A PROJECT REPORT

**On**

ENERGY TRACKING SYSTEM

*Submitted in partial fulfilment of the requirements for the award of the degrees*

***of***

### BACHELOR OF TECHNOLOGY

**in INFORMATION TECHNOLOGY**

*Submitted by:*

### Ayush Gupta, Antara A. Kumbhare & Aditya Ray

*Guided by:*

### Dr Ani Thomas

**Head Of Department, Information Technology Branch**



**BHILAI INSTITUTE OF TECHNOLOGY DURG**

#### DEPARTMENT OF INFORMATION TECHNOLOGY

**UGC Autonomous Institution**

**(Affiliated to CSVTU, Approved by AICTE, NBA &NAAC ACCREDIATED)**

**DURG– 491001, CHHATTISGARH, INDIA**

[**www.bitdurg.ac.in**](http://www.bitdurg.ac.in/) **SESSION: 2023-24**

**CANDIDATE’S DECLARATION**

We hereby declare that the project entitled **“Energy Tracking System”** submitted in partial fulfilment for the award of the degree of Bachelor of Technology in Information Technology completed under the supervision of **Dr Ani Thomas, Head Of Department, Information Technology Branch** BIT DURG is an authentic work.

Further, I/we declare that I/we have not submitted this work for the award of any other degree elsewhere.

Ayush Gupta

Antara A. Kumbhare

Aditya Ray

### Signature and name of the student(s) with date

**\_ \_ CERTIFICATE by PROJECT Guide(s)**

It is certified that the above statement made by the students is correct to the best of my/our knowledge.

**Signature of BTP Guide(s) with dates and their designation**



## BHILAI INSTITUTE OF TECHNOLOGY DURG

#### DEPARTMENT OF INFORMATION TECHNOLOGY

**UGC Autonomous Institution**

**(Affiliated to CSVTU, Approved by AICTE, NBA &NAAC ACCREDIATED) DURG– 491001, CHHATTISGARH, INDIA**

**Department of Information Technology**



## CERTIFICATE BY THE EXAMINERS

This is to certify that the Major Project work entitled “**Energy Tracking System”** is carried out by **Ayush Gupta (300103321025), Antara A. Kumbhare (300103321053), Aditya Ray (300103321056)** in partial fulfilment for the award of degree of **Bachelor of Technology** in **Information Technology**, **Bhilai Institute Of Technology**, **Durg** during the academic year 2023-2024.

Mr Amrendra Kumar Singh External Examiner Prof. Dr. Ani Thomas

Internal Guide HOD

**ACKNOWLEDGEMENTS**

We wish to acknowledge with a deep sense of hearty gratitude and indebtedness to **Mr Amrendra Kumar Singh** Information Technology, who gave us this opportunity to experience project work & his valuable suggestion during this project have been invaluable.

We take this opportunity to voice & record our sincerest gratefulness towards our esteem Supervisor **Dr Ani Thomas** under whose able guidance the project work has been brought to completion.

Our heart leaps up in thankfulness for his benevolence & time to time help, valuable suggestions, constructive criticism & active interest in the successful completion of this project work.

We are also thankful to all our honorable teachers of the Information Technology Department and our parents whose valuable support helped us and kept us motivated all through.

**Ayush Gupta**

**Antara A. Kumbhare**

**Aditya Ray**

**B.Tech. V Sem**

**Discipline of Information Technology**

**BIT DURG**

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter** | **Chapter Title** | **Section Title** | **Page no.** |
|  | **ABSTRACT** | Abstract | 6 |
| **I** | **INTRODUCTION** | Overview | 7 |
| Objectives and Scope | 7 |
| Importance Of Energy Tracking System | 7 |
| **II** | **PROBLEM IDENTIFICATION** | Problem Statement | 8 |
| **III** | **METHODOLOGY** | Project Design and Architecture | 8 - 9 |
| Implementation | 10 - 11 |
| Features and Functionality | 12 |
| Testing and Results | 13 - 14 |
| IV | **CONCLUSION** | Summary | 15 |
|  |  | Future Enhancements and Possible | 15-16 |
|  |  | Expansions |
| **V** | **CODE SNIPPETS** | Actual Code | 17 - 24 |
| Screenshots | 25 |
| **VI** | **BIBLOGRAPHY** | | 26 |

5

# ABSTRACT

The "Electricity Consumption Calculator" is a Java-based application designed to empower users in managing and optimizing their household energy usage. Focused on user-friendliness and real-time feedback, the calculator allows individuals to input details for a variety of common appliances, considering parameters such as quantity, usage time, and duration. Leveraging JavaFX for a visually appealing and intuitive interface, the application provides instant feedback on electricity consumption and costs, promoting informed decision-making. Users can customize predefined appliance data, add new devices, and export detailed reports for comprehensive insights. The system employs robust testing methodologies, including unit tests and user acceptance testing, to ensure accuracy, usability, and responsiveness. Anticipated results encompass precise calculations, an intuitive user interface, and valuable insights gleaned from user feedback, contributing to a comprehensive and user-centric electricity management tool. The "Electricity Consumption Calculator" is poised to enhance energy awareness, promote efficiency, and contribute to sustainable energy practices in both residential and commercial settings.

Top of Form

6

# INTRODUCTION

### Overview of the Project:

The "Electricity Consumption Calculator" project aims to address the critical need for efficient energy management in households. As the demand for electricity continues to rise, it becomes imperative for individuals to monitor and understand their electricity consumption patterns. This project introduces a user-friendly JavaFX application designed to assist users in estimating the electricity consumption and associated costs of various household appliances.

### Objectives and Scope:

The primary objective of this project is to develop a functional news aggregator utilizing Java programming language and JavaFX for the user interface. The project focuses on integrating these technologies to create a seamless and user-friendly energy tracking system.

"Electricity Consumption Calculator" project aims to achieve a balance between practical utility and educational value. Through a user-friendly interface and a diverse range of applicable appliances, the project seeks to empower users to make informed decisions about their energy consumption in both a practical and educational context.

### Importance of Energy Tracking System:

An energy tracking system holds paramount importance across diverse domains. By meticulously monitoring energy consumption, it enables significant cost savings through the identification and rectification of inefficiencies, facilitating accurate budgeting and financial planning. Moreover, the system plays a pivotal role in environmental sustainability, empowering users to actively reduce their carbon footprint and adopt resource-conserving practices. For businesses, energy tracking enhances operational efficiency by optimizing processes, predicting maintenance needs, and ensuring compliance with regulatory standards. Transparent reporting of energy usage not only aligns with corporate social responsibility goals but also incentivizes behavioral changes towards energy conservation. The system contributes to grid management and emergency preparedness, ensuring a reliable and resilient energy infrastructure. Ultimately, the multifaceted benefits of an energy tracking system underscore its critical role in promoting economic savings, environmental stewardship, and operational excellence.

# PROBLEM IDENTIFICATION

### Problem Statement:

The absence of a comprehensive and user-friendly energy tracking system poses a significant challenge, hindering individuals and organizations from effectively managing and optimizing their energy consumption. Without a streamlined solution, identifying and rectifying inefficiencies becomes challenging, leading to unnecessary costs and environmental impact. Businesses lack the tools necessary to strategically plan and reduce peak energy usage, resulting in operational inefficiencies and increased expenses. Moreover, the absence of transparent energy reporting impedes regulatory compliance and hinders efforts towards corporate social responsibility. In the absence of a robust energy tracking system, there is a critical gap in enabling informed decision-making, promoting sustainable practices, and contributing to a resilient and efficient energy ecosystem. Addressing this problem is crucial for realizing cost savings, enhancing environmental stewardship, and achieving operational excellence.

# METHODOLOGY

## Project Design and Architecture:

### Project planning and scope definition:

**Objective Refinement:** Clarified project objectives, emphasizing the development of a user-friendly Electricity Consumption Calculator for effective energy tracking.

**Scope Definition:** Clearly outlined the scope of the project, including features such as real-time calculations, predefined appliance data, and a JavaFX-based user interface.

### User Interface (UI):

Utilizes JavaFX to create the graphical interface for user interaction.

### System Design:

**Functional Requirements Specification:** Defined detailed functional requirements, including data input, real-time calculations, and result presentation.

**User Interface Design:** Created wireframes and prototypes to design an intuitive JavaFX-based interface, emphasizing simplicity and accessibility.

### Data Preprocessing:

**Predefined Appliance Data:** Compiled a list of common household appliances with associated

power ratings and electricity rates, incorporating this data into the application for user convenience.

### Application Development:

**JavaFX Implementation:** Developed the application using JavaFX, utilizing its features for graphical interface design, event handling, and seamless user interaction.

**Algorithm Implementation:** Implemented algorithms for real-time calculations of electricity consumption and associated costs based on user input.

## Implementation:

### Technology Stack:

**Programming Language:** Utilized Java for its versatility, platform independence, and extensive community support.

**User Interface Framework:** Employed JavaFX for GUI development, leveraging its rich set of components and ease of integration with Java applications.

### System Architecture:

**MVC Design Pattern:** Adopted the Model-View-Controller (MVC) design pattern for a modular and organized structure.

**Separation of Concerns:** Ensured clear separation between the user interface, application logic, and data representation.

**Data Model:**

**Lists for Appliance Data**: Implemented lists to store predefined appliance data, including names, power ratings, and electricity rates**.**

**Dynamic Data Storage**: Allowed for dynamic storage of user-inputted data, such as the number of appliances, usage times, and number of days.

### User Interface Development with JavaFX:

**JavaFX Components:** Utilized JavaFX components such as labels, text fields, buttons, and layout panes for constructing an intuitive and visually appealing interface.

**CSS Styling:** Applied CSS for basic styling, ensuring a consistent and visually appealing appearance.

### Real Time Calculations:

**Event Handling:** Implemented event handlers for user input fields to capture real-time changes.

**Calculation Logic:** Developed algorithms for calculating electricity consumption and cost based on user inputs, considering predefined appliance data.

### Predefined Appliance Data:

**Arrays and Lists:** Employed arrays and lists to store predefined appliance data, ensuring ease of access and modification.

**Initialization Process:** Integrated an initialization process to load default appliance data at the start of the application.

### User Input Handling:

**Event Listeners:** Implemented event listeners on input fields for the number of appliances, usage time, and number of days to capture and process user input.

**Exception Handling:** Incorporated exception handling to manage potential errors in user input, providing a smoother user experience.

### Code Structure and Modularity:

**Package Organization:** Organized code into packages for clear separation of concerns and improved maintainability.

**Modular Classes:** Designed modular classes for specific functionalities, promoting code readability and ease of future enhancements.

**Testing:**

**Unit Testing**: Conducted unit tests to ensure the functionality of individual methods and classes.

User Acceptance Testing (UAT): Engaged potential users in UAT sessions to validate the application's usability and identify any user experience issues.

## Features and Functionality:

**Appliance Selection:**

**Predefined Appliance Data:** Users can choose from a predefined list of common household appliances, each with associated power ratings and electricity rates.

**Dynamic Addition:** The application allows users to dynamically add and customize appliances beyond the predefined list.

**User Input:**

**Number of Appliances:** Users can input the quantity of each selected appliance.

**Usage Time:** Users specify the average daily usage time (in hours) for each appliance.

**Number of Days:** Users input the total number of days for which they want to calculate electricity

consumption and cost.

**Real-Time Calculations:**

**Electricity Consumption:** The application dynamically calculates the total electricity consumption

based on the entered data, considering power ratings, usage times, and the number of appliances.

**Electricity Cost:** Real-time cost estimation is provided, taking into account electricity rates and the

calculated consumption.

**User-Friendly Interface:**

**Intuitive Design:** The JavaFX-based interface is designed for simplicity and ease of use.

**Clear Input Prompts:** Prompts and placeholders guide users on what information is required in each

input field.

## Testing and Results:

## Testing Approach:

## Objective: Verify the functionality of individual methods and classes.

## Tests:

## Confirm accurate calculation of electricity consumption and cost.

## Validate input validation and error handling.

## Ensure proper updating of labels and real-time feedback.

## User Acceptance Testing (UAT):

## Objective: Evaluate the application's usability and user experience.

## Tests:

## Verify the intuitiveness of the user interface for inputting data.

## Assess the real-time updating of labels and feedback.

## Evaluate the customization features, including adding custom appliances.

## Expected Results:

## Calculation Accuracy: Unit tests should confirm that the calculations for electricity consumption and cost are accurate, considering predefined and user-inputted data.

## Input Validation: Validate that the application appropriately handles valid and invalid numeric inputs, providing clear error messages when necessary.

## Label Updating: Confirm that labels dynamically update in real-time as users input data, providing immediate and accurate feedback.

## User Acceptance Testing (UAT):

## Usability: Users should find the interface intuitive and easy to navigate, with clear prompts and labels guiding them through the input process.

## Real-Time Feedback: UAT should demonstrate that users receive instant feedback on their input, enhancing the overall user experience.

## Customization Features: Users should be able to successfully customize appliance data, add new appliances, and navigate the application with ease.

## Performance Evaluation:

## Response Time:

## Objective: Assess the application's responsiveness under varying levels of user input.

## Measurement: Measure the time taken for the application to recalculate and update results as users modify input fields.

## Resource Utilization:

## Objective: Evaluate the application's resource usage during regular operations.

## Measurement: Monitor memory and CPU usage to ensure the application operates efficiently.

# Conclusion and Future Work

**Conclusion:**

The Electricity Consumption Calculator is a comprehensive tool designed to assist users in estimating their electricity usage and associated costs based on the usage patterns of various appliances. By providing a user-friendly interface and real-time calculations, the application empowers users to make informed decisions about energy consumption and cost management.

During the development of this project, several key features were implemented, including the input of appliance details, dynamic calculations, and the display of individual and total electricity costs. The integration of JavaFX ensured a responsive and intuitive user interface.

**Scope for Future Enhancements:**

While the current version of the Electricity Consumption Calculator successfully meets its objectives, there are several areas where enhancements and additional features could be implemented to further improve its functionality and user experience. The scope for future enhancements includes:

1. **Real-time Data Updates:** Implement a mechanism to receive real-time updates on electricity rates and appliance power consumption. This could involve integrating with external APIs or databases that provide live data.
2. **User Authentication and Profile Management:** Introduce user authentication and profile management to allow users to save and retrieve their appliance data for future reference. This could enhance the personalization of the application.
3. **Usage History and Analytics:** Incorporate a feature to track and visualize historical electricity usage data. Users could benefit from insights and analytics on their consumption patterns over time.
4. **Notification System:** Implement a notification system to alert users when their electricity consumption is significantly higher than usual or when electricity rates change. This proactive approach can assist users in managing their energy usage effectively.
5. **Energy Saving Recommendations:** Integrate machine learning algorithms or predefined rules to provide users with personalized recommendations on how to optimize their energy consumption and reduce costs.
6. **Multi-Platform Support:** Extend the application to support multiple platforms, including mobile devices and web browsers, to increase accessibility and reach a broader user base.
7. **Localization:** Introduce localization features to support multiple languages and regional settings, making the application more user-friendly for a diverse audience.
8. **Integration with Smart Devices:** Explore integration with smart home devices to automate the collection of appliance usage data, further reducing manual input and enhancing accuracy.

Incorporating these enhancements will not only elevate the capabilities of the Electricity Consumption Calculator but also position it as a more versatile and valuable tool for users seeking real-time insights into their energy consumption habits.

**JavaFX User Interface Creation:**

# Code Snippets

import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.\*;

//import javafx.scene.layout.Background;

import javafx.scene.layout.GridPane;

import javafx.scene.layout.StackPane;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

import java.util.ArrayList;

import java.util.List;

public class ElectricityConsumptionCalculatorFX extends Application {

private final List<String> applianceNames = new ArrayList<>();

private final List<Double> powerRatings = new ArrayList<>();

private final List<Integer> numberOfAppliances = new ArrayList<>();

private final List<Double> usageTimes = new ArrayList<>();

private final List<Double> electricityRates = new ArrayList<>();

private final List<Integer> numberOfDays = new ArrayList<>();

private final Label totalCostLabel = new Label();

private final List<Label> consumptionLabels = new ArrayList<>();

private final List<Label> costLabels = new ArrayList<>();

public static void main(String[] args) {

launch(args);

}

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Electricity Consumption Calculator");

GridPane grid = new GridPane();

grid.setHgap(10);

grid.setVgap(10);

grid.setPadding(new Insets(20, 20, 20, 20));

Label titleLabel = new Label("Welcome to the Electricity Consumption Calculator (Standard values for India)");

titleLabel.setStyle("-fx-font-size: 16pt;");

GridPane.setConstraints(titleLabel, 0, 0, 4, 1);

initializeAppliances();

for (int i = 0; i < applianceNames.size(); i++) {

addUIComponents(grid, i);

}

Button calculateButton = new Button("Calculate");

calculateButton.setOnAction(e -> calculateTotalBill());

GridPane.setConstraints(calculateButton, 0, applianceNames.size() + 2);

totalCostLabel.setStyle("-fx-font-size: 14pt;");

GridPane.setConstraints(totalCostLabel, 0, applianceNames.size() + 3, 4, 1);

VBox contentLayout = new VBox(20); // Vertical box with 20-pixel spacing

contentLayout.setAlignment(Pos.CENTER);

contentLayout.getChildren().addAll(titleLabel, grid, calculateButton, totalCostLabel);

StackPane root = new StackPane(contentLayout);

Scene scene = new Scene(root, 800, 600);

primaryStage.setScene(scene);

primaryStage.show();

}

private void initializeAppliances() {

String[] predefinedAppliances = {

"Ceiling Fan", "Incandescent Bulb", "CFL Bulb", "LED Bulb", "Refrigerator",

"Television", "Air Conditioner", "Washing Machine", "Microwave Oven", "Electric Heater",

"Laptop", "Desktop Computer", "Router", "Water Heater", "Coffee Maker"

};

double[] predefinedPowerRatings = {

0.075, 0.1, 0.015, 0.005, 0.15, 0.1, 1.5, 0.8, 1.2, 2.0,

0.05, 0.1, 0.02, 2.0, 1.0

}; // Power ratings in kW

double[] predefinedElectricityRates = {

5.0, 3.0, 4.0, 2.0, 7.0, 4.0, 8.0, 7.0, 5.0, 6.0,

2.0, 4.0, 1.0, 7.0, 5.0

}; // Electricity rates per kWh

for (int i = 0; i < predefinedAppliances.length; i++) {

applianceNames.add(predefinedAppliances[i]);

powerRatings.add(predefinedPowerRatings[i]);

electricityRates.add(predefinedElectricityRates[i]);

// Initialize lists with default values

numberOfAppliances.add(0);

usageTimes.add(0.0);

numberOfDays.add(0);

}

}

private void addUIComponents(GridPane grid, int index) {

Label nameLabel = new Label(applianceNames.get(index));

GridPane.setConstraints(nameLabel, 0, index + 1);

TextField numAppliancesField = new TextField();

numAppliancesField.setPromptText("Number of Appliance");

GridPane.setConstraints(numAppliancesField, 1, index + 1);

TextField usageTimeField = new TextField();

usageTimeField.setPromptText("Usage per day (hours)");

GridPane.setConstraints(usageTimeField, 2, index + 1);

TextField numDaysField = new TextField();

numDaysField.setPromptText("Number of Days");

GridPane.setConstraints(numDaysField, 3, index + 1);

Label consumptionLabel = new Label();

consumptionLabels.add(consumptionLabel);

GridPane.setConstraints(consumptionLabel, 4, index + 1);

Label costLabel = new Label();

costLabels.add(costLabel);

GridPane.setConstraints(costLabel, 5, index + 1);

final String applianceName = applianceNames.get(index);

grid.getChildren().addAll(nameLabel, numAppliancesField, usageTimeField, numDaysField, consumptionLabel, costLabel);

numAppliancesField.setOnKeyReleased(e -> {

try {

numberOfAppliances.set(index, Integer.parseInt(numAppliancesField.getText()));

} catch (NumberFormatException | IndexOutOfBoundsException ignored) {

// Handle the exception or ignore it

}

});

usageTimeField.setOnKeyReleased(e -> {

try {

usageTimes.set(index, Double.parseDouble(usageTimeField.getText()));

} catch (NumberFormatException | IndexOutOfBoundsException ignored) {

// Handle the exception or ignore it

}

});

numDaysField.setOnKeyReleased(e -> {

try {

numberOfDays.set(index, Integer.parseInt(numDaysField.getText()));

} catch (NumberFormatException | IndexOutOfBoundsException ignored) {

// Handle the exception or ignore it

}

});

}

private void calculateTotalBill() {

double totalCost = 0;

for (int i = 0; i < applianceNames.size(); i++) {

double powerRating = powerRatings.get(i);

int numAppliances = numberOfAppliances.get(i);

double usageTime = usageTimes.get(i);

double electricityRate = electricityRates.get(i);

int numDays = numberOfDays.get(i);

double electricityConsumption = calculateElectricityConsumption(powerRating, numAppliances, usageTime, numDays);

double electricityCost = calculateElectricityCost(electricityConsumption, electricityRate);

totalCost += electricityCost;

// Update the labels for each appliance

consumptionLabels.get(i).setText("Consumption: " + electricityConsumption + " kWh");

costLabels.get(i).setText("Cost: Rs. " + electricityCost);

}

totalCostLabel.setText("Total Electricity Bill for all appliances: Rs. " + totalCost);

}

private double calculateElectricityConsumption(double powerRating, int numAppliances, double usageTime, int numDays) {

return powerRating \* numAppliances \* usageTime \* numDays;

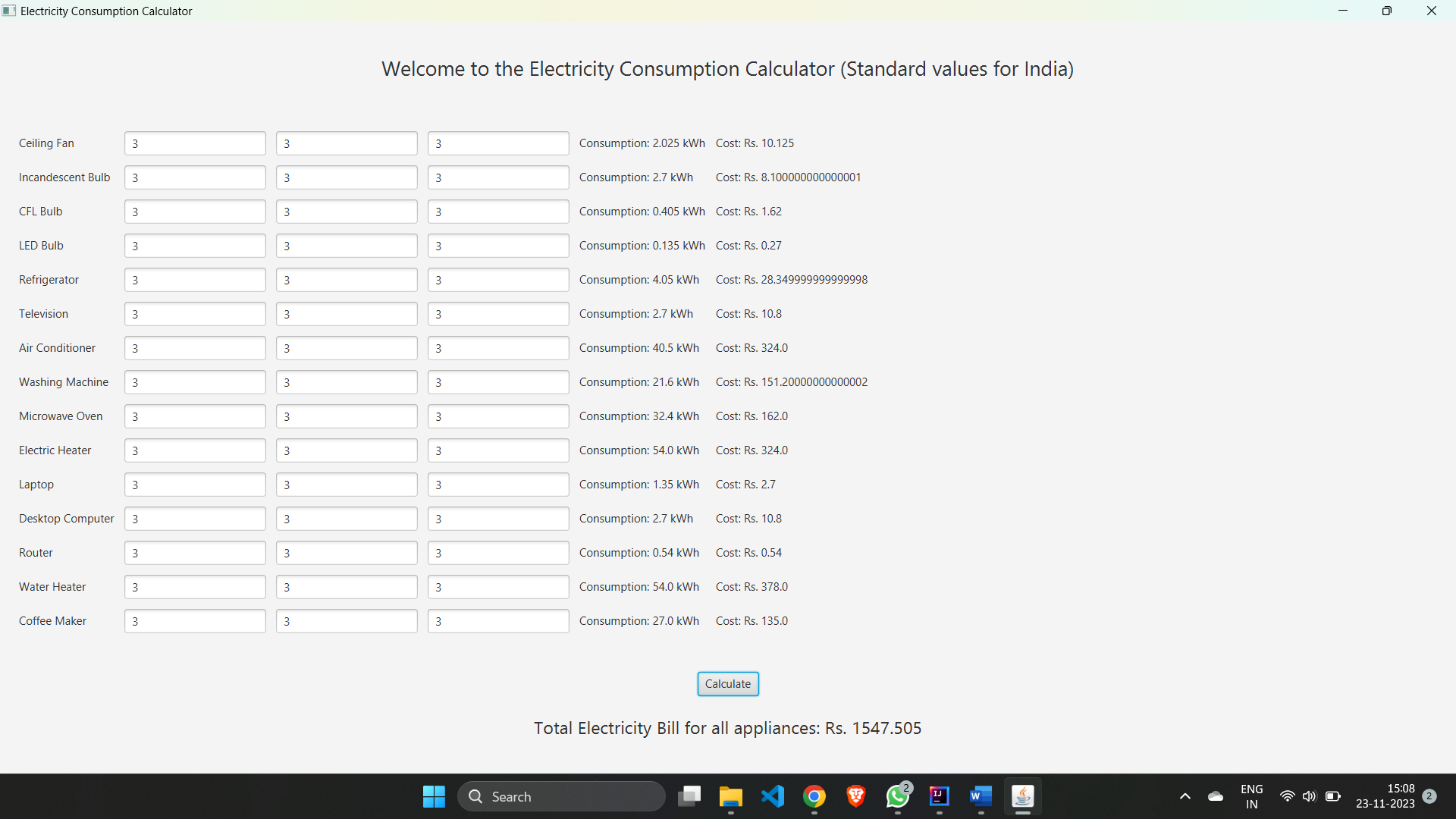
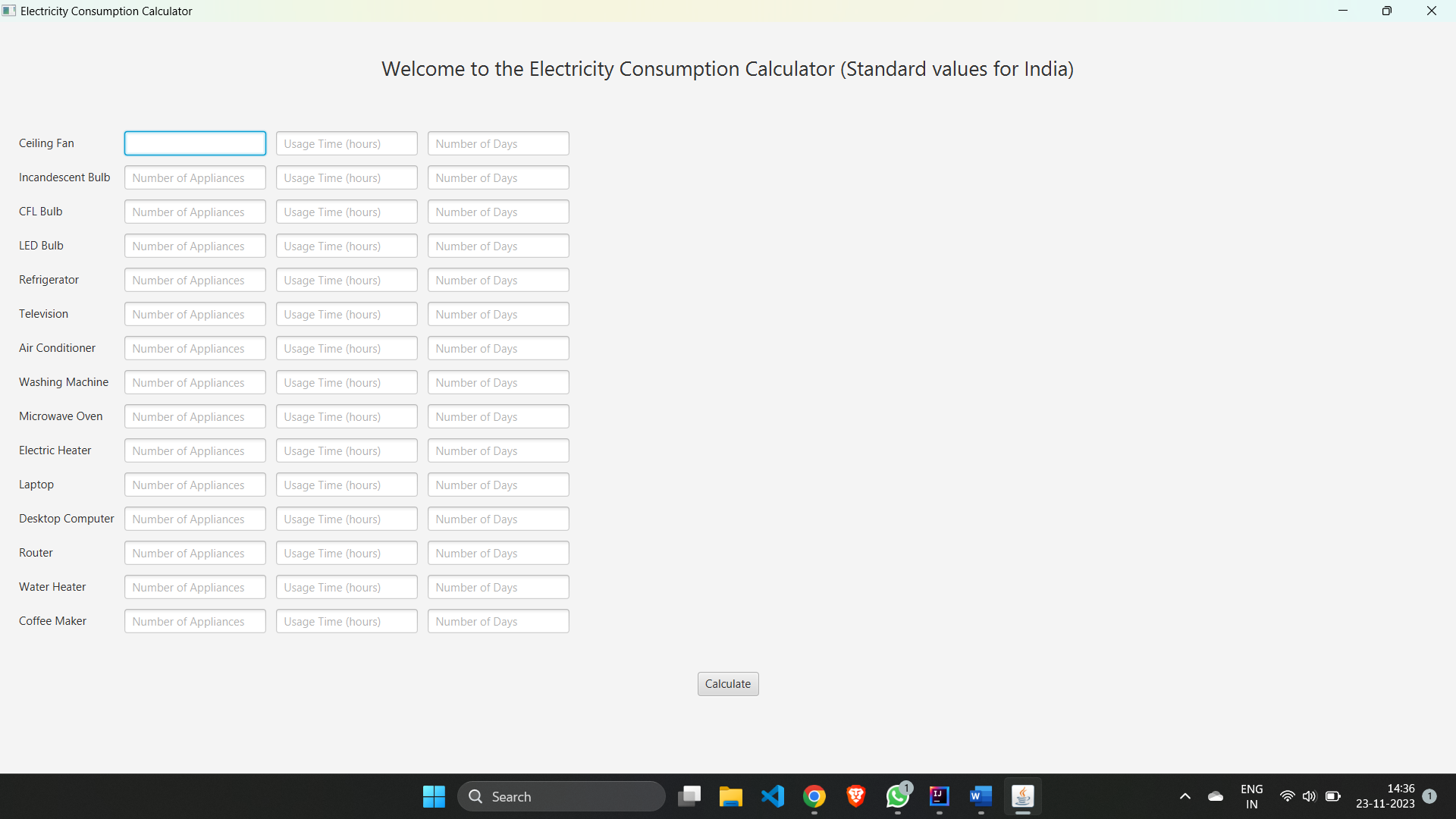
}

private double calculateElectricityCost(double electricityConsumption, double electricityRate) {

return electricityConsumption \* electricityRate;

}}

**Screenshots of the User Interface:**

****

# BIBLOGRAPHY

|  |  |  |
| --- | --- | --- |
| **1.** | JAVA Language | https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html |
| **2.** | JAVA FX (JFX) | https://docs.oracle.com/javafx/2/ |