**Electricity Bill Calculator**

**Question No: 19**

**Design and develop a responsive website to calculate the electricity bill using Spring Boot and React.**

**Conditions for calculating the bill:**

* **For first 50 units** – Rs. 3.50/unit
* **For next 100 units** – Rs. 4.00/unit
* **For next 100 units** – Rs. 5.20/unit
* **For units above 250** – Rs. 6.50/unit

You can make use of **Bootstrap** and **jQuery**.

**Environment Setup**

**Spring Boot Setup (Backend)**

1. **Install Spring Boot Extension Pack in VS Code**:
   * Open **VS Code** and navigate to the Extensions panel (Ctrl+Shift+X).
   * Search for **Spring Boot Extension Pack** and click **Install**. This will install useful extensions like **Spring Boot Tools**, **Spring Initializr**, etc.
2. **Create Spring Boot Project**:
   * **Open VS Code** and press *Ctrl+Shift+P* to open the Command Palette.
   * Type *Spring Initializr: Generate a Maven Project* and select it.
   * Select **Maven** as the build tool.
   * Choose **Java** as the language.
   * Select the **Spring Boot version** (e.g., 2.7.x or the latest stable version).
   * Group: *com.example*
   * Artifact: *electricitybill*
   * Choose dependencies:
     + **Spring Web** (for building RESTful APIs)
     + **Spring Boot DevTools** (for automatic restarts during development)
   * Click **Generate**, and VS Code will create the project for you.
3. **Navigate to the Project Folder**:
   * In the terminal, run the following command to navigate to the project folder:

*cd electricitybill*

1. **Create the Model:**

*In src/main/java/com/example/electricitybill/model/ElectricityBill.java:*In src/main/java/com/example/electricitybill/model/ElectricityBill.java:  
  
package com.example.electricitybill.model;  
  
public class ElectricityBill {  
  
 private int units;  
 private double totalBill;  
  
 *// Getters and Setters*  
 public int getUnits() {  
 return units;  
 }  
  
 public void setUnits(int units) {  
 this.units = units;  
 }  
  
 public double getTotalBill() {  
 return totalBill;  
 }  
  
 public void setTotalBill(double totalBill) {  
 this.totalBill = totalBill;  
 }  
}

1. **Create the Service (Business Logic):**

*src/main/java/com/example/electricitybill/service/BillCalculatorService.java:*src/main/java/com/example/electricitybill/service/BillCalculatorService.java:  
  
package com.example.electricitybill.service;  
  
import org.springframework.stereotype.Service;  
  
@Service  
public class BillCalculatorService {  
  
 public double calculateElectricityBill(int units) {  
 double totalBill = 0;  
  
 if (units <= 50) {  
 totalBill = units \* 3.50;  
 } else if (units <= 150) {  
 totalBill = 50 \* 3.50 + (units - 50) \* 4.00;  
 } else if (units <= 250) {  
 totalBill = 50 \* 3.50 + 100 \* 4.00 + (units - 150) \* 5.20;  
 } else {  
 totalBill = 50 \* 3.50 + 100 \* 4.00 + 100 \* 5.20 + (units - 250) \* 6.50;  
 }  
  
 return totalBill;  
 }  
}

1. **Create the Controller:**

src/main/java/com/example/electricitybill/controller/BillController.java:  
  
package com.example.electricitybill.controller;  
  
import com.example.electricitybill.model.ElectricityBill;  
import com.example.electricitybill.service.BillCalculatorService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/api/bill")  
@CrossOrigin(origins = "http://localhost:3000")  
public class BillController {  
  
 @Autowired  
 private BillCalculatorService billCalculatorService;  
  
 @PostMapping("/calculate")  
 public ElectricityBill calculateBill(@RequestBody ElectricityBill bill) {  
 double totalBill = billCalculatorService.calculateElectricityBill(bill.getUnits());  
 bill.setTotalBill(totalBill);  
 return bill;  
 }  
}

1. **Main Application Class:**

src/main/java/com/example/electricitybill/ElectricityBillApplication.java:  
  
package com.example.electricitybill;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class ElectricityBillApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.run(ElectricityBillApplication.class, args);  
 }  
}

1. **Running the Spring Boot Application**:
   * Use the following command to run the Spring Boot application in Windows (from the project folder):
2. *mvnw.cmd spring-boot:run*To verify that Maven is installed, open the Command Prompt and run:  
   *mvn -v*

This will start the backend server on http://localhost:8080.

**Frontend Setup (React)**

1. **Create React Application**:
   * Open a new terminal and run the following command to create a React app named frontend:
2. *npx create-react-app frontend*
3. **Navigate to the frontend folder**:
   * After the project creation is complete, change to the frontend directory:
4. *cd frontend*
5. **Install Dependencies**:  
   * You need to install **Axios** (for HTTP requests) and **Bootstrap** (for responsive UI):
6. *npm install axios bootstrap*
7. **Update src/App.js:**

import React, { useState } from 'react';  
import axios from 'axios';  
import 'bootstrap/dist/css/bootstrap.min.css';  
  
function App() {  
 const [units, setUnits] = useState('');  
 const [totalBill, setTotalBill] = useState(null);  
 const [error, setError] = useState('');  
  
 const handleChange = (event) => {  
 setUnits(event.target.value);  
 };  
  
 const handleSubmit = async (event) => {  
 event.preventDefault();  
 if (units <= 0 || isNaN(units)) {  
 setError('Please enter a valid number of units');  
 return;  
 }  
  
 try {  
 *// Send POST request to Spring Boot backend*  
 const response = await axios.post('http://localhost:8080/api/bill/calculate', { units });  
 setTotalBill(response.data.totalBill);  
 setError('');  
 } catch (error) {  
 setError('Error calculating the bill. Please try again.');  
 }  
 };  
  
 return (  
 <div className="container mt-5">  
 <h1 className="text-center">Electricity Bill Calculator</h1>  
 <form onSubmit={handleSubmit}>  
 <div className="form-group">  
 <label htmlFor="units">Enter Units:</label>  
 <input  
 type="number"  
 className="form-control"  
 id="units"  
 value={units}  
 onChange={handleChange}  
 placeholder="Enter number of units"  
 required  
 />  
 </div>  
 {error && <div className="alert alert-danger mt-3">{error}</div>}  
 <button type="submit" className="btn btn-primary mt-3">Calculate</button>  
 </form>  
  
 {totalBill !== null && (  
 <div className="mt-4">  
 <h3>Total Bill: Rs. {totalBill}</h3>  
 </div>  
 )}  
 </div>  
 );  
}  
  
export default App;

1. **Start the React Application**:
   * Run the following command to start the React development server:
2. *npm start*

This will start the React app on http://localhost:3000.

**Running the Application**

1. **Run the Spring Boot Backend**:
   * In a separate terminal window, go to the electricitybill folder and run:
2. *mvnw.cmd spring-boot:run*
3. **Run the React Frontend**:
   * In the terminal, navigate to the frontend folder and run:
4. *npm start*

The React app will now be accessible at http://localhost:3000.

1. **Testing**:
   * Open a web browser and go to http://localhost:3000.
   * Enter a value for the number of units (e.g., 350) and click **Calculate**.
   * The frontend will make a request to the backend to calculate the bill, and the result will be displayed.