**Sign-up Page Documentation for Medicosphere**

**1. Introduction**

This document serves as a comprehensive guide for the design and implementation of the sign-up page within the Medicosphere Spring Boot application. It outlines the functional requirements, design considerations, API documentation, database schema, security measures, testing strategy, and deployment instructions for the sign-up functionality.

**2. Functional Requirements**

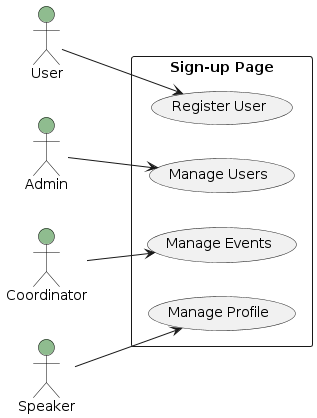
The sign-up page aims to provide a seamless user registration experience, catering to different user roles with distinct privileges:

**Admin:** Possesses full administrative rights, including user management and system settings.

**Coordinator:** Manages event-related activities, such as creating and managing events, inviting speakers, and overseeing attendee registrations.

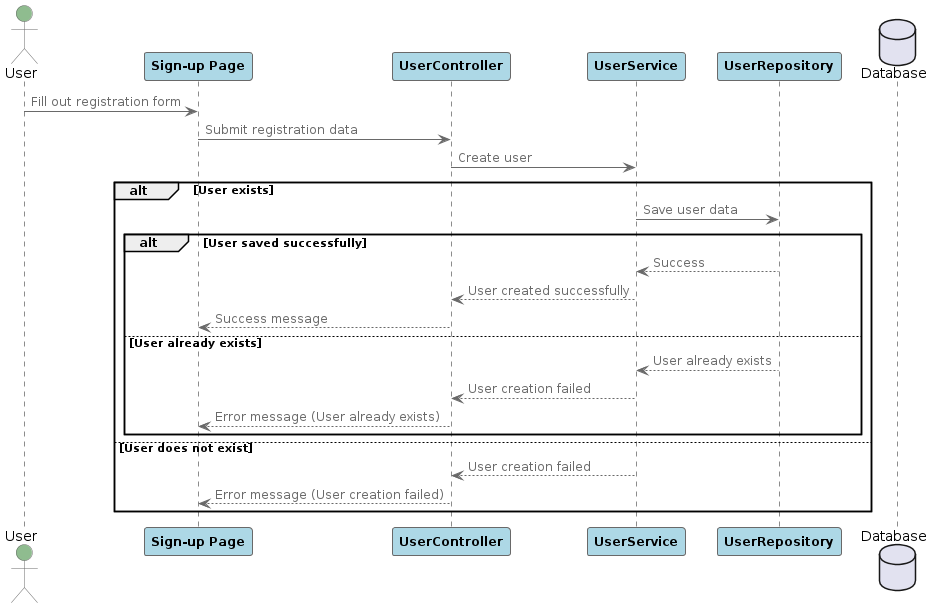
**Speaker:** Manages personal profile information and interacts with event attendees.

**3. Use Case Diagram**

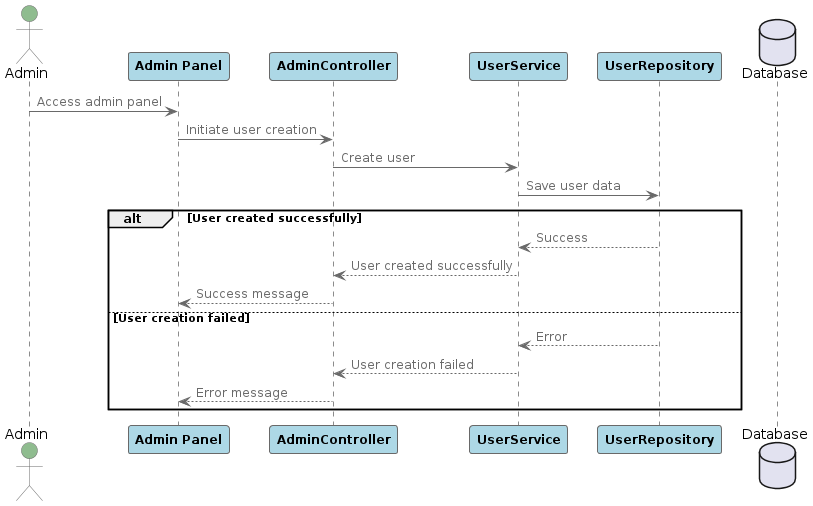


**4. Sequence Diagrams**

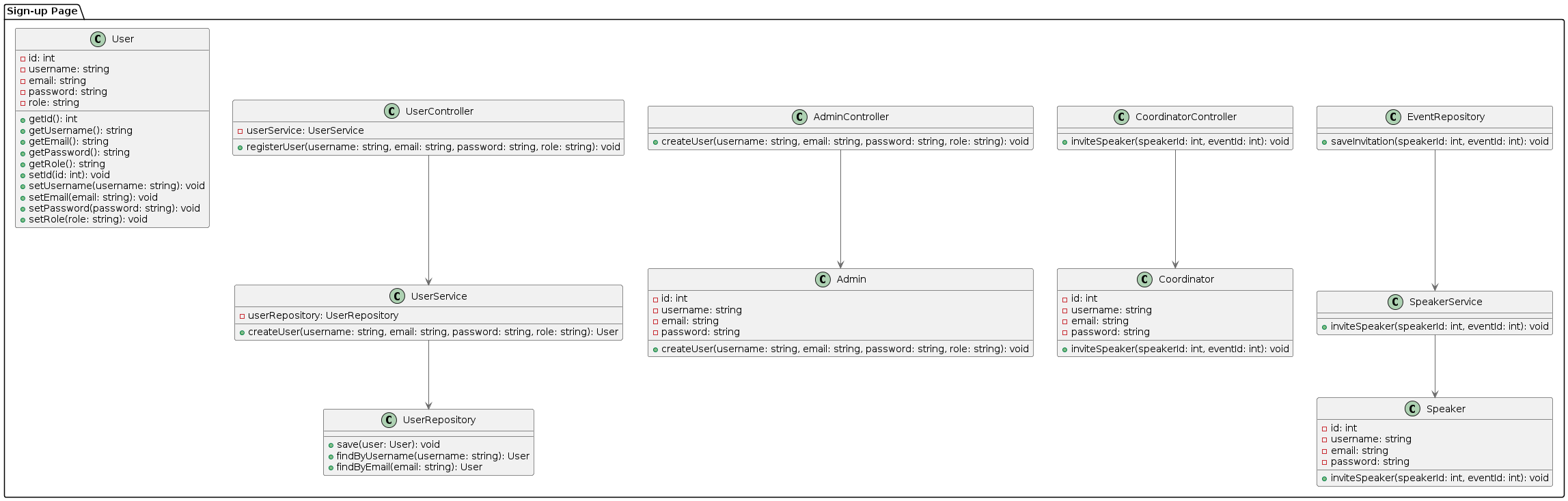
User Registration Process



Admin Creating a New User



**5. Class Diagram**



**6. API Documentation**

POST /api/signup

Description: Endpoint for user registration.

Request Parameters:

username (string)

email (string)

password (string)

role (string)

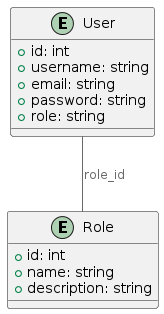
Response:

Success: HTTP 200 OK

Failure: HTTP 4xx or 5xx status codes with error message

Additional endpoints as needed...

**7. Database Schema**



**8. Security Considerations**

Implement secure password hashing algorithms (e.g., bcrypt) to protect user passwords.

Enforce strict input validation to prevent injection attacks and ensure data integrity.

Utilize industry-standard authentication and authorization mechanisms to control access to sensitive endpoints and data.

**9. Error Handling**

Provide clear and informative error messages for invalid requests or failed operations to aid in troubleshooting and user guidance.

Implement robust exception handling to gracefully manage unexpected errors and maintain system stability.

**10. Testing Strategy**

Develop a comprehensive suite of automated unit tests to verify the correctness of individual components.

Perform integration tests to validate the interaction between different layers and components.

Conduct end-to-end testing to ensure the seamless functionality of the user registration process.

**11. Deployment Instructions**

Configure environment-specific properties and settings for seamless deployment across development, staging, and production environments.

Utilize containerization technologies (e.g., Docker) and orchestration tools (e.g., Kubernetes) for efficient and scalable deployment.

**12. Conclusion**

This document provides a detailed and structured approach to implementing the sign-up page within the Medicosphere Spring Boot application. By adhering to the guidelines outlined herein, developers can ensure the successful integration of the sign-up functionality, maintaining security, reliability, and scalability across the application.

**File Structure**

**signup-page/**

**│**

**├── build.gradle**

**├── src/**

**│ ├── main/**

**│ │ ├── java/**

**│ │ │ └── com/**

**│ │ │ └── medicosphere/**

**│ │ │ └── signupage/**

**│ │ │ ├── controller/**

**│ │ │ │ ├── UserController.java**

**│ │ │ │ └── AdminController.java**

**│ │ │ ├── service/**

**│ │ │ │ ├── UserService.java**

**│ │ │ │ ├── SpeakerService.java**

**│ │ │ │ └── CoordinatorService.java**

**│ │ │ ├── repository/**

**│ │ │ │ ├── UserRepository.java**

**│ │ │ │ └── RoleRepository.java**

**│ │ │ ├── model/**

**│ │ │ │ ├── User.java**

**│ │ │ │ └── Role.java**

**│ │ │ └── util/**

**│ │ │ └── EncryptionUtil.java**

**│ │ ├── resources/**

**│ │ │ ├── static/**

**│ │ │ │ └── css/**

**│ │ │ │ └── style.css**

**│ │ │ ├── templates/**

**│ │ │ │ └── signup.html**

**│ │ │ └── application.properties**

**│ │ └── webapp/ (For Servlet based applications)**

**│ └── test/**

**│ ├── java/**

**│ │ └── com/**

**│ │ └── medicosphere/**

**│ │ └── signuppage/**

**│ │ └── controller/**

**│ │ └── UserControllerTest.java**

**│ └── resources/**

**└── settings.gradle**