**Document Guide for Running the Bug Tracking System on Tomcat using JSP and Servlets**

**1. Prerequisites**

Before we start, ensure to have the following software installed:

* **Java Development Kit (JDK) 8 or higher**: Make sure the Java environment variable (JAVA\_HOME) is set.
* **Apache Tomcat 9 or higher**: A servlet container for deploying and running your Java web applications.
* **Integrated Development Environment (IDE)**: Eclipse, IntelliJ IDEA, or NetBeans (recommended for ease of use with Java web projects).
* **MySQL or any relational database** (if using a database backend).

**2. Project Structure Overview**

The project would follow a standard Java web project structure:

bash

Copy code

BugTrackingSystem/

│

├── src/

│ ├── main/

│ │ ├── java/ # Java source files

│ │ │ ├── com.example.dao # DAO layer

│ │ │ ├── com.example.service # Service layer

│ │ │ ├── com.example.exceptions # Custom exceptions

│ │ │ ├── com.example.model # Model classes

│ │ │ ├── com.example.servlet # Servlet classes

│ │ └── resources/

│ └── test/

│

├── src/main/webapp/ # Web application directory

│ ├── WEB-INF/

│ │ ├── web.xml # Deployment descriptor

│ ├── jsp/ # JSP files

│ ├── css/ # CSS files

│ ├── js/ # JavaScript files

│ └── images/ # Image files

│

└── pom.xml # Maven Project Object Model (if using Maven)

**3. Configuring Apache Tomcat**

1. **Download and Install Apache Tomcat:**
   * Download Apache Tomcat 9 or higher from [Apache Tomcat official website](https://tomcat.apache.org/).
   * Extract the downloaded ZIP file to a directory of your choice (e.g., C:\apache-tomcat-9.x).
2. **Set Up Apache Tomcat in Your IDE:**
   * Open your IDE (Eclipse/IntelliJ IDEA/NetBeans).
   * Configure a new server:
     + **Eclipse:** Window > Preferences > Server > Runtime Environments > Add > Select Apache Tomcat v9.x > Provide the installation directory > Finish.
     + **IntelliJ IDEA:** File > Settings > Build, Execution, Deployment > Application Servers > + > Select Tomcat Server > Provide the installation directory > Apply > OK.

**4. Setting Up the Database (Optional)**

As we are using a relational database like MySQL:

1. **Install MySQL** (if not already installed) and start the MySQL service.
2. **Create the Database and Tables:**
   * Use the MySQL command line or a GUI tool like MySQL Workbench.
   * Create a database, e.g., bug\_tracking\_system.
   * Create necessary tables for users, projects, bugs, etc., as per your application requirements.
3. **Configure Database Connection:**
   * Update your web.xml or a db.properties file with the correct database connection details (URL, username, password).
   * Ensure the JDBC driver for MySQL is added to the WEB-INF/lib directory or configured as a Maven dependency.

**5. Deploying the Application on Tomcat**

1. **Build the Project:**
   * If using Maven, run mvn clean install to build the project. This will generate a WAR (Web Application Archive) file in the target directory.
   * If using an IDE, right-click on the project > Run As > Maven build... > Goals: clean install > Run.
2. **Deploy the WAR File to Tomcat:**
   * Copy the generated BugTrackingSystem.war file to the webapps directory of your Tomcat installation.
   * Alternatively, you can use your IDE:
     + **Eclipse:** Right-click on the project > Run As > Run on Server > Choose the configured Tomcat server > Finish.
     + **IntelliJ IDEA:** Right-click on the project > Run > Edit Configurations > Add a new Tomcat configuration > Deploy the artifact.
3. **Start Tomcat Server:**
   * Start Tomcat via the command line (bin/startup.bat for Windows or bin/startup.sh for Linux/Mac).
   * Or start from the IDE by clicking the "Start" button in the Servers view (Eclipse) or Run Configuration (IntelliJ).

**6. Accessing the Application**

1. **Open Your Web Browser:**
   * Go to http://localhost:8080/BugTrackingSystem/.
2. **Testing the Application:**
   * **User Registration:** Register different types of users (Project Manager, Developer, Tester).
   * **Login:** Use different credentials to log in and verify that each role is directed to the correct main page.
   * **Project Management:** Test creating projects, assigning team members, and setting project statuses.
   * **Bug Reporting and Management:** Test bug reporting by testers, bug assignment by project managers, and bug status updates by developers.

**7. Important Configuration Files**

1. **web.xml (Deployment Descriptor):**
   * Ensure all servlets are properly mapped, e.g.,:

xml

Copy code

<servlet>

<servlet-name>LoginServlet</servlet-name>

<servlet-class>com.example.servlet.LoginServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>LoginServlet</servlet-name>

<url-pattern>/login</url-pattern>

</servlet-mapping>

1. **Database Configuration (db.properties or similar):**
   * If using a database, ensure correct properties are set:

properties

Copy code

db.url=jdbc:mysql://localhost:3306/bug\_tracking\_system

db.username=root

db.password=yourpassword

db.driverClassName=com.mysql.cj.jdbc.Driver

1. **Logging Configuration (Optional):**
   * Configure log4j.properties or logging.properties for application logging.