# **Calculator Application Documentation**

## **Project Overview**

The **Calculator Application** is a highly advanced Java-based web application meticulously designed to streamline and enhance arithmetic computation processes. By leveraging a modular and scalable architecture, it seamlessly integrates Java servlets, JavaServer Pages (JSP), and a MySQL database. This integration facilitates efficient tracking of operations and provides an intuitive user interface for robust and effective interaction. The application is built with a focus on maintainability, extensibility, and performance, making it a foundational tool for both educational and professional computational tasks.

## **Components**

### **1. Backend**

#### **Main.java**

* Operates as the primary entry point of the application, initializing essential configurations and orchestrating the runtime environment to ensure optimal performance.

#### **DatabaseConnection.java**

* Provides robust and reliable connectivity to the MySQL database through advanced connection pooling techniques.
* Implements sophisticated error-handling mechanisms, ensuring database interactions are both efficient and resilient under high load conditions.

#### **CalculationsDAO.java**

* Manages all CRUD (Create, Read, Update, Delete) operations associated with the calculations table in the database.
* Guarantees the persistence and integrity of computation history, supporting comprehensive auditing and data analysis capabilities.

#### **Calculations.java**

* Defines the Calculations data model, encapsulating core properties such as operation, operand1, operand2, and result.
* Serves as a bridge between the application logic and the database layer, ensuring seamless data flow.

#### **CalculationServlet.java**

* Functions as the central processing unit for client-initiated requests related to computations.
* Collaborates with the DAO layer to retrieve, store, and manage computation data while dynamically returning results to the user interface.

### **2. Frontend**

#### **index.jsp**

* Acts as the user-facing interface, enabling users to input numerical data and select specific arithmetic operations.
* Leverages JSP to dynamically render results, ensuring a responsive and engaging user experience.

#### **web.xml**

* Constitutes the application’s deployment descriptor, detailing servlet mappings, session configurations, and additional operational directives.

## **Features**

* **Comprehensive Arithmetic Support**: Executes fundamental operations, including addition, subtraction, multiplication, and division, with precision and reliability.
* **Real-Time Processing**: Utilizes servlet technology to deliver instantaneous computation results.
* **Persistent History Tracking**: Stores a detailed log of all user calculations in the MySQL database for future reference.
* **Dynamic Interface**: Enhances user interaction through JSP-driven dynamic result rendering.

## **Database Setup**

### **SQL File: createtable.sql**

This script establishes the schema for the calculations table, which is integral to the application’s data storage capabilities:

CREATE TABLE calculations (

id INT AUTO\_INCREMENT PRIMARY KEY,

operand1 DOUBLE NOT NULL,

operand2 DOUBLE NOT NULL,

operation VARCHAR(10) NOT NULL,

result DOUBLE NOT NULL,

timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

### **Steps to Set Up the Database:**

1. **Create the Database**: Initialize a new database in your MySQL server environment.

**Execute the SQL Script**: Run the provided createtable.sql file to generate the necessary table structure:  
 mysql -u username -p database\_name < createtable.sql

1. **Update Connection Settings**: Modify the DatabaseConnection.java file to reflect your database credentials and configuration.

## **Build and Deployment**

### **Prerequisites**

* **JDK 11 or Higher**: Ensure the latest version of the Java Development Kit is installed.
* **Apache Maven**: Used for project build automation and dependency management.
* **MySQL Server**: Required for data storage and retrieval operations.

### **Steps to Run the Application:**

**Clone the Repository**:  
 git clone https://github.com/your-username/tushar-502-calculator.git

cd tushar-502-calculator

**Install Dependencies and Build**: Execute the Maven build process to resolve dependencies and prepare the application:  
 mvn clean install

1. **Deploy to Server**:
   * Transfer the generated WAR file to a servlet container such as Apache Tomcat.
   * Start the server and access the application through a web browser at http://localhost:8080/tushar-502-calculator/.

## **Future Enhancements**

1. **Advanced Operations**: Incorporate complex mathematical functionalities, including exponential computations, logarithms, and trigonometric evaluations.
2. **User Authentication**: Implement robust user authentication and authorization mechanisms to facilitate personalized calculation histories.
3. **Enhanced UI/UX**: Integrate modern frontend frameworks such as React.js or Angular to improve the aesthetic appeal and usability of the application.
4. **Analytics and Reporting**: Introduce data analytics modules to provide insights into user behavior and calculation trends.
5. **Mobile Compatibility**: Develop a mobile-responsive version of the application to cater to users on handheld devices.

## **License**

This project is distributed under the MIT License, granting users the freedom to modify and utilize the software in accordance with the specified terms.