Q w i k T i x

E v e n t B o o k i n g S y s t e m

S a t y a P r a n e e t h R e d d i

7 0 0 7 4 7 4 5 1

C I S 5 6 9 0

A d v a n c e d S y s t e m s P r o j e c t .

Table of content:

|  |  |  |
| --- | --- | --- |
| Index | Header | Page No |
| 1 | Cover Page | 1 |
| 2 | Table of content | 2 |
| 3 | System description | 3 |
| 4 | Technologies used | 3 |
| 5 | Use cases specification | 4 |
| 6 | Class diagram | 24 |
| 7 | ER Diagram | 25 |
| 8 | Sequence diagrams | 26 |
| 9 | Package Diagram | 30 |
| 10 | Architecture Diagram | 31 |
| 11 | Dependencies Used in Project | 32 |

System Description:

The goal of this Application ‘QwikTix’, an Event Booking System is, it allows users to efficiently browse, select, view, book tickets for a wide range of events. The system will provide a user-friendly interface to view event details, reserve tickets for the events and receive confirmation via email for the booked event. Additionally it also has an wishlist functionality, where users can wishlist any number of events for future reservation purposes.

The other side of the application is where an admin can add and manage events, so that users can view the events added/modified by admin, admin can also add and manage reservations.

User can only access system when the customer is a registered QwikTix customer, so User can register with QwikTix and then login, additionally when a user forget password, user can change to a new password by getting an otp to users mail.

Technology/Tools that are used to implement QwikTix:

Tech Stack: Java, SpringBoot, REST Api, SpringMVC, SpringDataJpa(Hibernate), Spring Security, MySQL, AWS S3, Java Mail Sender(Gmail), HTML5, CSS3, Thymeleaf, Bootstrap, AdminLTE3.

a. Backend Framework: Spring Boot is used as a backend framework. It is simple, efficient and provides a robust foundation for developing Restful end points using Java. It makes application setup easy. Spring Security is used to implement login functionality.

b. Database: Spring Data JPA is used for data persistence, allowing integration with a relational database management system MySQL.

c. Front-End Development: HTML5, CSS3, Thymeleaf, Bootstrap, AdminLTE3.

Use Cases:

QT01: Register User

|  |
| --- |
| **Use Case:** Register User |
| ID: QT01 |
| Brief Description:   * Allows a user to create a new user account in the system. |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. The user is not already logged in to the system. 2. User is in Login Page. |
| Main flow/Steps:   1. User launches the website on a browser. 2. User navigates to qwiktix url and login page is displayed. 3. User selects Register new Account 4. User enters valid data and submits form |
| Postconditions:   * User account is created and is taken to Login Page with success message. |
| Alternative flows:   1. Details entered by User are wrong, System displays error. 2. User enters age less than 18, system display error. |

QT02: Login User:

|  |
| --- |
| **Use Case:** Login User |
| ID: QT02 |
| Brief Description:   * Allows a user to log in to their account in the system. |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. User accesses the login page. |
| Main flow/Steps:   1. User enters valid data 2. User clicks on Login button |
| Postconditions:   * User is authenticated and gets access to application and is in User dashboard. |
| Alternative flows:   1. Details entered by User are invalid.    1. System displays error saying invalid login details. |

QT03: Add Event

|  |
| --- |
| **Use Case:** Add Event |
| ID: QT03 |
| Brief Description:   * Allows admin to add a new event to the system. |
| Primary Actors: Admin |
| Secondary Actors: |
| Preconditions:   1. Admin is logged in to the system. |
| Main flow/Steps:   1. Admin navigates to Events section in Sidebar 2. Admin clicks on New Event 3. Admin enters all the valid event data. 4. Admin clicks submit and event will be added. |
| Postconditions:   * Event is added to the system and becomes visible in Events page. |
| Alternative flows:   1. Details entered by Admin are invalid, System displays error saying invalid event details. 2. Admin uploads an image larger than max size, System displays error 3. Admin selects past date for an event, system displays error |

QT04: View All Events

|  |
| --- |
| **Use Case:** View All Events |
| ID: QT04 |
| Brief Description:   * Allows user to view all current events in the system. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in. 2. Admin has previously added one or more events |
| Main flow/Steps:   1. User is navigated to Dashboard 2. All events are displayed. |
| Postconditions:   * All events from current date are displayed. |
| Alternative flows: |

QT05: View Event Details

|  |
| --- |
| **Use Case:** View Event Details |
| ID: QT05 |
| Brief Description:   * Allows a user to view detailed information about a specific event. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is in all events/dashboard page 2. Admin must have added one or more events before. |
| Main flow/Steps:   1. User navigates to dashboard 2. User clicks on Book button for a particular event |
| Postconditions:   * User can see the details of selected event. |
| Alternative flows:   1. Admin can see each event details from admin dashboard, by clicking view button |

QT06: Edit Event

|  |
| --- |
| **Use Case:** Edit Event |
| ID: QT06 |
| Brief Description:   * Allows admin to modify the details of an existing event. |
| Primary Actors: Admin |
| Secondary Actors: User |
| Preconditions:   1. Admin is in admin dashboard page |
| Main flow/Steps:   1. Admin selects edit button 2. Admin changes details with valid data 3. Clciks submit |
| Postconditions:   * User can see the updated event details in events page. * Admin can see updated event details in events page |
| Alternative flows:   1. When admin enters invalid event data, error message will be displayed on submitting |

QT07: Delete Event

|  |
| --- |
| **Use Case:** Delete Event |
| ID: QT07 |
| Brief Description:   * Allows admin to remove an event from the system. |
| Primary Actors: Admin |
| Secondary Actors: User |
| Preconditions:   1. Admin is in admin dashboard page 2. There is atleast one current event. |
| Main flow/Steps:   1. Admin selects delete button 2. Admin is asked with another confirmation popup to delete 3. Admin selects confirm 4. Admin is navigate back to Events page. |
| Postconditions:   * Event is permanently deleted and will not be displayed in all events page |
| Alternative flows: |

QT08: Make Reservation

|  |
| --- |
| **Use Case:** Make Reservation |
| ID: QT08 |
| Brief Description:   * Allows user to reserve tickets for an event. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User is in Dashboard 2. User selects book button of an event 3. Event details about the event is displayed 4. User clicks on Book this Event button 5. User enter no of tickets 6. User clicks submit booking |
| Postconditions:   * Reservation is displayed in current reservations * An email will be sent to users mail |
| Alternative flows:   1. Admin can make new reservation, by selectin event and user with no of tickets. |

QT09: View Reservation Details

|  |
| --- |
| **Use Case:** View Reservation Details |
| ID: QT09 |
| Brief Description:   * Allows a user to view the details of their existing reservations. |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User clicks on Reservations section in sidebar 2. User clicks on All bookings 3. User can see list of all current and past reservations 4. User clicks on view icon of any reservation |
| Postconditions:   * User sees reservation info with relevant information such as event name, date, and number of tickets. |
| Alternative flows:   1. Admin can view reservation, from reservations section of admin dashboard. |

QT10: Edit Reservation

|  |
| --- |
| **Use Case:** Edit Reservation |
| ID: QT10 |
| Brief Description:   * Allows user to modify the details of an existing reservation, such as changing the number of tickets. |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User clicks on Reservations section in sidebar 2. User clicks on All bookings 3. User can see list of all current and past reservations 4. User clicks on view icon of any reservation |
| Postconditions:   * Changes made by the user are saved, updating the reservation details accordingly and can be seen in reservations section |
| Alternative flows:   1. Admin can edit reservation, from reservations section of admin dashboard. |

QT11: Cancel Reservation

|  |
| --- |
| **Use Case:** Cancel Reservation |
| ID: QT11 |
| Brief Description:   * Allows a user to cancel an existing reservation |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User clicks on Reservations section in sidebar 2. User selects show all reservations 3. All reservations will be displayed 4. User clicks on delete option of the reservation 5. User confirms to delete reservation |
| Postconditions:   * Cancelled Reservation is not displayed to user in Reservations section. |
| Alternative flows:   1. Admin can cancel reservation, from reservations section of admin dashboard. |

QT12: Filter Events

|  |
| --- |
| **Use Case:** Filter Events |
| ID: QT12 |
| Brief Description:   * Allows a user/admin to filter events, based on key word. |
| Primary Actors: User/Admin |
| Secondary Actors: |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User clicks on search bar in dashboard 2. User enters any key word 3. User enters location name 4. User enter date/month 5. User enters event name 6. User enters Venue 7. User enters any keyword of description 8. User enters category |
| Postconditions:   * Events associate with particular key word will be displayed  1. Events associated only with entered location will be displayed. 2. Events associated only with entered date/month will be displayed. 3. Events associated only with entered name will be displayed. 4. Events associated only with entered venue will be displayed. 5. Events associated only with entered keyword description will be displayed. 6. Events associated only with entered category will be displayed. |
| Alternative flows: |

QT13: View Admin Dashboard

|  |
| --- |
| **Use Case:** View Admin Dashboard |
| ID: QT13 |
| Brief Description:   * Allows an admin to access the administrative dashboard for monitoring system activities and managing events and reservations. |
| Primary Actors: Admin |
| Secondary Actors: |
| Preconditions:   1. Admin has access to system/app url |
| Main flow/Steps:   1. Admin is in Login page 2. Admin enters admin credentials 3. Username: admin, password: 123456 (pre defined admin credentials) 4. Admin clicks login |
| Postconditions:   * Admin dashboard will be displayed |
| Alternative flows: |

QT14: View Wishlist

|  |
| --- |
| **Use Case:** View Wishlist |
| ID: QT14 |
| Brief Description:   * Allows user to view their wishlist of favorite events. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system 2. Admin should have added some events before. |
| Main flow/Steps:   1. User is in dashboard where all events are displayed 2. User clicks on wishlist icon 3. User is navigated to wishlist page. |
| Postconditions:   * User sees a list of events they have saved to their wishlist for future reference. |
| Alternative flows: |

QT15: Add to Wishlist

|  |
| --- |
| **Use Case:** Add to Wishlist |
| ID: QT15 |
| Brief Description:   * Allows a user to add an event to their wishlist |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system 2. Admin should have added some events before. |
| Main flow/Steps:   1. User is in dashboard where all events are displayed 2. User goes to an event where user would like to wishlist 3. User clicks on wishlist icon 4. User is navigated to wishlist page. |
| Postconditions:   * Event is added to the user's wishlist. |
| Alternative flows: |

QT16: Remove from Wishlist

|  |
| --- |
| **Use Case:** Remove from Wishlist |
| ID: QT16 |
| Brief Description:   * Allows user to remove an event from their wishlist. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system 2. Admin should have added some events before. 3. User should have already wishlisted one or more events. |
| Main flow/Steps:   1. User is in dashboard where all events are displayed 2. User clicks on wishlist icon 3. User is navigated to wishlist page. Wishlisted events for the user are displayed 4. User navigates to an event, where user would like to remove from wishlist 5. User clicks on delete button |
| Postconditions:   * Event is removed from the user's wishlist, with a message |
| Alternative flows: |

QT17: View Reserved Tickets Admin

|  |
| --- |
| **Use Case:** View All Reserved Tickets Admin |
| ID: QT17 |
| Brief Description:   * Allows user to remove an event from their wishlist. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. Admin is logged in to the system |
| Main flow/Steps:   1. Admin clicks on Reservations section in sidebar 2. Admin clicks on All bookings 3. Admin can see list of all current and past reservations for all users |
| Postconditions:  Admin sees a list of all tickets reserved |
| Alternative flows: |

QT18: Send Email Notification

|  |
| --- |
| **Use Case:** Send Email Notification |
| ID: QT18 |
| Brief Description:   * Automatically sends email notifications to users for reservation confirmations. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system 2. Admin should have added events into system |
| Main flow/Steps:   1. User is in Dashboard 2. User selects book button of an event 3. Event details about the event is displayed 4. User clicks on Book this Event button 5. User enter no of tickets 6. User clicks submit booking |
| Postconditions:   * Email notification is sent to the user's registered email address containing relevant information. |
| Alternative flows:   1. Admin can make new reservation, by selectin event and user with no of tickets. |

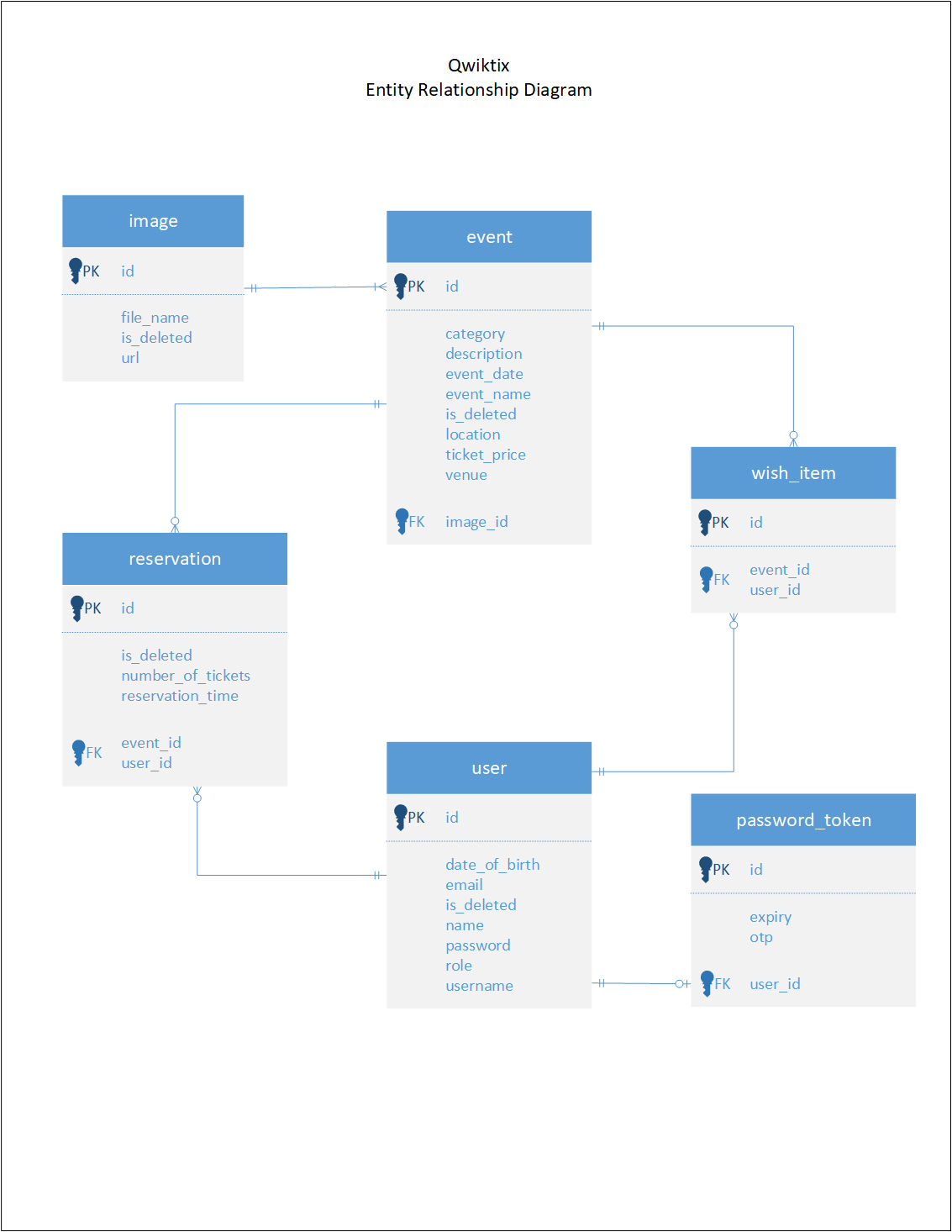
QT19: Contact Us

|  |
| --- |
| **Use Case:** Contact Us |
| ID: QT19 |
| Brief Description:  Allows user to send email to admin via Systems interface regaring any query. |
| Primary Actors: User |
| Secondary Actors: Admin |
| Preconditions:   1. User is logged in to the system |
| Main flow/Steps:   1. User is in Dashboard 2. User clicks on Contact section of Sidebar 3. User enters all details: name, email, subject, message 4. User clicks send message |
| Postconditions:   * Email is sent to admin/application with user submitted data. |
| Alternative flows: |

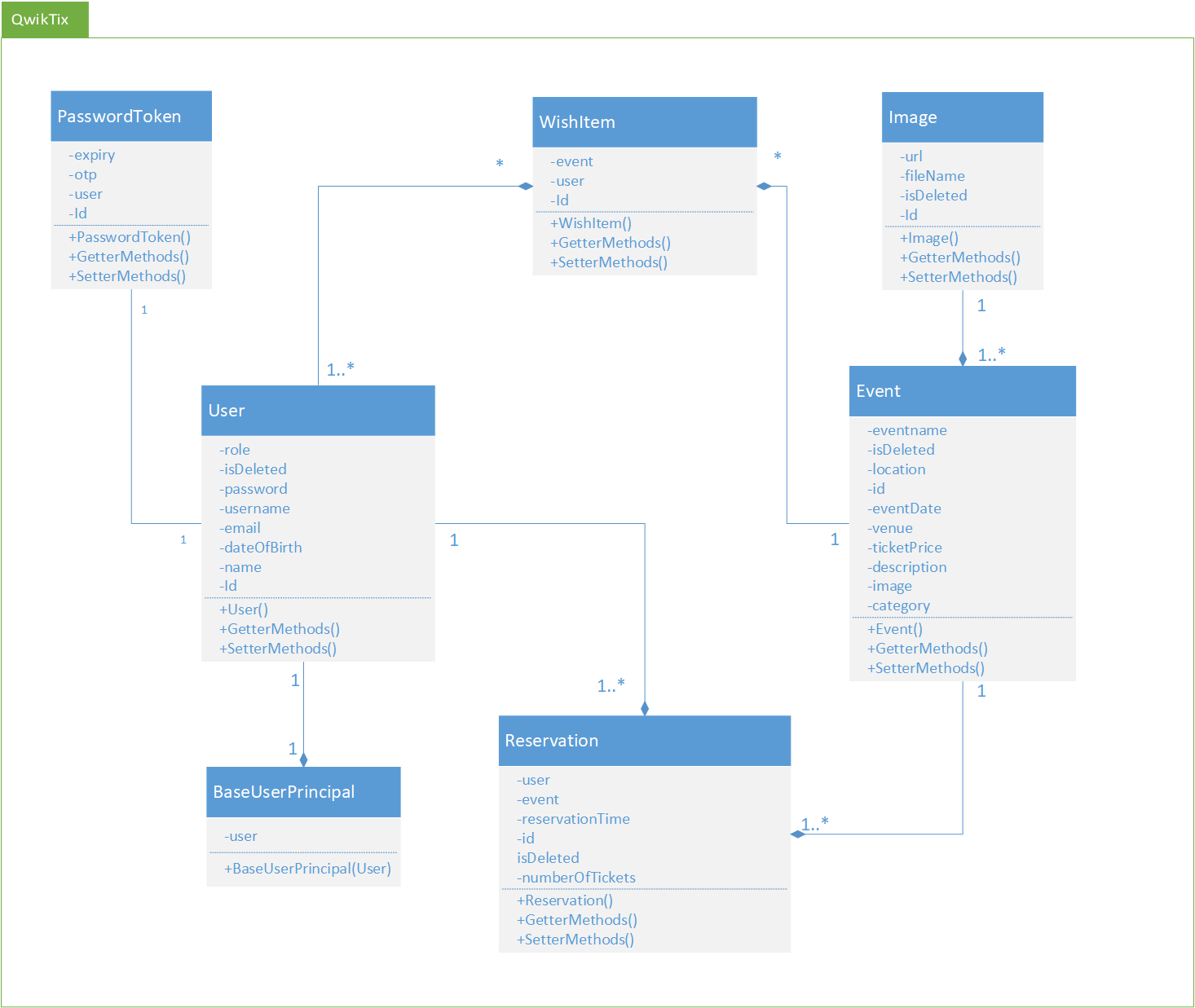
QT20: Forgot Password

|  |
| --- |
| **Use Case:** Forgot Password |
| ID: QT20 |
| Brief Description:  Allows user to set new password |
| Primary Actors: User |
| Secondary Actors: |
| Preconditions:   1. User is in Login page |
| Main flow/Steps:   1. User clicks on I forgot my password 2. User enter valid registered mail id 3. An otp will be sent to mail 4. User enters otp |
| Postconditions:  User will be navigated to Login page and can login with new password. |
| Alternative flows: |

**Entity Relationship Diagram** (Crows Foot Database notation):

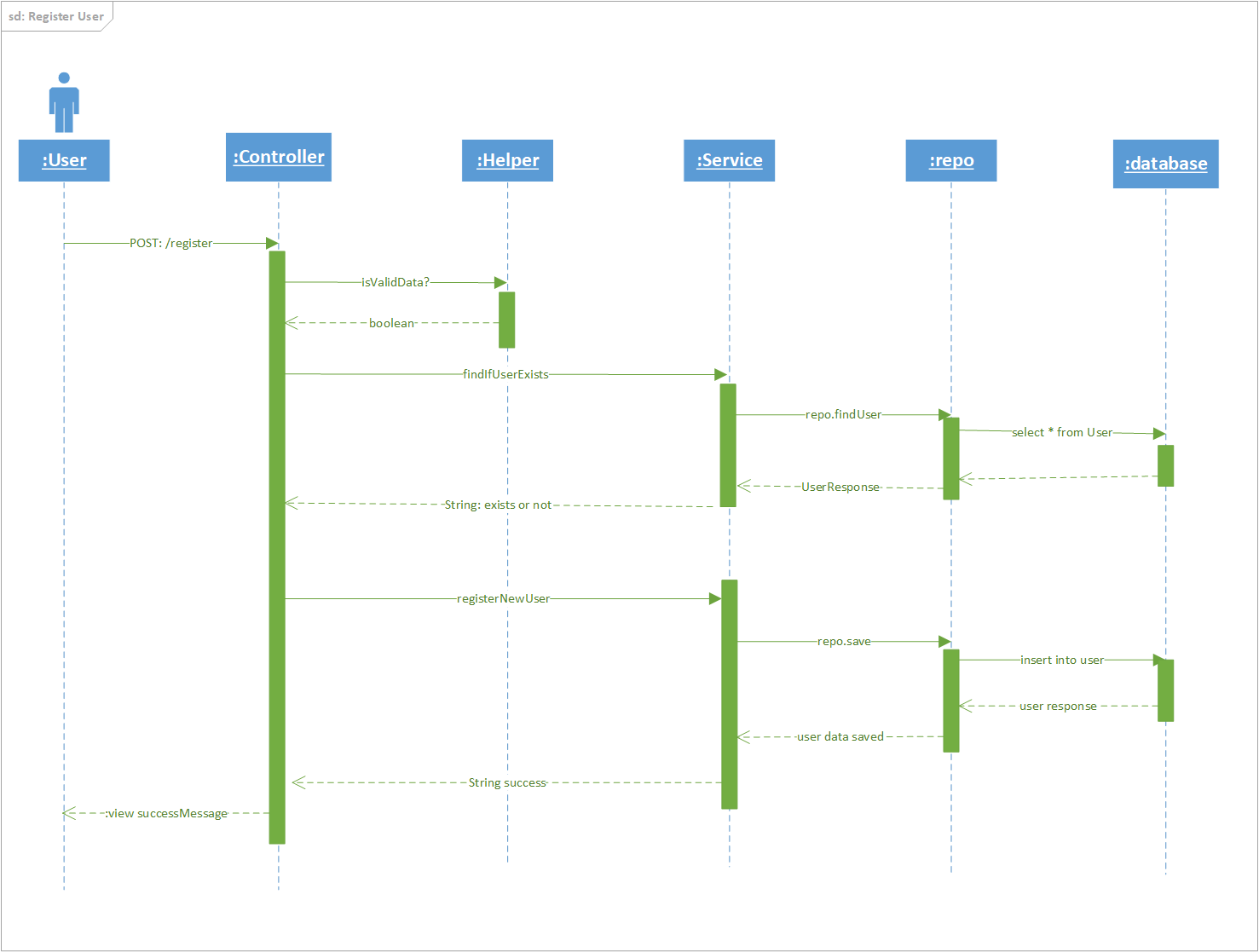


Class Diagram:

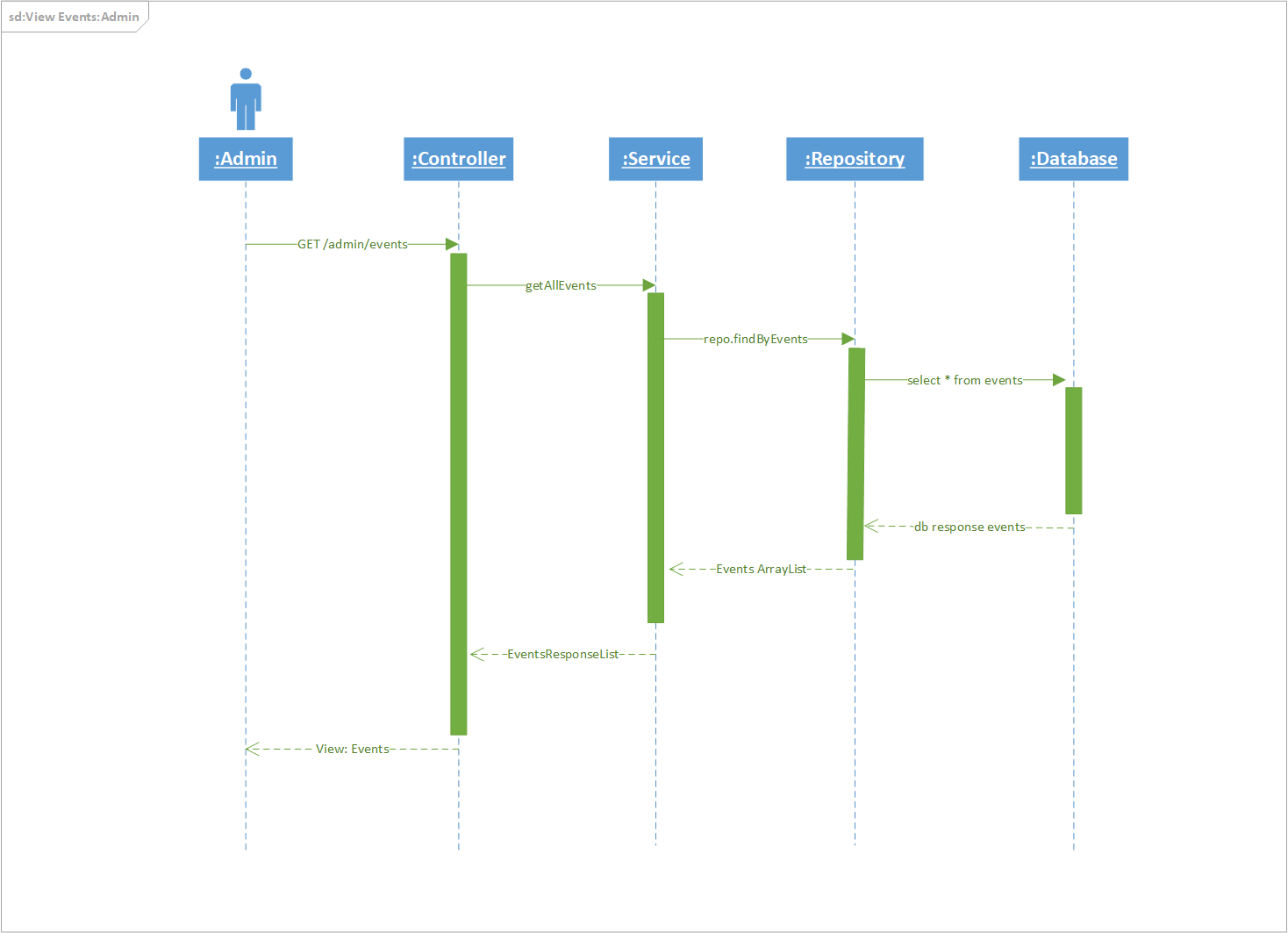
**

Sequence Diagrams:

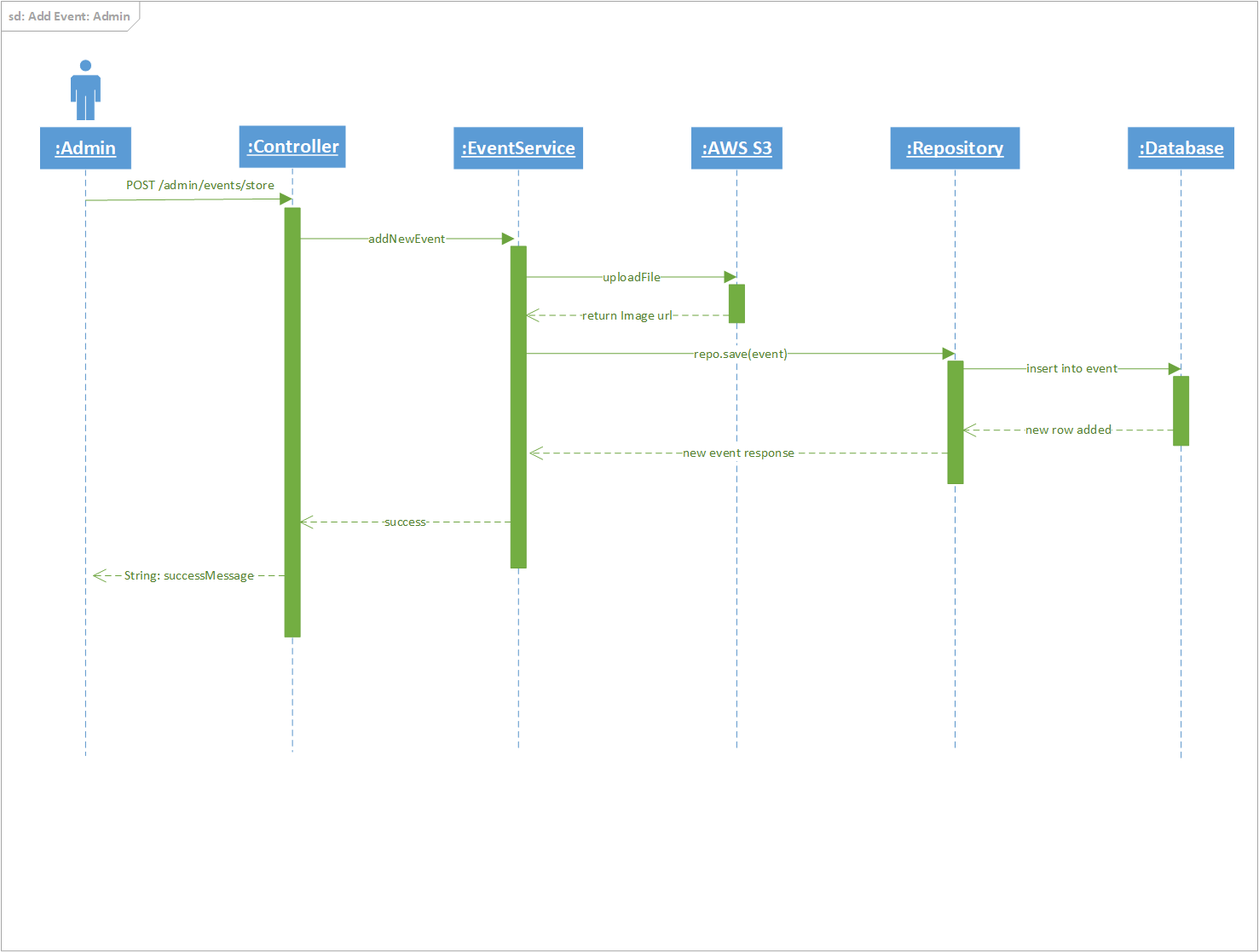
*sd:Register user*



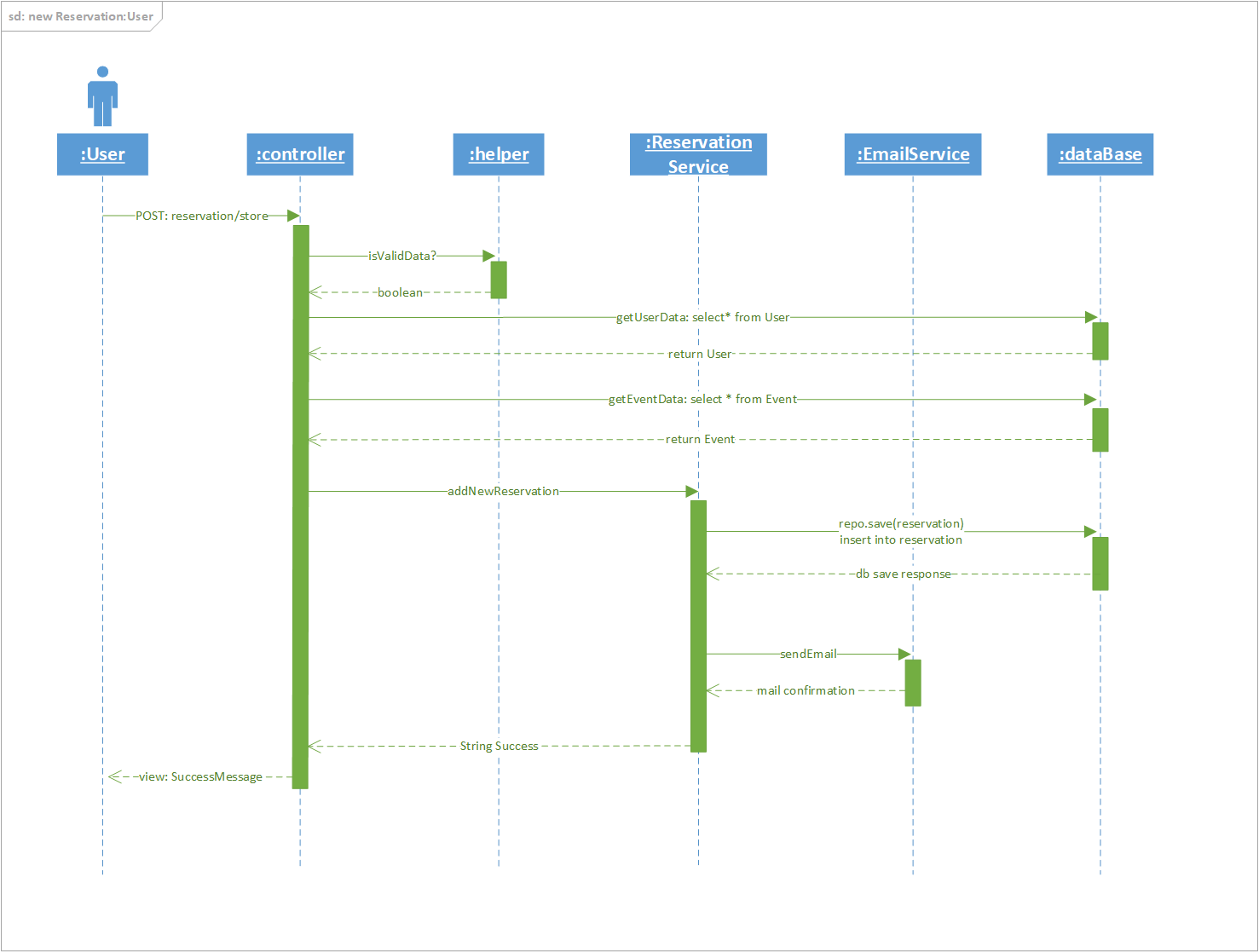
*sd:Admin view Events*



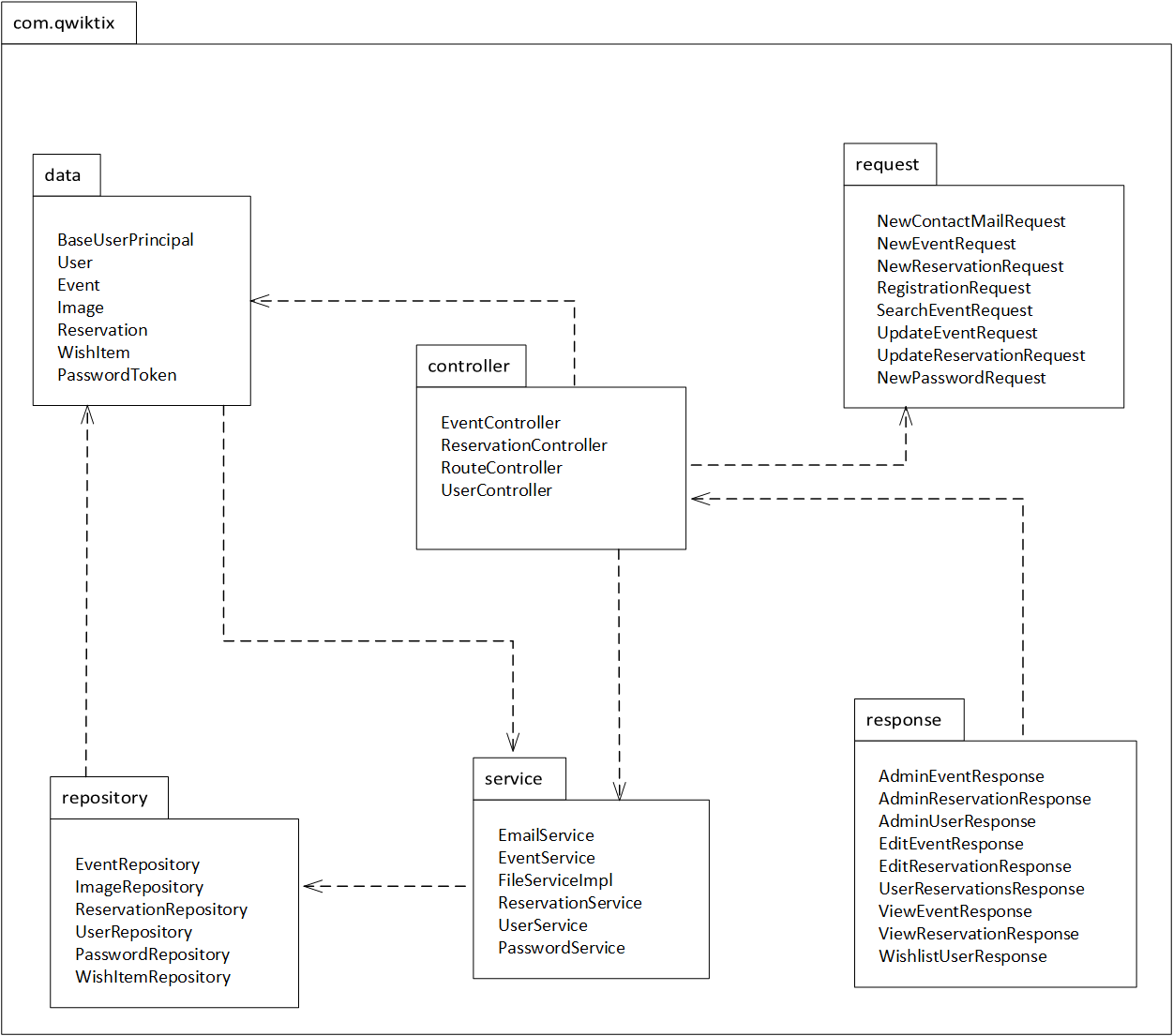
*sd:save event*



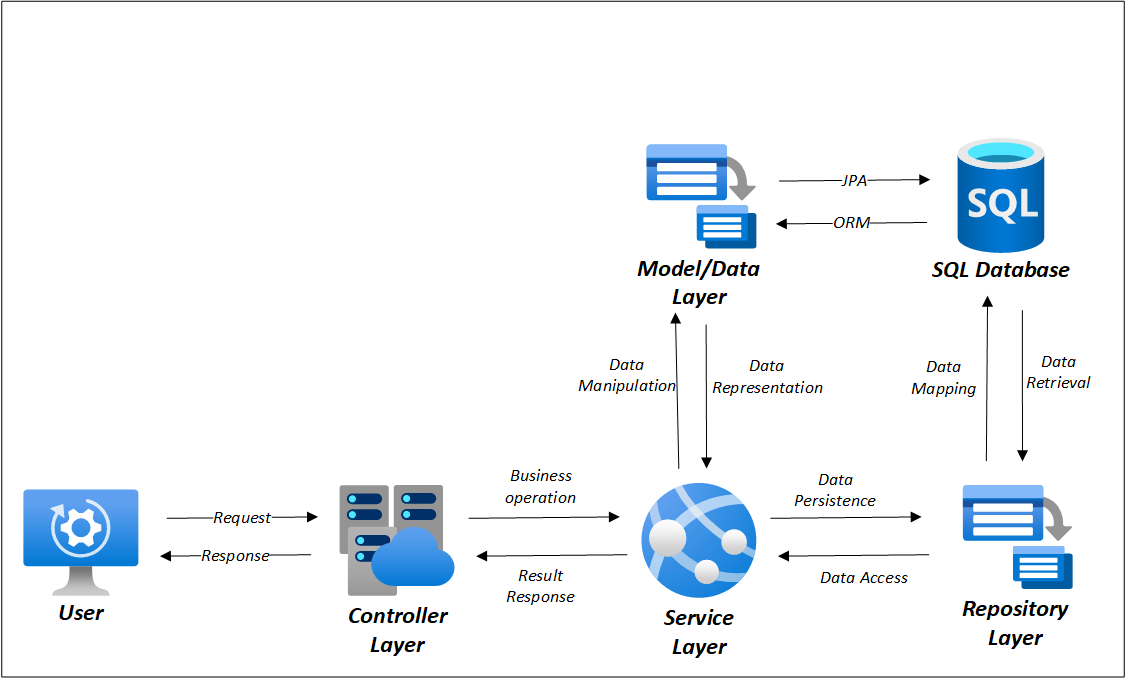
*Sd: new Reservation*



Package Diagram:

**

Architecture Diagram:



Dependencies Used in Project:

1. spring-boot-starter-web: Provides essential libraries for building web applications using Spring MVC.
2. spring-boot-starter-data-jpa: Starter for using Spring Data JPA for database access.
3. spring-boot-starter-validation: Starter for integrating validation support into Spring Boot applications.
4. spring-boot-starter-security: Starter for integrating Spring Security into Spring Boot applications for authentication and authorization.
5. commons-io: Apache Commons IO library for common input/output operations.
6. aws-java-sdk: SDK for interacting with Amazon Web Services (AWS) from Java applications.
7. jaxb-api: Java Architecture for XML Binding (JAXB) API for converting Java objects to and from XML.
8. AdminLTE: WebJar for AdminLTE, a free Bootstrap dashboard template.
9. thymeleaf-layout-dialect: Thymeleaf extension for creating reusable layouts and templates.
10. spring-boot-devtools: Developer tools for Spring Boot applications, providing automatic restarts, among other features, in development.
11. mysql-connector-j: MySQL JDBC driver for connecting Spring Boot applications to MySQL databases.
12. spring-boot-starter-thymeleaf: Starter for using Thymeleaf as the template engine in Spring Boot applications.
13. bootstrap: WebJar for Bootstrap, a popular front-end framework for building responsive websites and web applications.
14. lombok: Library for reducing boilerplate code in Java applications by providing annotations to generate common code.
15. spring-boot-starter-mail: Starter for sending emails from Spring Boot applications.
16. spring-boot-starter-test: Starter for testing Spring Boot applications with JUnit and other testing libraries.