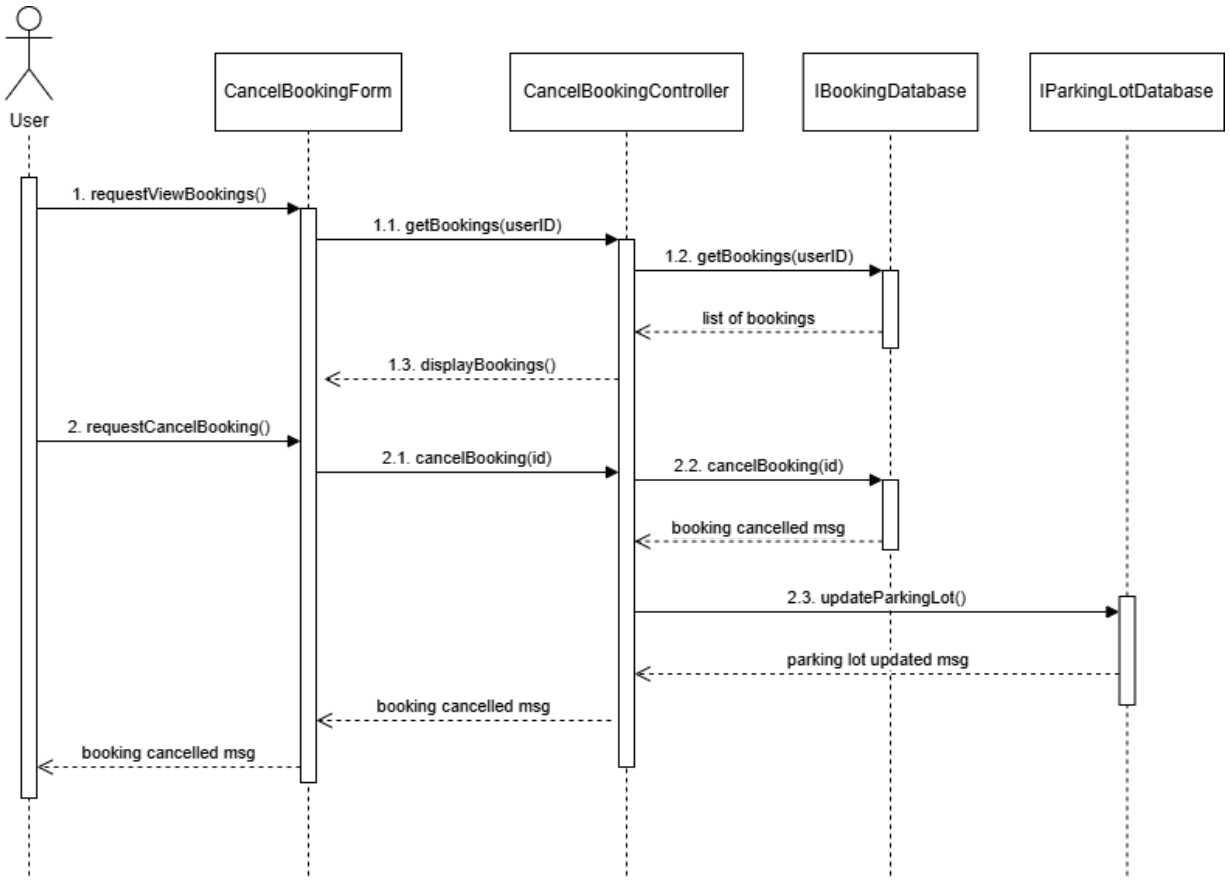

Figure 3-12. Design sequence diagram for the Cancel Booking *use case*

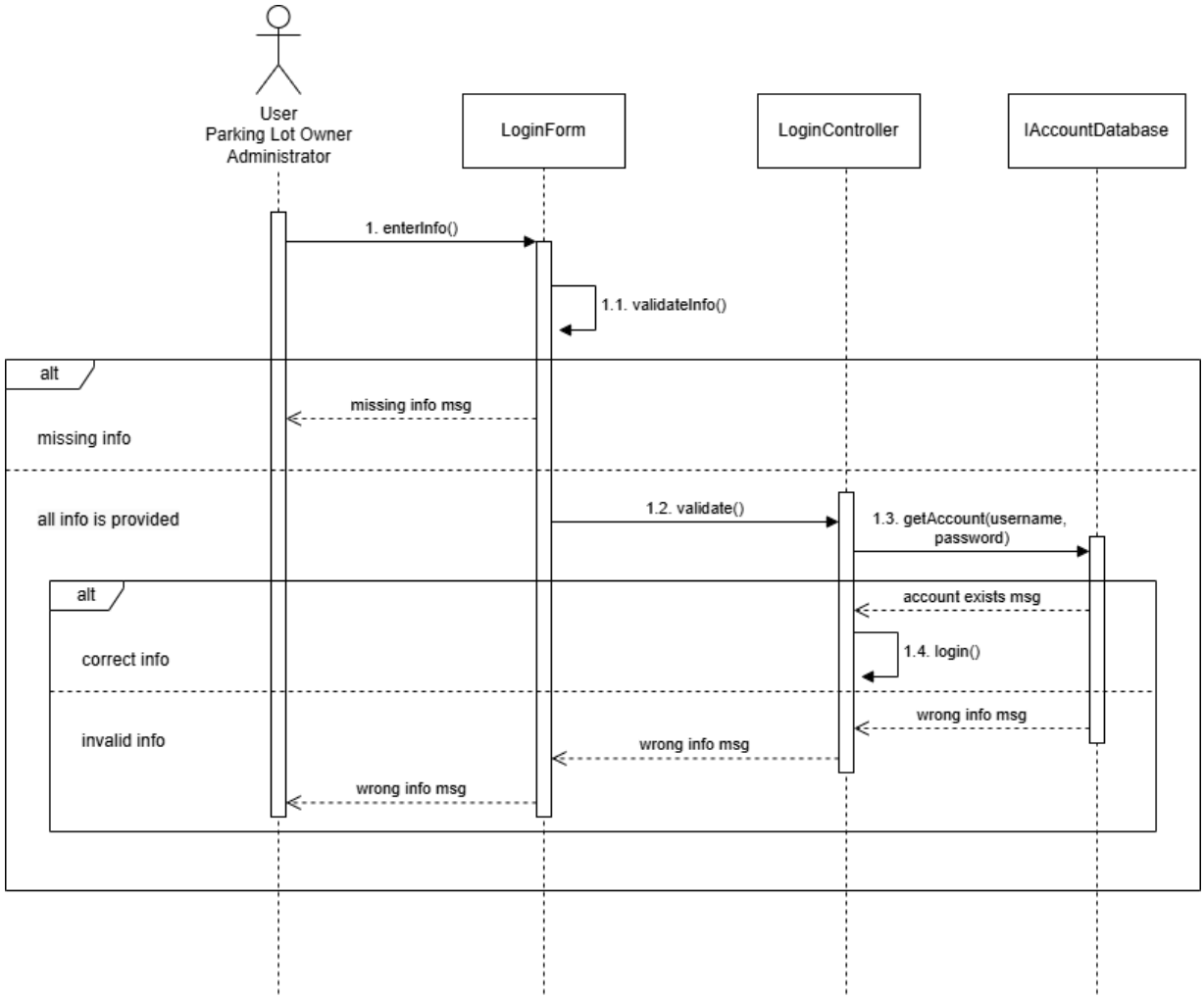

Figure 3-13. Design sequence diagram for the Login *use case*

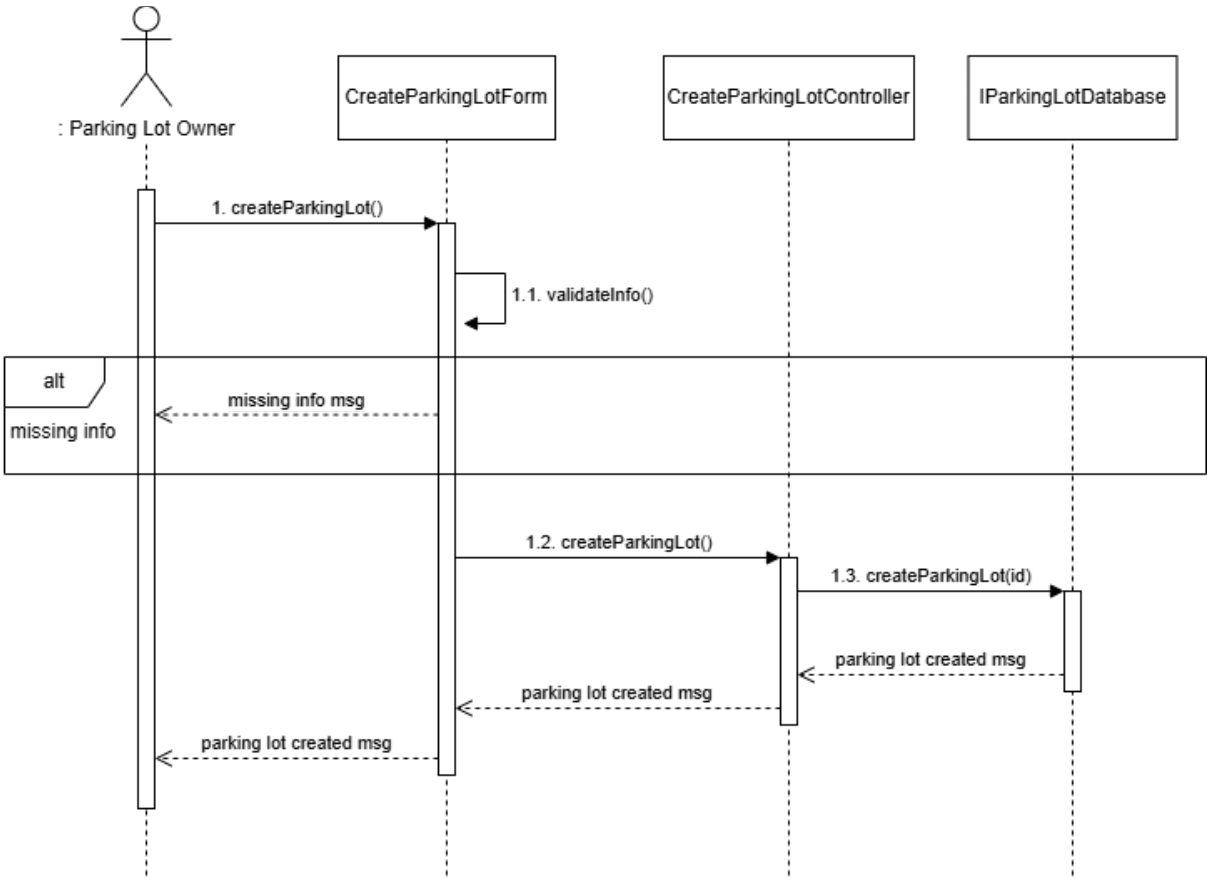

Figure 3-14. Design sequence diagram for the Create Parking Lot *use case*

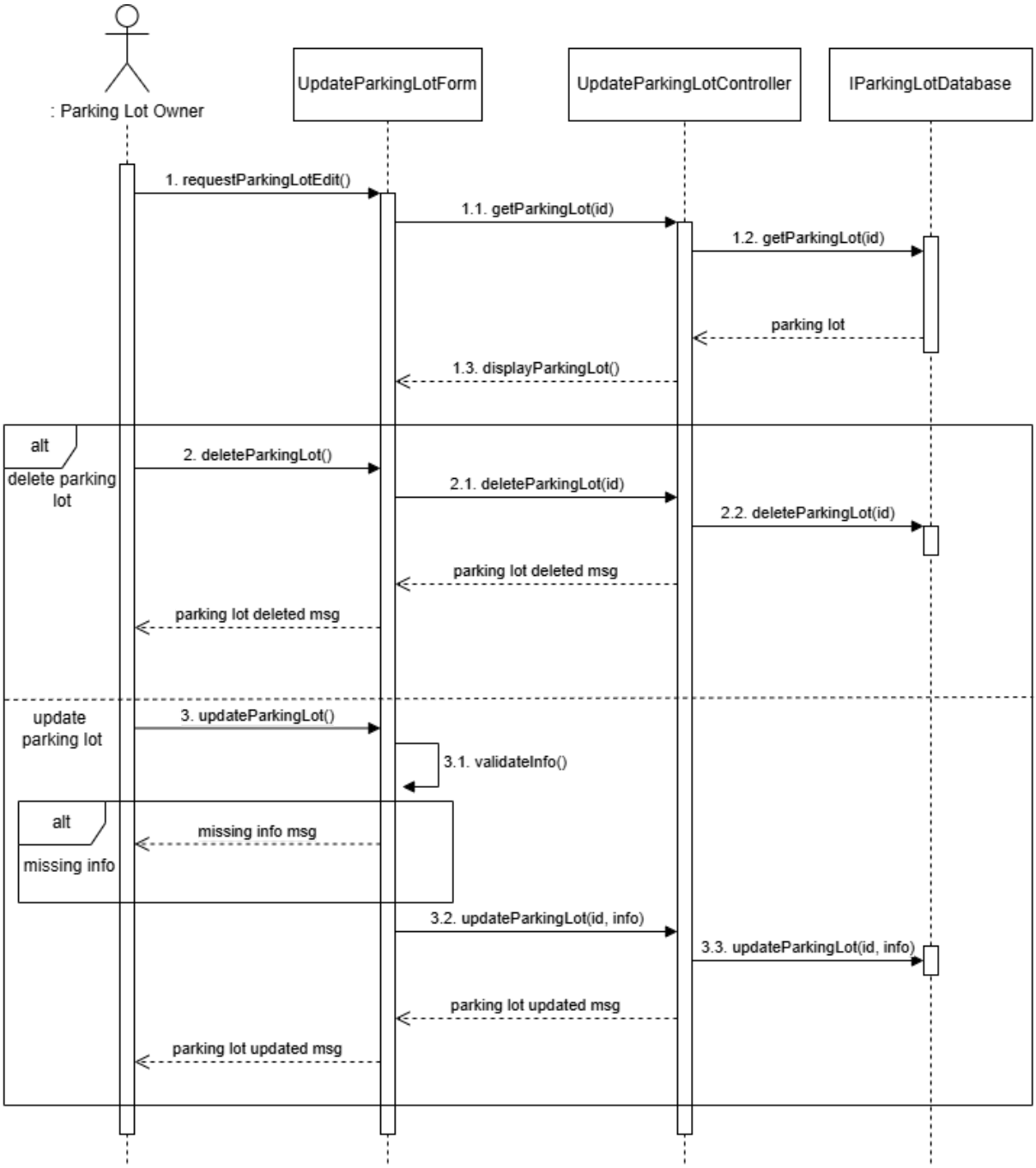

Figure 3-15. Design sequence diagram for the Update Parking Lot *use case*

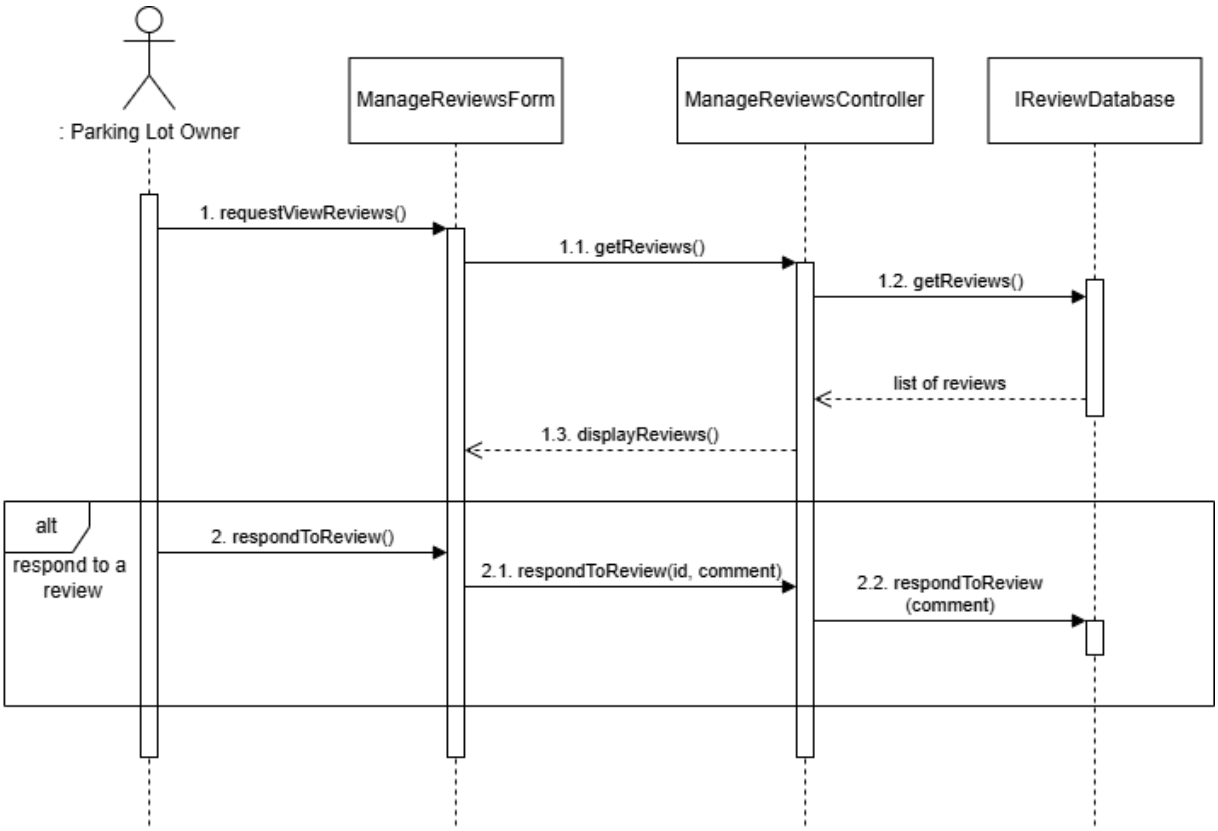

Figure 3-16. Design sequence diagram for the Manage Reviews *use case*

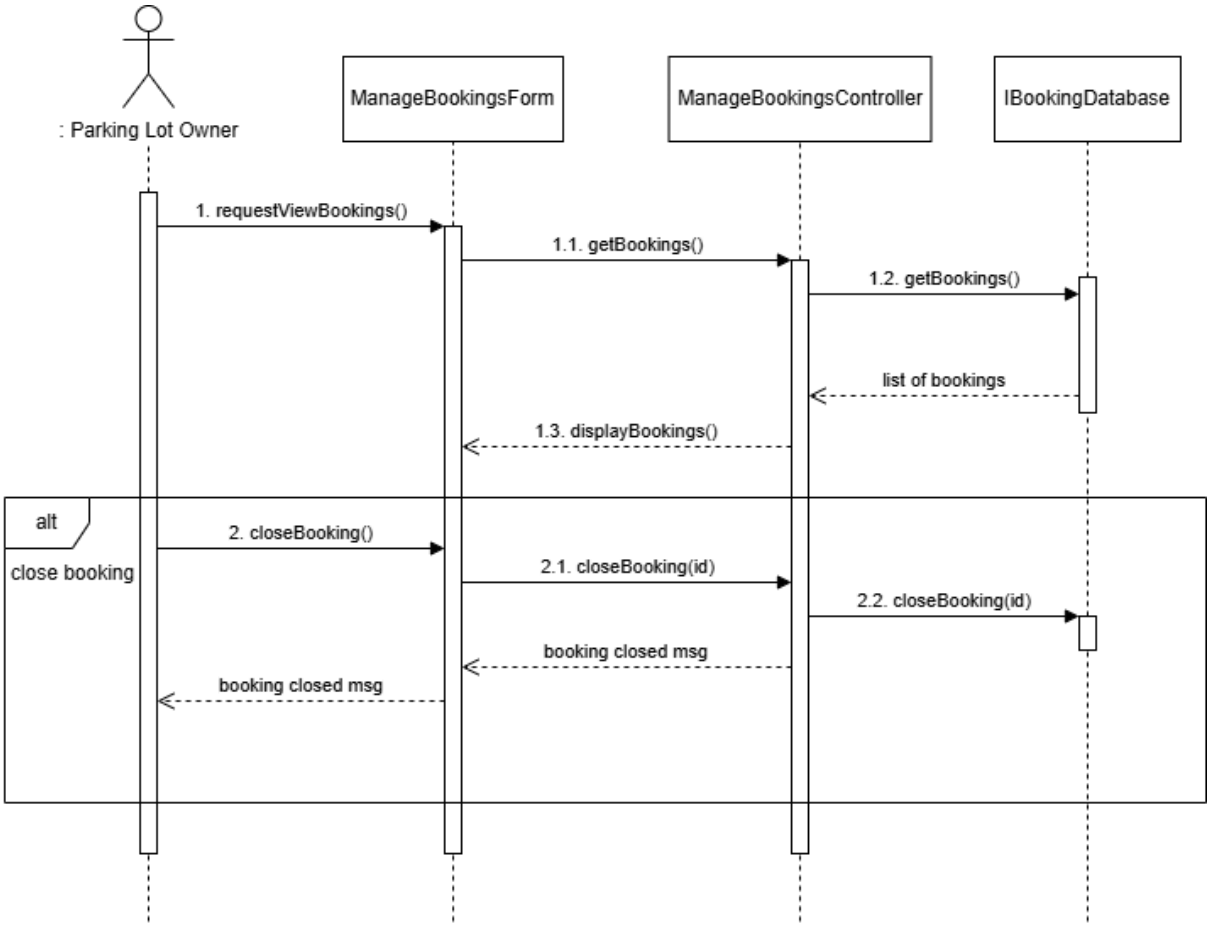

Figure 3-17. Design sequence diagram for the Manage Bookings *use case*

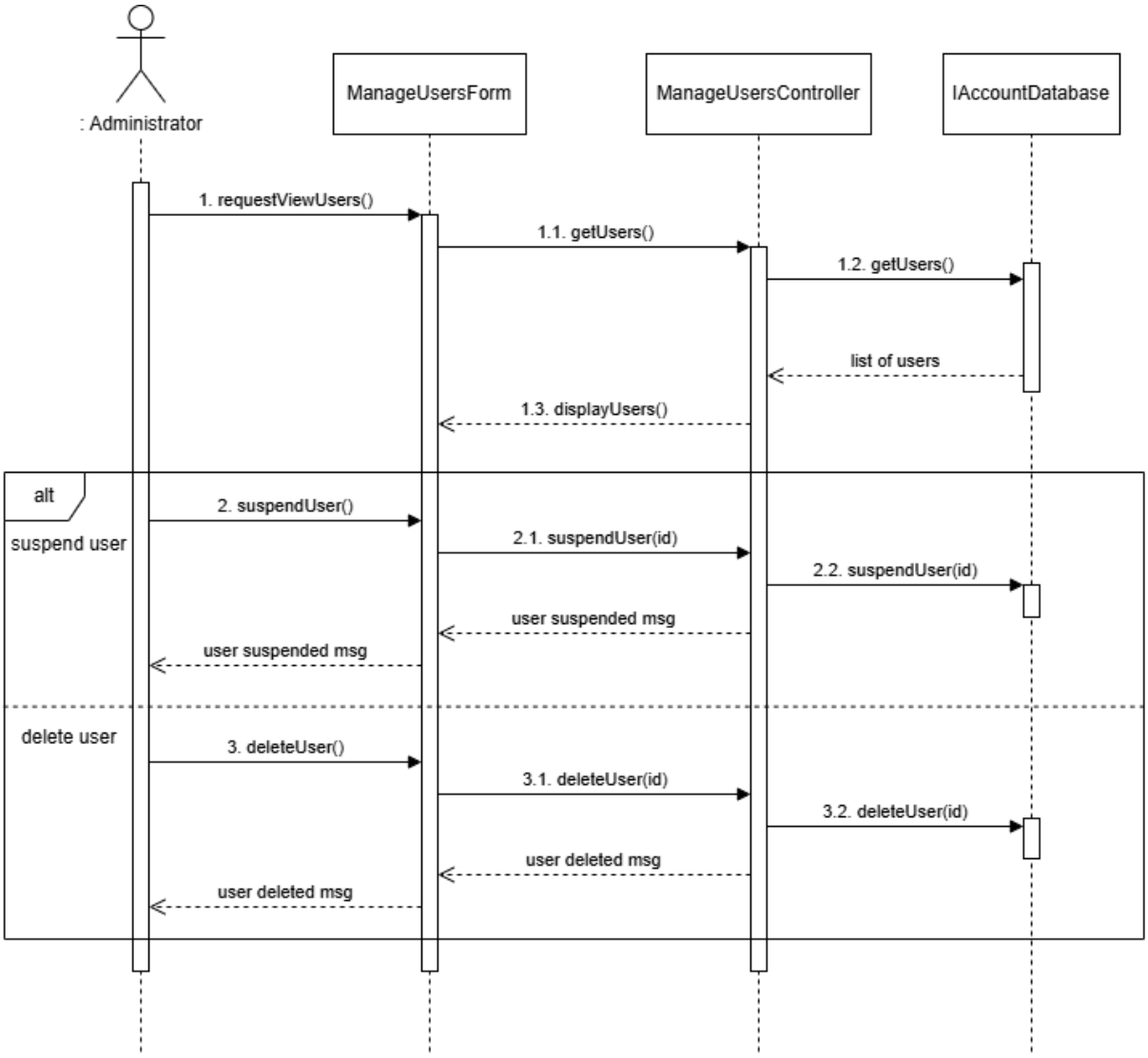

Figure 3-18. Design sequence diagram for the Manage Users *use case*

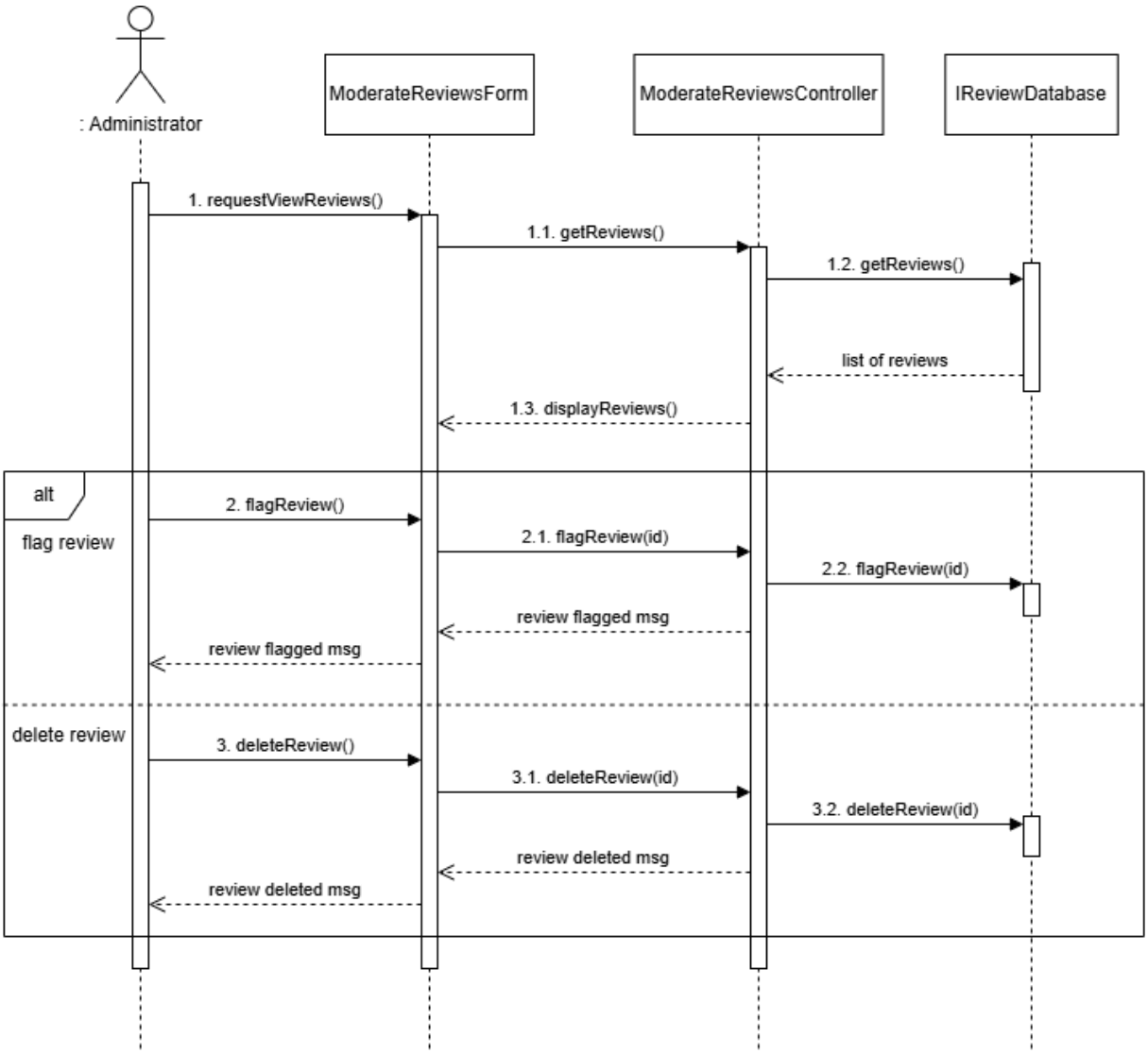

Figure 3-19. Design sequence diagram for the Moderate Reviews *use case*

3.4.2. Design views of participating classes

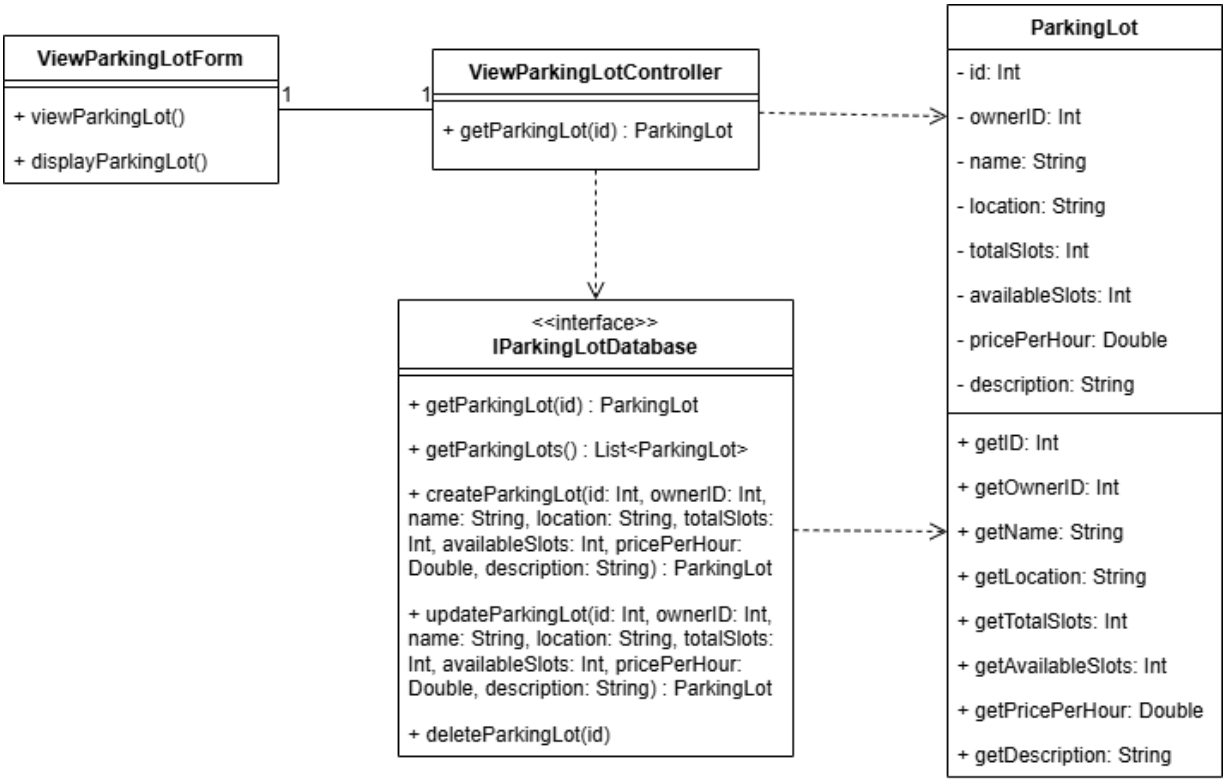

Figure 3-20. Design VOPC for the View Parking Lot Information *use case*

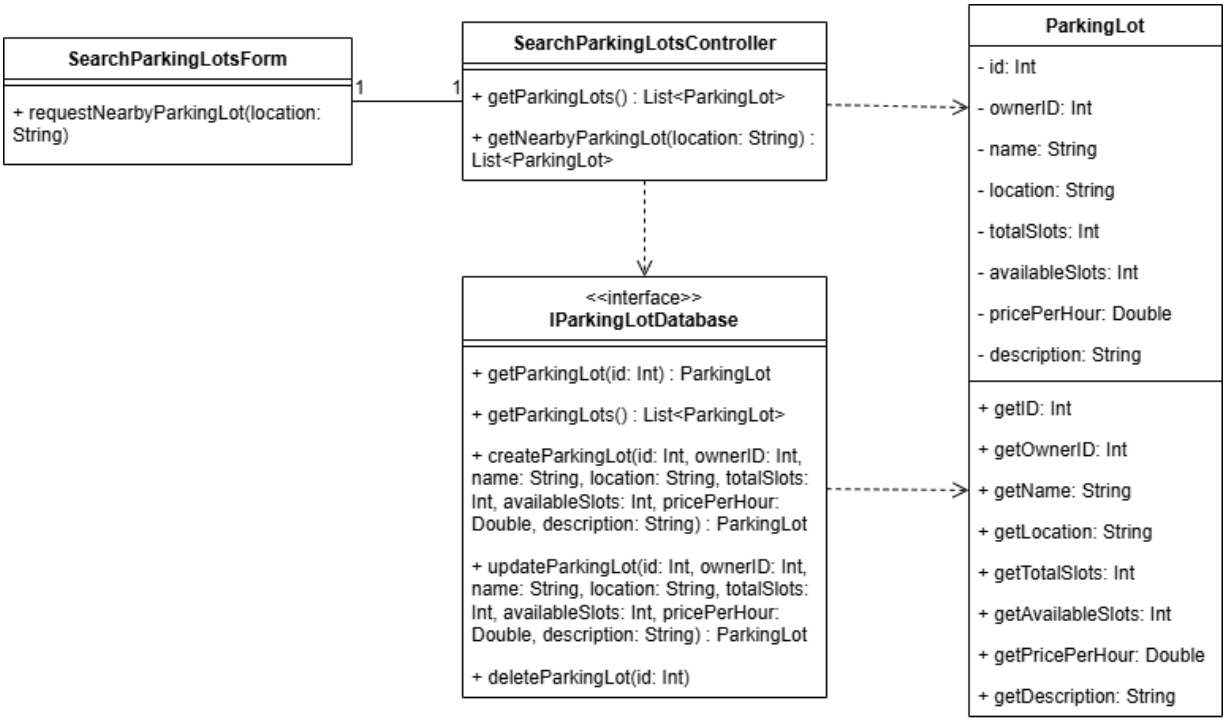

Figure 3-21. Design VOPC for the Search Parking Lots *use case*

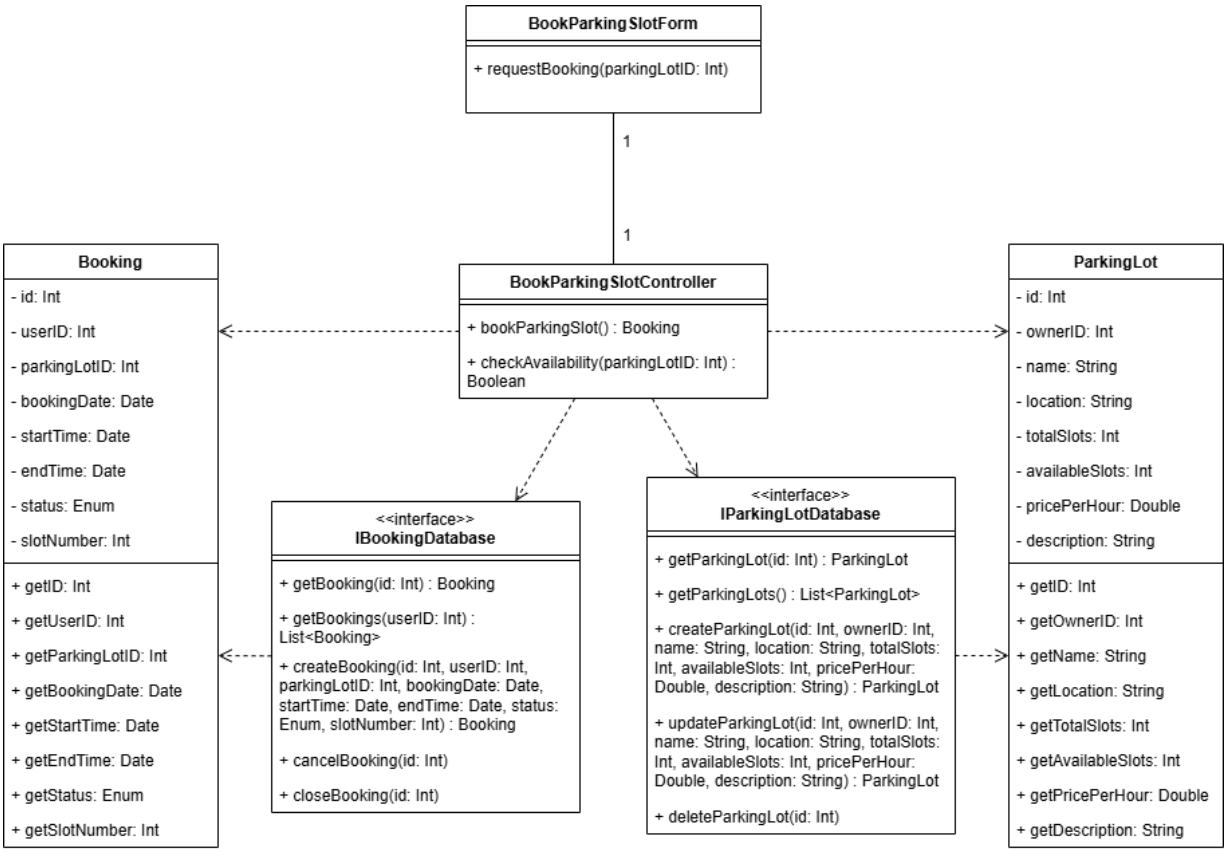

Figure 3-22. Design VOPC for the Book Parking Slot *use case*

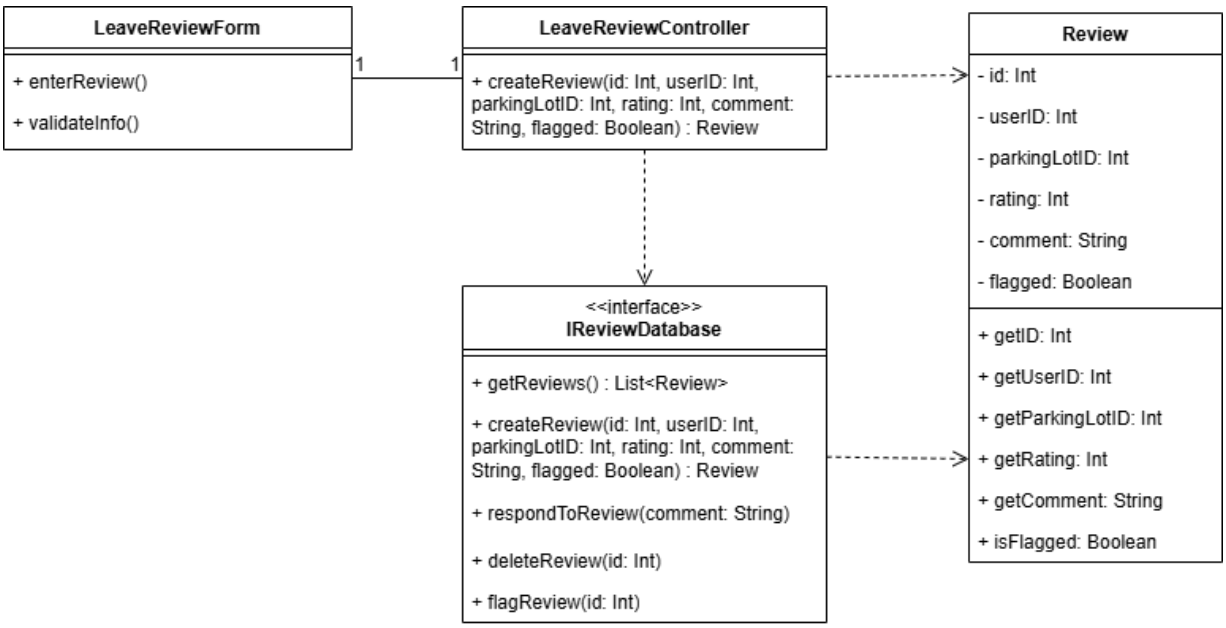

Figure 3-23. Design VOPC for the Leave Review *use case*

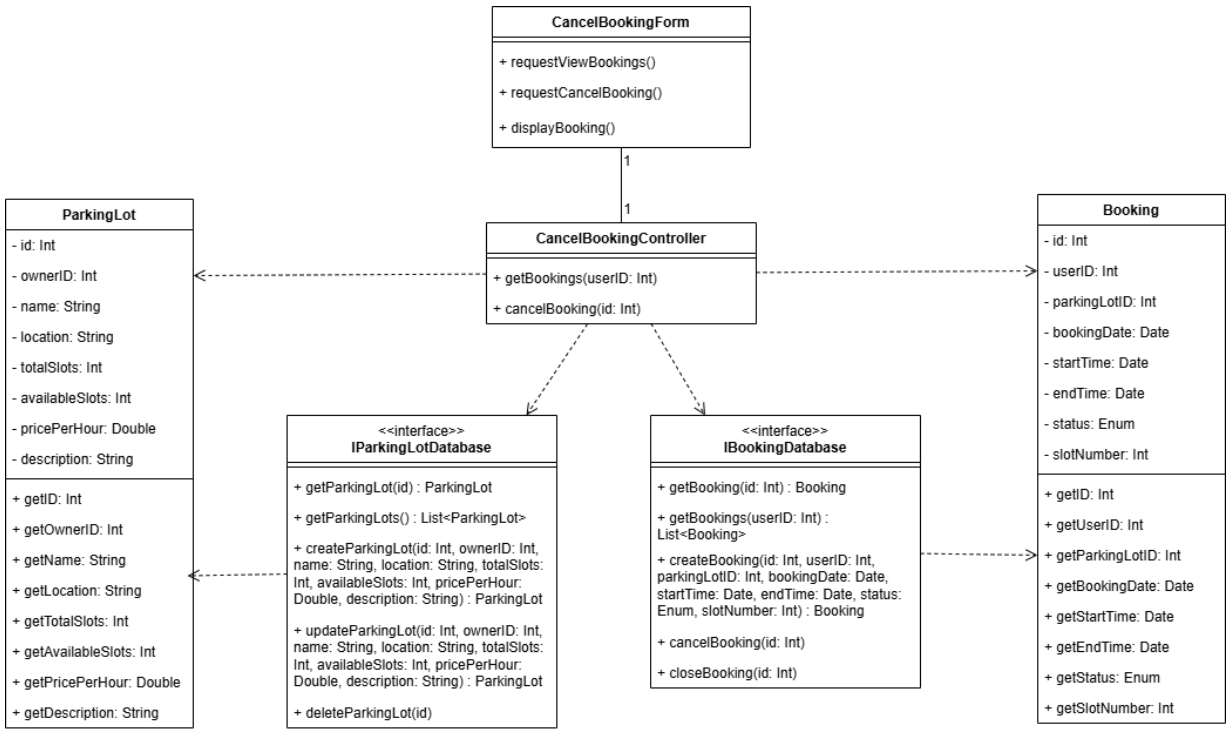

Figure 3-24. Design VOPC for the Cancel Booking *use case*

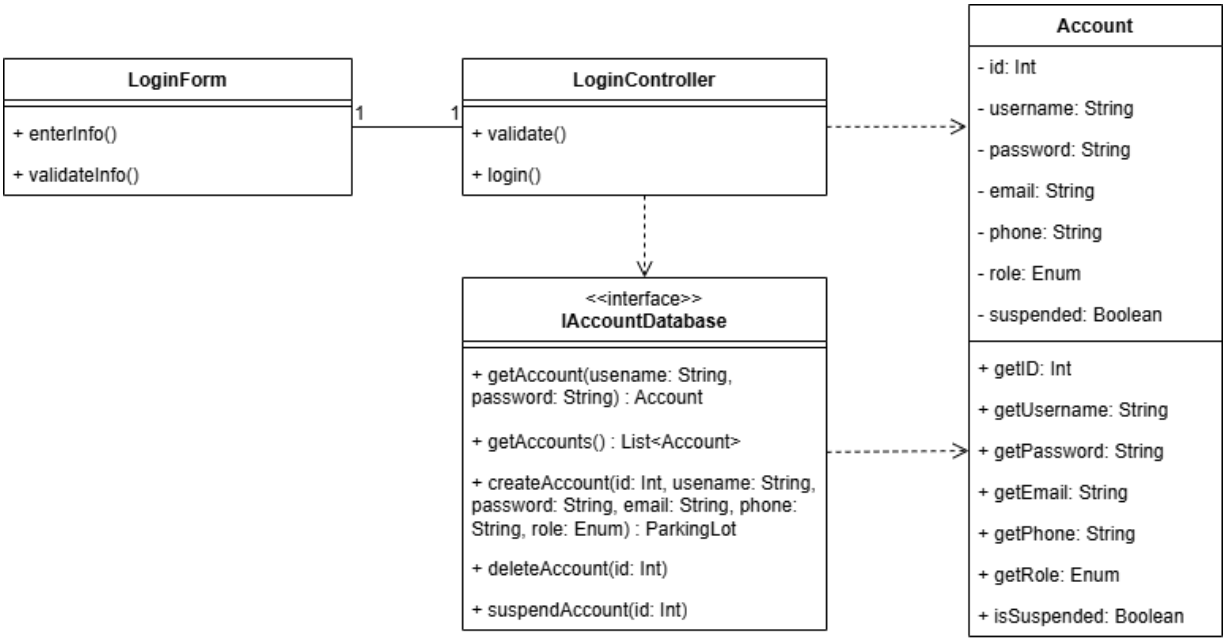

Figure 3-25. Design VOPC for the Login *use case*

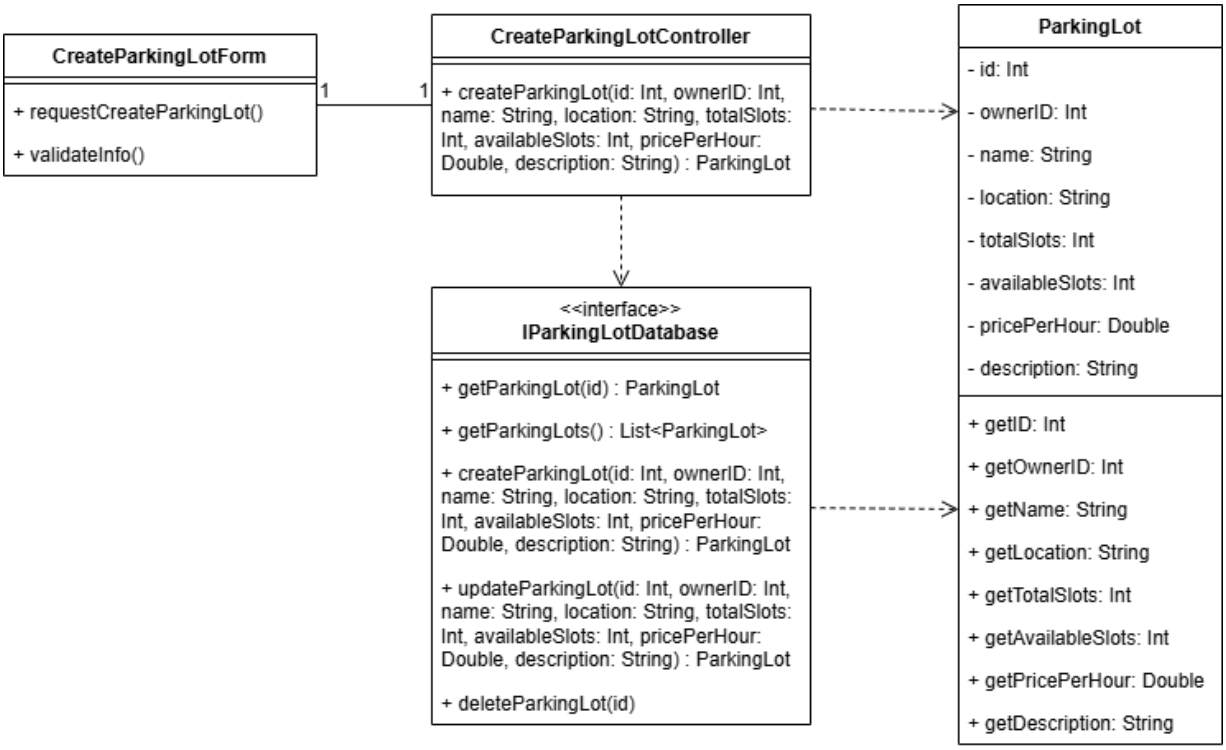

Figure 3-26. Design VOPC for the Create Parking Lot *use case*

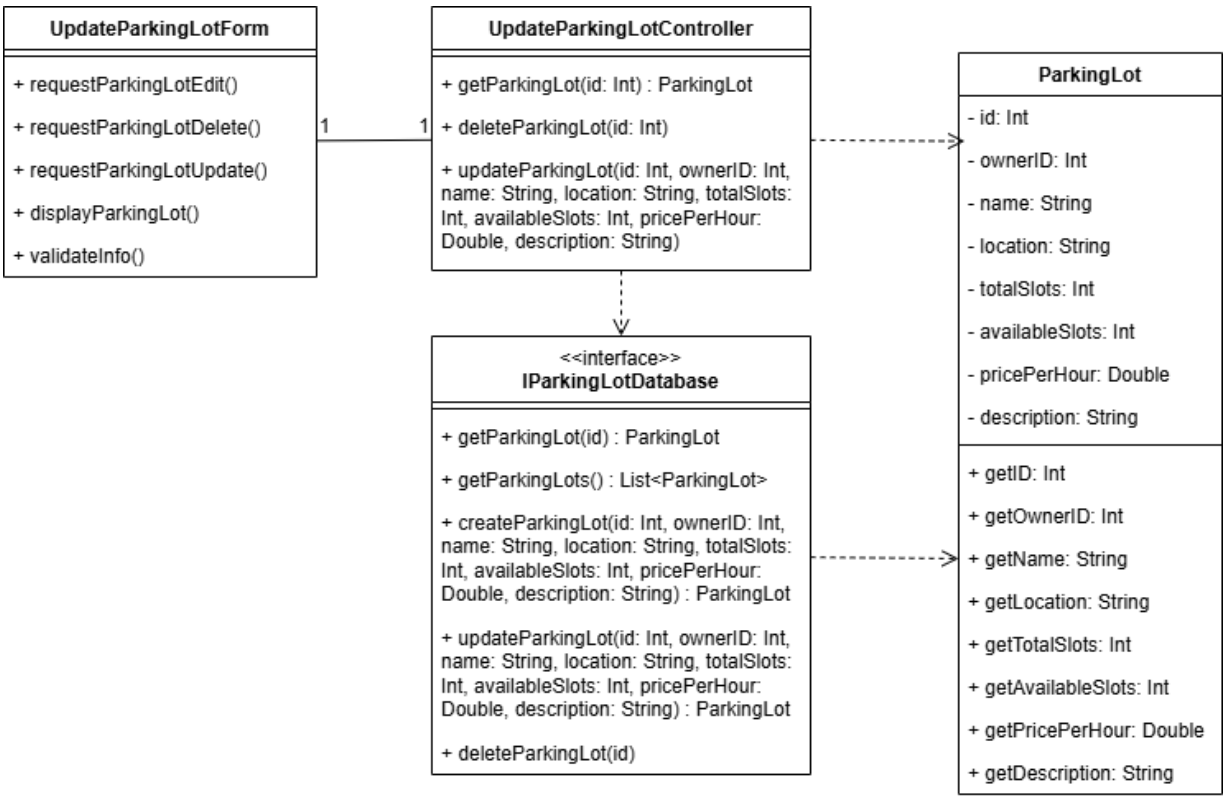

Figure 3-27. Design VOPC for the Update Parking Lot *use case*

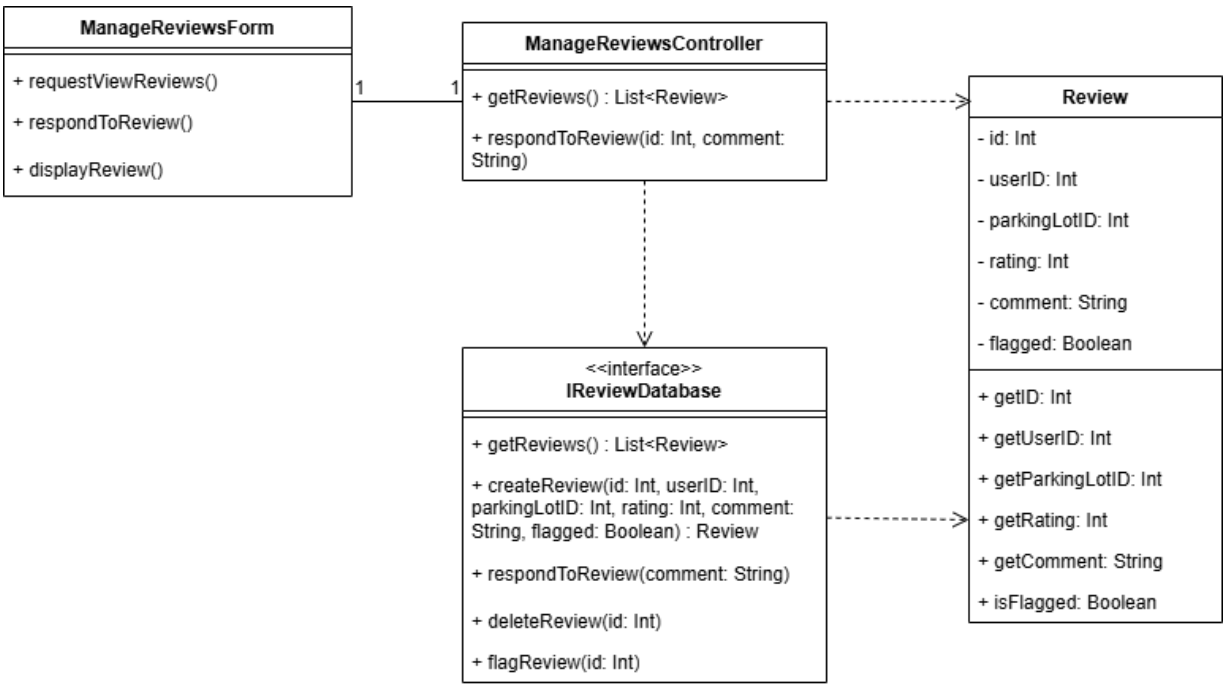

Figure 3-28. Design VOPC for the Manage Reviews *use case*

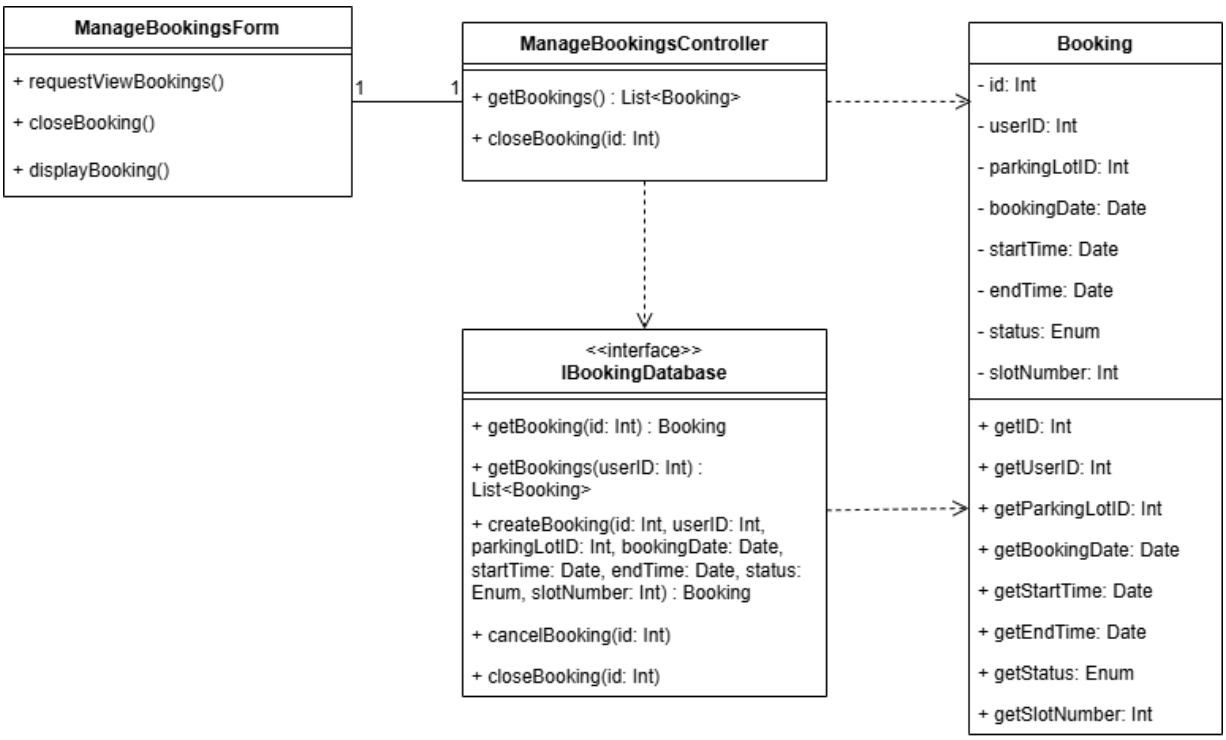

Figure 3-29. Design VOPC for the Manage Bookings *use case*

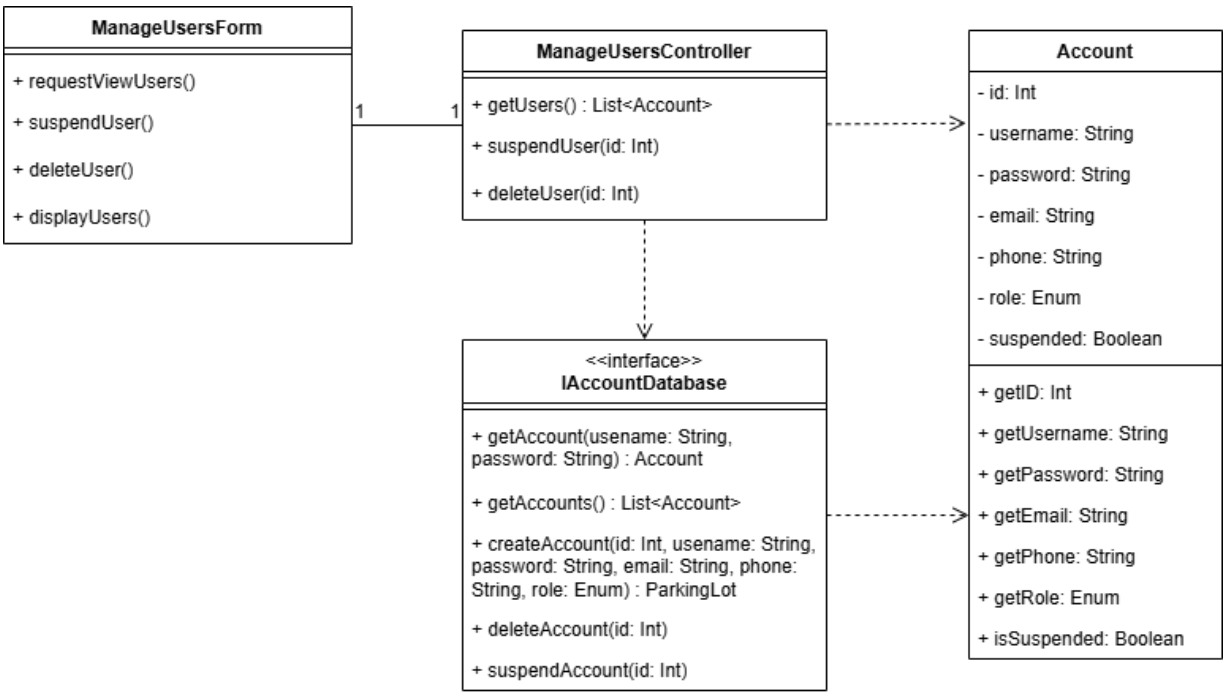

Figure 3-30. Design VOPC for the Manage Users *use case*

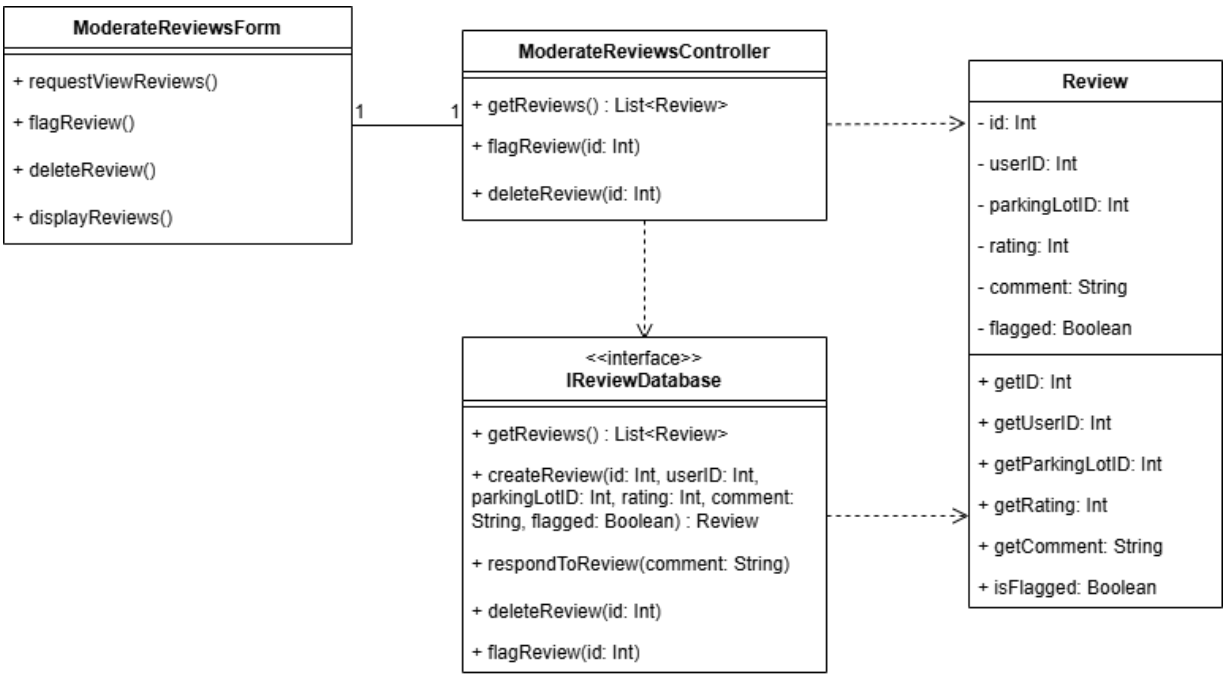

Figure 3-31. Design VOPC for the Moderate Reviews *use case*

3.5. Subsystem design

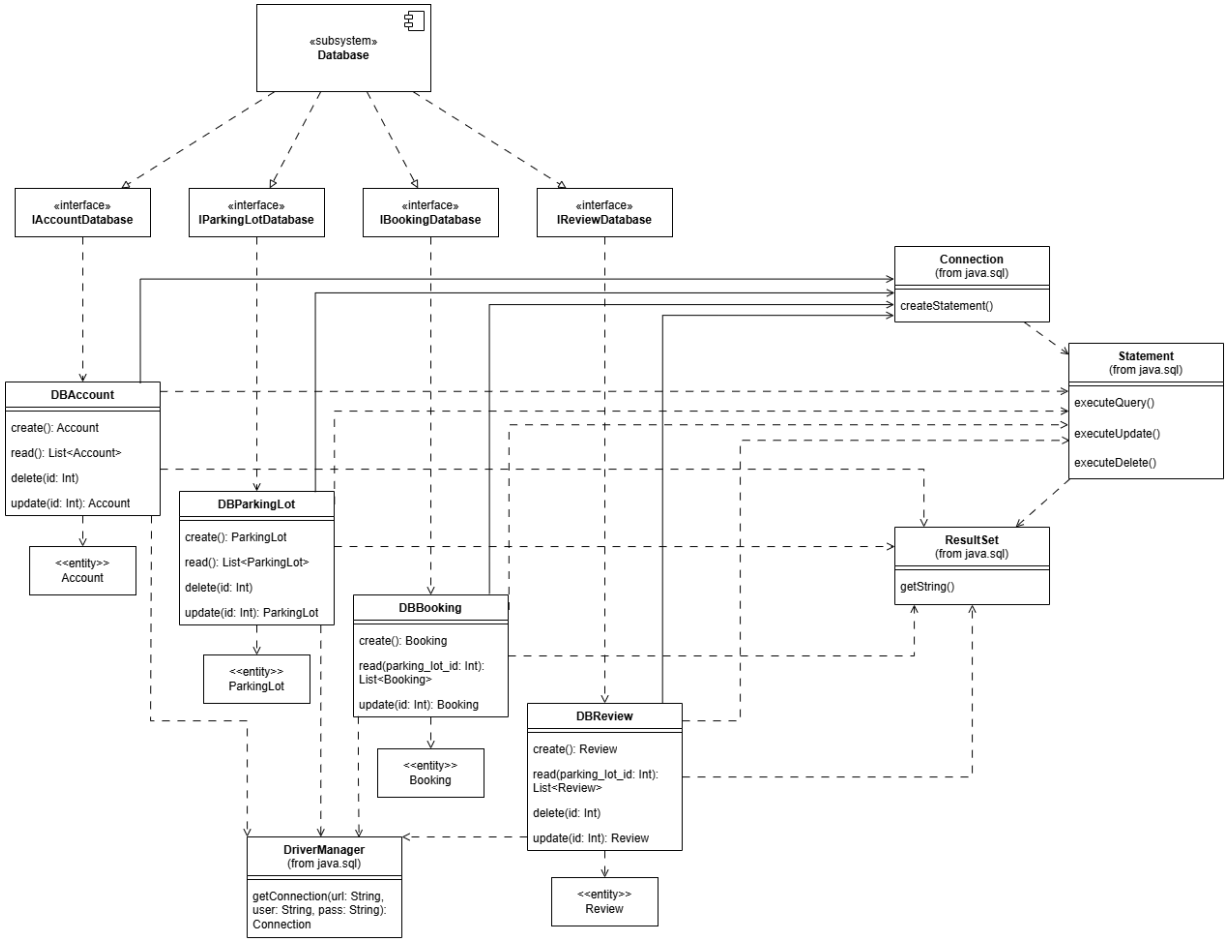

Figure 3-32. Database *subsystem diagram*

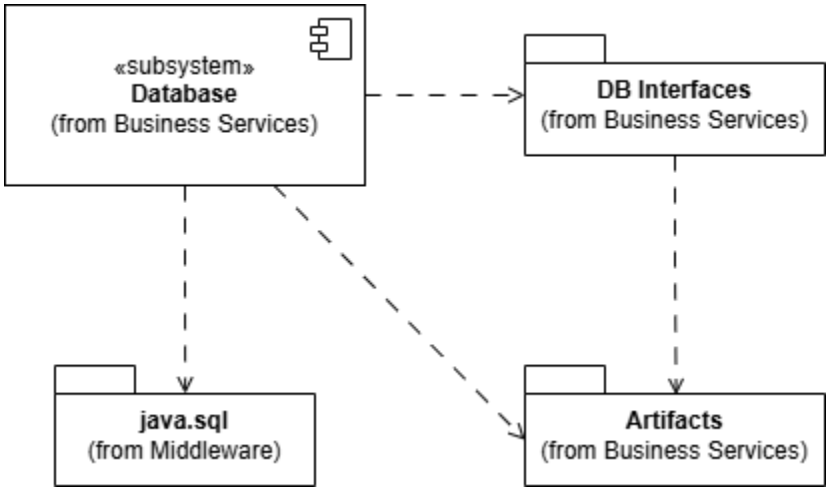

Figure 3-33. Subsystem dependencies class diagram

3.6. Database design

MySQL EER diagram for the relational database.

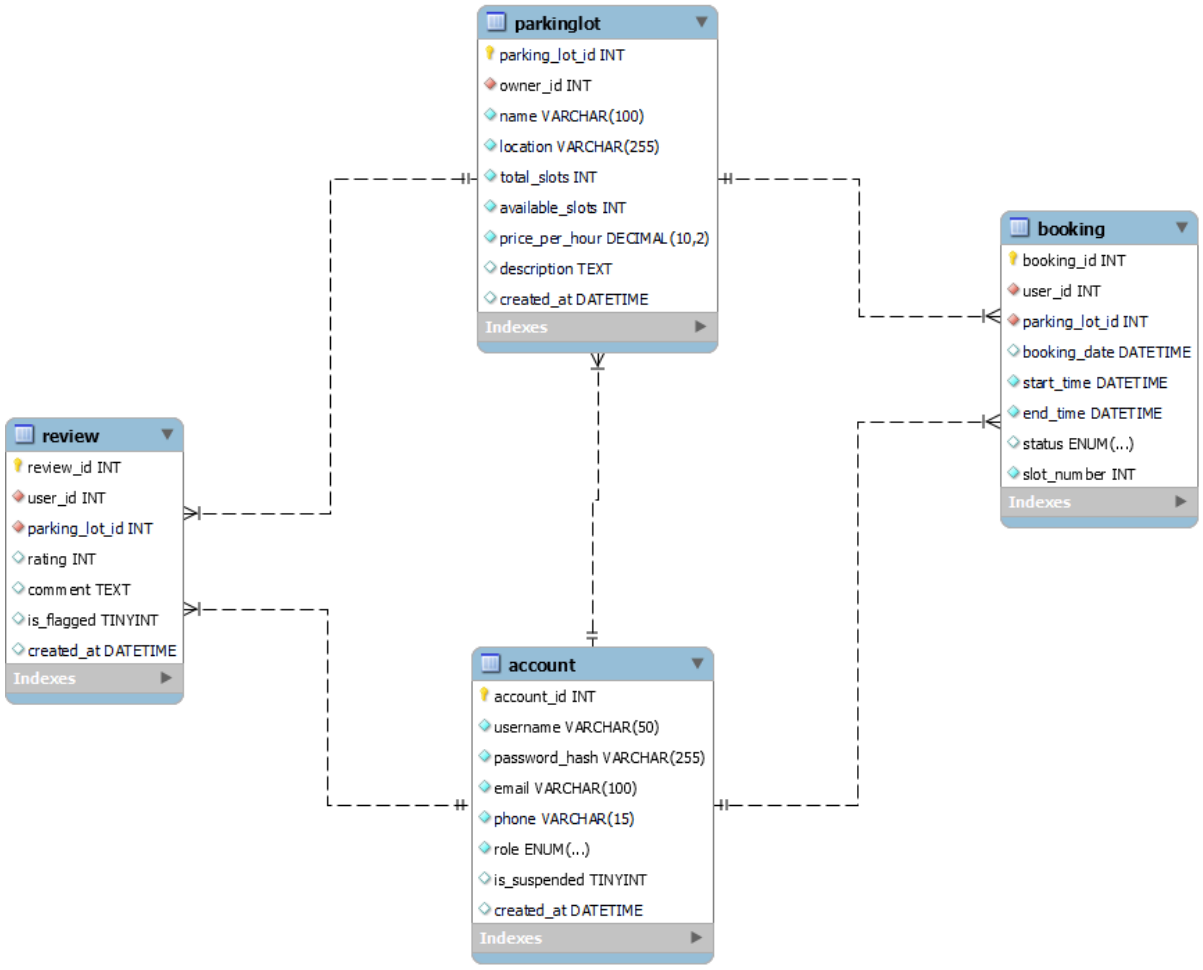

Figure 3-34. The relational data model