**Specification Document: Railway Crossing Status Application Prototype**

**Application Capabilities:**

**Real-Time Status Communication:**

The application will provide real-time status updates of railway crossings to the public.

Users will receive timely information regarding whether a crossing is open, closed, or under maintenance.

**User-Friendly Interface:**

The application will feature an intuitive and user-friendly interface to ensure easy navigation.

Users can quickly access the status of specific railway crossings without any technical complexity.

**Notification System:**

Implement a notification system to alert users about changes in crossing status, ensuring they stay informed even when not actively using the app.

**Crossing Details:**

Display additional information about each railway crossing, such as location, schedule, and any upcoming maintenance activities.

**Access Control:**

Implement access control to restrict certain features to authorized personnel (e.g., maintenance staff, administrators).

**Appearance:**

**Clean and Intuitive Design:**

Use a clean and modern design to enhance user experience.

Ensure the interface is intuitive for users of all ages and backgrounds.

**Visual Indicators:**

Incorporate visual indicators (color codes, icons) to quickly convey the status of railway crossings.

Use easily distinguishable elements to enhance accessibility.

**Responsive Design:**

Ensure the application is responsive to different devices (desktop, tablet, mobile) for a seamless user experience.

**User Interactions:**

**Interactive Map:**

Implement an interactive map to allow users to explore and select specific railway crossings.

Clicking on a crossing should provide detailed information and the current status.

**Search Functionality:**

Include a search functionality for users to find a specific railway crossing quickly.

**User Feedback:**

Provide a feedback mechanism for users to report issues or suggest improvements.

Java EE Concepts to be Used:

**Servlets:**

Use Servlets to handle backend processing, manage HTTP requests, and interact with the application's logic.

**JavaServer Pages (JSP):**

Develop the frontend using JSP for dynamic content generation and presentation.

**Enterprise JavaBeans (EJB):**

Utilize EJB for managing business logic and improving modularity.

**Java Message Service (JMS):**

Consider JMS for implementing the notification system.

**Frontend and Backend Technologies:**

**Frontend:**

JavaServer Pages (JSP) for dynamic content generation.

HTML5, CSS3, and JavaScript for designing and enhancing the user interface.

**Backend:**

Servlets for handling HTTP requests and managing backend processes.

JavaBeans for modular and reusable components.

Utilize frameworks like Spring for additional support if needed.

**Database:**

**MySQL:**

Implement MySQL for the database to store information about railway crossings, their statuses, and additional details.

Use JDBC for basic CRUD operations or consider Hibernate for enhanced data handling.

Note: The above specifications provide a foundation for the prototype. Actual implementation details may vary based on project requirements and discussions with stakeholders during the development process.