



TEAM DIONYSUS FINAL PROJECT DOCUMENTATION

TECHVOLT

Submitted by:

Enriquez, John Paolo E.

Polido, Gabriel

Ramos, Andriel Dinn A.

I. INTRODUCTION

The Energy Consumption Tracker is a cutting-edge project that emerges as a crucial solution to address the global demand for affordable and clean energy, in perfect alignment with Sustainable Development Goal 7: Affordable and Clean Energy. With the world facing pressing environmental challenges due to rising energy demands and greenhouse gas emissions, this application represents a significant step towards a more sustainable future. By providing a user-friendly platform, the Energy Consumption Tracker empowers individuals and communities to take charge of their energy consumption patterns and actively participate in creating a greener planet.

At its core, the Energy Consumption Tracker offers a comprehensive tool that allows users to effortlessly monitor and manage their energy usage. Through an intuitive interface, users can input their energy data, track usage trends over time, and set personalized energy-saving goals. The application utilizes advanced data visualization and analysis techniques to present energy usage patterns in a visually appealing manner, making it easier for users to understand and interpret their consumption behavior. This real-time insight empowers individuals to make informed decisions about their energy usage, promoting a culture of responsible energy consumption. In addition to facilitating data tracking and analysis, the Energy Consumption Tracker goes beyond being a mere monitoring tool. It actively engages with users by providing personalized recommendations on how to reduce energy consumption and adopt more sustainable practices. By suggesting energy-efficient alternatives, such as utilizing energy-efficient appliances or adopting renewable energy sources, the application guides users towards greener choices, which, in turn, contributes to reducing their carbon footprint and fostering a more sustainable environment.

One of the key strengths of the Energy Consumption Tracker lies in its potential to foster a sense of community engagement and collective responsibility for energy conservation. The application offers social features that enable users to compare their energy consumption with that of their peers or community averages. This fosters healthy competition and a sense of camaraderie among users, encouraging them to strive for more significant energy-saving achievements collectively. By creating a platform for knowledge-sharing and best practices, the Energy Consumption Tracker builds a strong network of environmentally-conscious individuals who actively work towards a common goal of reducing energy consumption and promoting sustainability. As the Energy Consumption Tracker gains popularity and attracts a growing user base, its impact will ripple through various sectors. Governments and policymakers can leverage the anonymized data collected by the application to develop more targeted energy policies and incentive programs. Utility companies can utilize the insights to optimize energy distribution and plan for peak demand periods more efficiently. Ultimately, this application has the potential to not only empower individuals but also play a significant role in driving systemic change toward a future where affordable and clean energy is a reality for all. With its user-friendly interface, data-driven insights, and community-oriented approach, the Energy Consumption Tracker emerges as a powerful tool to shape a greener and more sustainable world for generations to come.

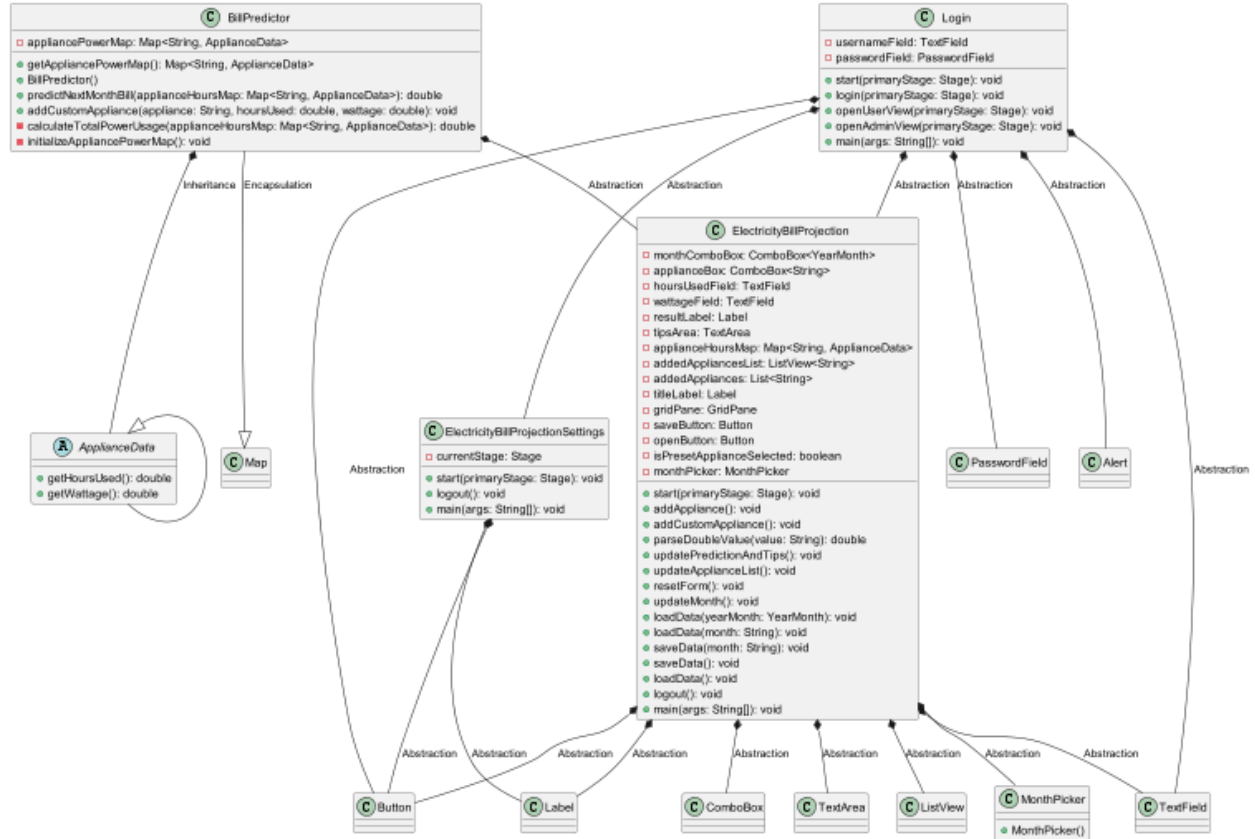
II. PROPOSED APPLICATION

TechVolt, the Energy Consumption Tracker, is an ambitious project designed to revolutionize the way users interact with their energy consumption data and empower them to contribute to a sustainable future. The three-tier architecture, consisting of the client, server, and database components, ensures a seamless and efficient flow of information between users and the application's backend.

At the client tier, the JavaFX-based user interface provides a responsive and user-friendly experience. Users can easily navigate the intuitive interface to input their energy data, set energy-saving goals, and receive personalized recommendations. The interface's user-centric design fosters engagement and encourages users to actively participate in energy conservation efforts. With a user-friendly platform at their fingertips, users can take charge of their energy consumption patterns and make informed decisions to reduce their environmental impact. The server tier, built using Java EE, plays a pivotal role in handling user requests and processing data. It manages user authentication and login processes to ensure secure access to sensitive energy consumption information. Additionally, the server processes energy data input, visualizes data trends, and provides real-time goal tracking and recommendations. Through this intelligent processing, users gain valuable insights into their energy usage, empowering them to optimize their consumption patterns and minimize waste. The heart of TechVolt lies in the database tier, powered by MySQL. This tier securely stores crucial user details, energy consumption data, and personal goals. By maintaining a central repository of user information, TechVolt can deliver seamless experiences across devices and enable users to access their energy data from anywhere, anytime. The secure and scalable nature of the database ensures that as the user base grows, the application can efficiently manage the influx of data while safeguarding user privacy.

The key classes depicted in the UML diagram - User, EnergyData, Goal, and SmartMeter - embody the core functionalities of TechVolt. The User class enables account creation and login, ensuring a personalized experience for each user. The EnergyData class empowers users to input and retrieve their energy consumption data, fostering transparency and understanding of their usage patterns. The Goal class enables users to set, track, and be notified of their energy-saving goals, incentivizing proactive energy conservation. Additionally, if applicable, the SmartMeter class provides real-time energy data, further enhancing the application's accuracy and responsiveness. The Input-Process-Output (IPO) chart demonstrates the functionality of the BillPredictor class within TechVolt. By taking appliance usage hours and power ratings as input, the BillPredictor class can accurately predict the user's next month's electricity bill. This prediction empowers users to anticipate their expenses and proactively implement energy-saving practices to achieve cost reductions. Furthermore, the TipsGenerator class complements the prediction by generating personalized tips for reducing electricity consumption. These tips offer actionable advice to users, enabling them to adopt sustainable practices effectively.

III. IMPLEMENTATION/OOP ASPECTS



(TechVolt PUML)

a. Explanation

The system is composed of several classes representing different components. The main class is "BillPredictor," which contains methods for predicting the next month's electricity bill based on the usage of various appliances. It has a private field called "appliancePowerMap," which is a map that associates the names of appliances with their power consumption data represented by the abstract class "ApplianceData." The "BillPredictor" class also has methods to add custom appliances and calculate the total power usage based on the provided data.

The "ElectricityBillProjection" class is responsible for the user interface, and it interacts with the "BillPredictor" to provide predictions. It has various UI components such as ComboBoxes, TextFields, Labels, and Buttons for input and display purposes. Users can add appliances, input their hours of usage and wattage, and receive predictions for the next month's bill. There are also methods to load and save data, as well as a "MonthPicker" class to handle selecting months. Additionally, there is a "Login" class that likely represents the authentication process for accessing the system, with fields for username and password.

The associations between classes represent the relationships between different components. For example, "BillPredictor" uses a map data structure ("Encapsulation") to store appliance data, and it has a "has-a" relationship with "ElectricityBillProjection" for abstraction purposes. The "ApplianceData" class serves

as a base class ("Inheritance") for storing common attributes of appliances, and specific appliances extend from it. The other associations depict how UI components are used in the "ElectricityBillProjection" and "Login" classes for handling user interactions.

b. OOP Concepts Used

The PlantUML diagram illustrates the Object-Oriented Programming (OOP) aspects utilized in an application. The diagram consists of several classes representing different entities in the system. These classes demonstrate the concept of abstraction, where each class represents a real-world entity or a specific concept in the application domain. Additionally, the diagram employs inheritance, indicated by the arrow notation <|--, where the abstract class `ApplianceData` is used as a base class, potentially serving as a blueprint for other classes that extend its functionality.

Moreover, the diagram showcases the principle of encapsulation in the `BillPredictor` class. By defining private fields, such as `appliancePowerMap`, and providing public methods like `getAppliancePowerMap()`, `predictNextMonthBill()`, and `addCustomAppliance()`, the class controls access to its internal state, ensuring data integrity and security.

Furthermore, the diagram represents association relationships between classes using the *-- notation. For instance, the `BillPredictor` class is associated with the `ElectricityBillProjection` class, implying that `BillPredictor` utilizes `ElectricityBillProjection` in some way.

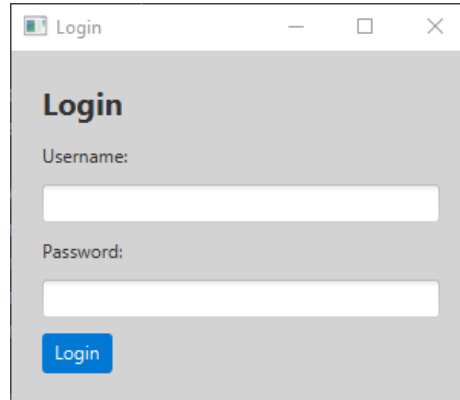
Additionally, the diagram incorporates composition, indicated by the --|> notation, where the `BillPredictor` class has a composition relationship with the `Map` class. This means that `BillPredictor` contains and manages a `Map` instance.

The diagram does not explicitly show polymorphism, but it is a crucial aspect in OOP. Polymorphism allows objects of different classes to be treated as objects of a common superclass, facilitating code reuse and flexibility through method overriding and interfaces.

The diagram also uses access modifiers, such as + for public visibility and - for private visibility, to denote the visibility of class members (attributes and methods). For example, `+BillPredictor()` indicates a public constructor for the `BillPredictor` class.

IV. WALKTHROUGH/DATA/RESULTS

a. *Login*

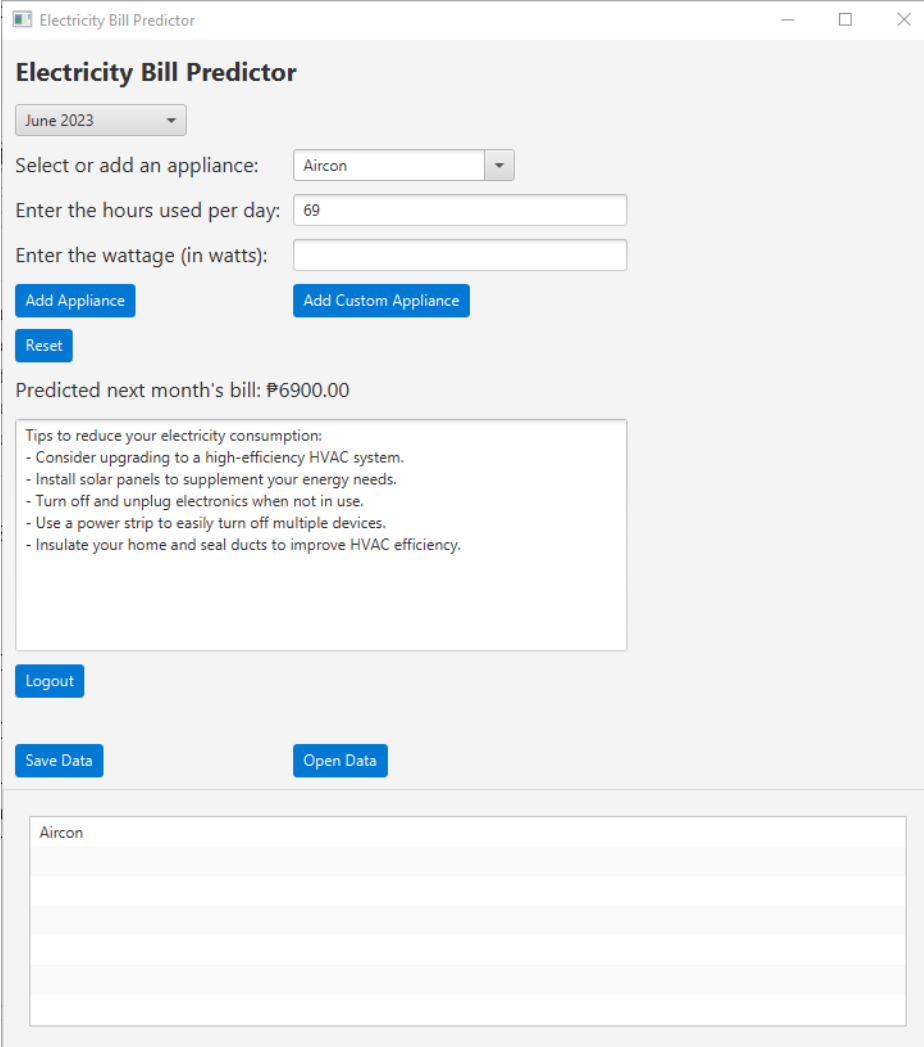


(TechVolt Login Screen)

The Java class "Login" is a part of a JavaFX application used for handling user authentication and login functionality. It extends the "Application" class, making it an entry point for the JavaFX application. The class provides a graphical user interface (GUI) that allows users to enter their credentials (username and password) and then validates them to grant access to different views based on their roles (user or admin). In the "start" method, the GUI elements are constructed using JavaFX components such as Labels, TextFields, and Buttons, and they are organized in a VBox layout. The UI design includes a title, "Login," and two input fields for username and password, along with a login button. When the "Login" button is clicked, the "login" method is invoked.

The "login" method retrieves the values entered by the user in the username and password fields and checks them against hardcoded credentials for a user and an admin. If the entered credentials match those of the user, the application opens the user view by calling the "openUserView" method, which closes the current stage and opens a new "ElectricityBillProjection" stage. If the entered credentials match those of the admin, the application opens the admin view by calling the "openAdminView" method, which closes the current stage and opens a new "ElectricityBillProjectionSettings" stage. If the credentials do not match either, an error message is displayed in the form of an alert dialog.

b. ElectricityBillCalculator



The screenshot shows a Java Swing window titled "Electricity Bill Predictor". The interface includes a month selection dropdown set to "June 2023". Below this, there's a section for adding appliances with a label "Select or add an appliance:" followed by a dropdown menu currently showing "Aircon". There are input fields for "Enter the hours used per day:" (containing "69") and "Enter the wattage (in watts):". Below these are three buttons: "Add Appliance", "Add Custom Appliance", and "Reset". The application displays a "Predicted next month's bill: ₱6900.00". A text box contains "Tips to reduce your electricity consumption:" followed by a list of five suggestions. At the bottom, there are "Logout", "Save Data", and "Open Data" buttons. A table at the very bottom lists the added appliance "Aircon" with empty rows for additional entries.

(TechVolt Electricity Bill Calculator Screen)

The Electricity Bill Projection application is a sophisticated and user-centric Java program that empowers users to effectively manage their electricity consumption and make well-informed predictions about their monthly bills. This powerful tool is composed of several interlinked Java classes that work harmoniously to create a seamless and efficient user experience. These classes include `ApplianceData`, `BillPredictor`, `ElectricityBillProjection`, `MonthPicker`, and `TipsGenerator`, each playing a unique and critical role in the application's functionality.

The foundation of the application lies in the `ApplianceData` class, which acts as a data container for individual electrical appliances. By storing vital information about each appliance, such as its daily usage hours and wattage, this class allows users to define specific appliances with their respective usage

data. Through instances of the `ApplianceData` class, users can build a comprehensive and personalized profile of their appliances, setting the stage for accurate and detailed electricity consumption projections.

The core functionality of the application is facilitated by the robust `BillPredictor` class. Serving as the predictive engine, this class harnesses the information stored in the `ApplianceData` instances to calculate the total power usage and project the electricity bill for the upcoming month. By skillfully employing algorithms and mathematical computations, the `BillPredictor` class ensures precise and reliable predictions based on a user's selected appliances and their corresponding usage patterns. Moreover, it offers additional features such as adding custom appliances, initializing a map of preset appliances and performing the necessary calculations for bill projection.

The `ElectricityBillProjection` class is the pivotal JavaFX application class responsible for creating the intuitive and interactive graphical user interface (GUI). This class enables users to effortlessly interact with the application, presenting them with a wide array of functionalities. Through the GUI, users can seamlessly select from a diverse list of preset appliances or input their custom appliances and usage data. Upon entering the required information, users can instantly access a clear and comprehensive view of the projected electricity bill for the upcoming month. Additionally, the `ElectricityBillProjection` class integrates essential features for saving and loading data for different months, facilitating effortless tracking of electricity usage over time and empowering users to make well-informed decisions about their consumption habits.

The `MonthPicker` class further enhances the user experience by providing a custom JavaFX control that extends `ComboBox<YearMonth>`. This ingenious class simplifies the process of selecting a specific month, presenting users with an aesthetically pleasing and user-friendly interface that displays formatted date strings such as "January 2023." With its seamless integration into the application's workflow, the `MonthPicker` class ensures that users can easily and conveniently select their desired month within a predefined range, optimizing their overall experience.

Beyond electricity consumption projections, the `TipsGenerator` class elevates the application's utility by generating personalized energy-saving tips based on the predicted electricity bill. Taking into account the value of the projected bill, the `TipsGenerator` class crafts tailored recommendations to optimize energy usage and reduce overall costs. By offering a range of actionable strategies, from powering off and unplugging idle devices to investing in energy-efficient appliances and lighting solutions, this class empowers users to actively participate in conserving energy and adopting eco-friendly practices.

V. CONCLUSION AND FUTURE WORK

The Energy Consumption Tracker project represents a crucial step in addressing the pressing need for affordable and clean energy solutions worldwide. As we continue to witness the impacts of climate change and resource depletion, the importance of transitioning towards sustainable practices becomes increasingly evident. This project's focus on providing a comprehensive platform for users to monitor, analyze, and optimize their energy consumption is a significant contribution to the global efforts in achieving Sustainable Development Goal 7 (SDG 7) - ensuring access to affordable, reliable, sustainable, and modern energy for all.

At the heart of the Energy Consumption Tracker lies its user-centric approach, which empowers individuals to take control of their energy usage. By enabling user registration and login functionalities, the project fosters engagement and ownership, allowing users to create personalized profiles and set energy-saving goals tailored to their specific needs and preferences. This level of personalization is crucial in motivating users to actively participate in the platform and take actionable steps towards a more sustainable lifestyle.

The data input feature, which allows users to record and monitor their energy consumption, serves as the foundation for the entire platform's functionalities. By collecting and analyzing real-time energy usage data, users gain valuable insights into their consumption patterns, peak hours, and areas of inefficiency. Such granular information can lead to a deeper understanding of energy usage behaviors and facilitate informed decision-making. To enhance the user experience, the Energy Consumption Tracker leverages data visualization and analysis tools. These tools present energy consumption data in clear and visually appealing formats, making it easier for users to comprehend and identify trends or anomalies. Visual representations of data can significantly influence behavior, driving users to adopt energy-saving habits through increased awareness of their consumption patterns. In addition to data analysis, the platform offers energy-saving tips and personalized recommendations. These recommendations are based on the analysis of individual energy usage data and can vary depending on the user's lifestyle, location, and other relevant factors. By providing customized suggestions, the Energy Consumption Tracker empowers users to make incremental changes and optimize their energy consumption effectively.

The project's comparative analysis feature allows users to compare their energy consumption with peers or similar households. This aspect fosters a sense of healthy competition and social responsibility, as users can see how their consumption measures up to others and strive to achieve energy efficiency on par with the best practices in their community. Such social comparison can drive behavioral changes and create a ripple effect of sustainable habits within society. The reporting capabilities of the Energy Consumption Tracker enable users to access detailed summaries of their energy usage over time. These reports provide valuable feedback on progress towards energy-saving goals and highlight areas where further improvements can be made. Regular updates and progress tracking can help reinforce positive behaviors and motivate users to maintain their commitment to energy conservation.

As technology continues to evolve, the Energy Consumption Tracker looks to the future by exploring integration with smart meters or IoT devices. This forward-thinking approach allows for real-time data capture, eliminating the need for manual input and enhancing the accuracy and convenience of the platform. Smart meters and IoT devices enable seamless data transmission, enabling users to access up-to-date information and insights, thus encouraging more proactive energy management.

The project's potential integration with emerging technologies does not stop at smart meters and IoT devices. With the advent of artificial intelligence and machine learning, the Energy Consumption Tracker could leverage advanced algorithms to continuously improve its energy analysis and recommendations. AI-powered predictive models could forecast future energy consumption patterns and suggest preemptive measures to optimize energy usage, making the platform even more effective and valuable to users. To maximize the impact of the Energy Consumption Tracker, partnerships with

government agencies, utilities, and environmental organizations become critical. Collaborating with such stakeholders can lead to valuable data exchanges, access to policy support, and the potential to scale the project's reach nationwide or even globally. Government incentives for energy conservation, financial support, and access to subsidies for energy-efficient upgrades could be integrated into the platform to further incentivize users and create a more extensive network of sustainable energy advocates.

In the long run, the Energy Consumption Tracker has the potential to extend its scope beyond individual users to include industries, commercial establishments, and large-scale energy consumers. By providing tailored solutions for businesses and organizations to monitor and optimize their energy consumption, the platform can significantly impact overall energy efficiency and promote sustainable practices across different sectors. Furthermore, exploring partnerships with renewable energy providers and integrating renewable energy tracking features could be a game-changer. By incorporating data from solar panels, wind turbines, and other renewable sources, users can not only monitor their consumption but also track their contributions to the grid and their overall carbon footprint. This integration aligns perfectly with the global push towards renewable energy adoption and allows users to actively participate in the global effort to combat climate change. To foster a culture of sustainability and community engagement, the Energy Consumption Tracker could establish forums or social platforms within the application. These spaces would enable users to connect with like-minded individuals, share energy-saving tips, success stories, and challenges faced during their sustainable journey. Building a vibrant community of users can create a sense of collective responsibility and encourage continuous learning and improvement.

VI. CONTRIBUTIONS (Individual contributions)

Member	Contribution
Enriquez, John Paolo E.	<ul style="list-style-type: none">- Project Documentation- Project Proposal- Code- Poster- Video Presentation
Polido, Gabrie	<ul style="list-style-type: none">- Project Documentation- Project Proposal- Code- Poster- Video Presentation
Ramos, Andriel DInn A.	<ul style="list-style-type: none">- Project Documentation- Project Proposal- Code- Poster- Video Presentation

VII. REFERENCES

April 2023 rates updates. Meralco. (n.d.-a).
<https://company.meralco.com.ph/news-and-advisories/april-2023-rates-updates-0>
Electricity bill program in Java - Javatpoint. www.javatpoint.com. (n.d.).
<https://www.javatpoint.com/electricity-bill-program-in-java>
GeeksforGeeks. (2023, April 4). Program to Calculate electricity bill. GeeksforGeeks.
<https://www.geeksforgeeks.org/program-to-calculate-electricity-bill/>
JpixtaJpixta, & Galen NareGalen Nare . (1959, November 1). How to code a very simple login system with Java. Stack Overflow.
<https://stackoverflow.com/questions/16627910/how-to-code-a-very-simple-login-system-with-java>
March 2023 rates updates. Meralco. (n.d.-b).
<https://company.meralco.com.ph/news-and-advisories/march-2023-rates-updates-0#:~:text=5453>
Mercurio, R. (2023, May 11). Meralco increases electricity rates in May. Philstar.com.
<https://www.philstar.com/headlines/2023/05/12/2265681/meralco-increases-electricity-rates-may>
MK, W.-. (2023, July 24). Java program to Calculate Electricity Bill: Example. Learn Java.
<https://javatutoring.com/calculate-electricity-bill-java-program/>

Suresh. (2022, August 28). Java program to Calculate electricity bill. Tutorial Gateway.
<https://www.tutorialgateway.org/java-program-to-calculate-electricity-bill/>
User authentication program in Java. Sanfoundry. (2022, May 23).
<https://www.sanfoundry.com/java-program-illustrate-how-user-authentication-done/>