# Software Requirements Specification

## for

## Electricity Billing System

**Version <1.0>**

**Group No.:** 2

| | |
|---|---|
| **Mohammed Yousef Mohammed Abdulkarem** | **1221305727** |
| **Mohammed Aamena Mohammed Abdulkarem** | **1221305728** |
| **Muhammad Faiz bin Ilyasa** | **1221309435** |
| **Chua Cheng Zong** | **1221309173** |

**Date:** 08 DEC 2024

# Contents

# Revision

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Draft Type and Number | Full Name | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 00/00/00 |

# Project Introduction

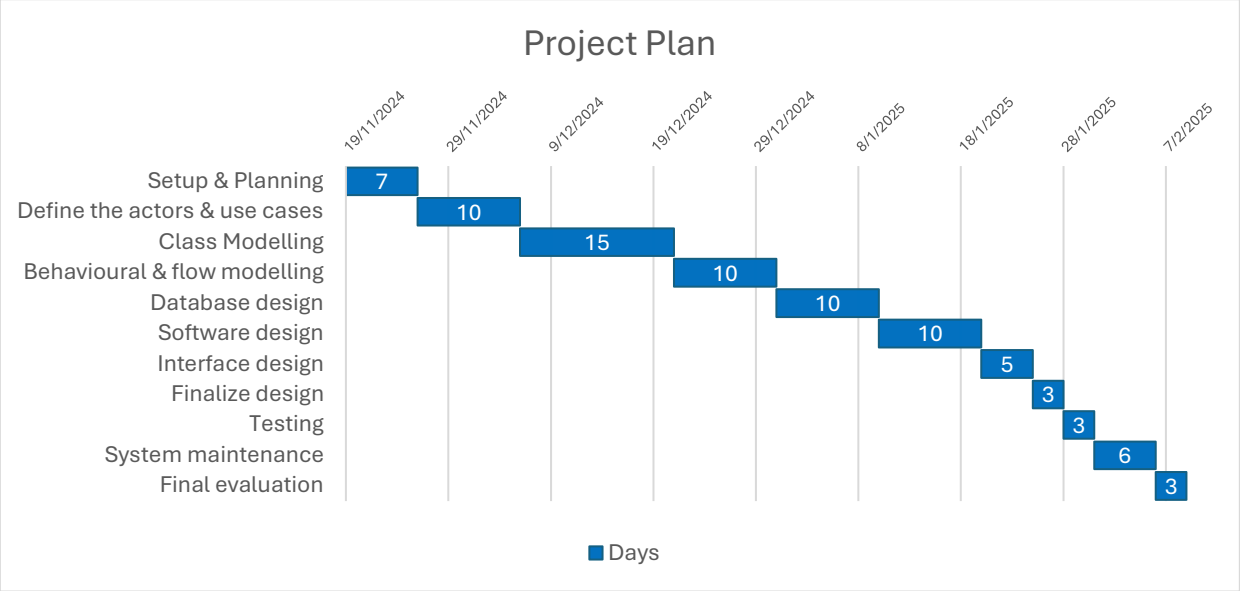
## Team Members

| Name | Actor/Processes |
|---|---|
| Mohammed Yousef Mohammed Abdulkarem | **Customer**<br>Register, Login, Reset Password, Update Personal Details, View Bills, Make Payments, Submit Issues, and Submit Feedback. |
| Mohammed Aamena Mohammed Abdulkarem | **Support Admin**<br>Login, View Bill Details, and Resolve Customer Issues. |
| Muhammad Faiz bin Ilyasa | **Staff**<br>Login, Track Overdue Bills, and Usage Monitoring. |
| Chua Cheng Zong | **Utility Providers**<br>Login, Update Meter Readings and Tariffs, and Generate Monthly Bills. |

## Problem statement

Current approach of managing electricity lacks in terms of efficiency and accuracy, which could cause errors in meter readings, delays, and ineffective tracking of payment activities. This can lead to customers facing difficulties in paying their monthly bills, while the system's staff struggle in monitoring system performance and provide support to troubled customers. An automated electricity bill system with improved functionalities ensures user experience is enhanced while allowing for easier management for the staff.

# Project Plan




| Tasks | Start Date | Days Needed | Completion |
|---|---|---|---|
| Setup & Planning | 19/11/2024 | 7 | 26/11/2024 |
| Define the actors & use cases | 26/11/2024 | 10 | 6/12/2024 |
| Class Modelling | 6/12/2024 | 15 | 21/12/2024 |
| Behavioural & flow modelling | 21/12/2024 | 10 | 31/12/2024 |
| Database design | 31/12/2024 | 10 | 10/1/2025 |
| Software design | 10/1/2025 | 10 | 20/1/2025 |
| Interface design | 20/1/2025 | 5 | 25/1/2025 |
| Finalize design | 25/1/2025 | 3 | 28/1/2025 |
| Testing | 28/1/2025 | 3 | 31/2/2025 |
| System maintenance | 1/2/2025 | 6 | 6/2/2025 |
| Final evaluation | 6/2/2025 | 3 | 9/2/2025 |

# System Overview

## Description

The Electricity Billing System provides an integrated, scalable solution for automating and optimizing electricity service management. It supports smooth interaction among customers, utility providers, administrative staff, and payment processors to ensure smooth communication and seamless operations. Key features include user registration, secure authentication, accurate billing, payment processing, customer support, and feedback management. By prioritizing data integrity, robust security, and operational efficiency, the system delivers a user-friendly platform that ensures reliability and effectiveness in managing electricity services, setting a high standard for streamlined and efficient service delivery.

## Actors

| Actor | Use Cases |
|---|---|
| **Customer** | Register, Login, Reset Password, Update Personal Details, View Bill Details, Make a Payment and Receive Payment Receipt, Submit Feedback, and Submit Issues. |
| **Utility Providers** | Login, Update Meter Readings and Tariffs, and Generate Monthly Bills. |
| **Support Admin** | Login, View Bill Details, Submit and Resolve Customer Issues. |
| **Staff** | Login, Track Overdue Bills, and Usage Monitoring. |

## Assumptions and Dependencies

### Assumptions:

**User Access and Connectivity**

- Users will have access to devices capable of running modern web browsers and will have stable internet connectivity.
- Customers will provide valid email addresses, meter number and mobile numbers for notifications and account management.

**Data Availability and Accuracy**

- Utility providers will provide accurate and up-to-date meter readings and tariffs for accurate billing.
- Customers are expected to provide valid personal information.
- The system will automatically check the input data for validation, such as meter readings and customer information, to minimize errors in billing.

**Scalability Requirements**

The system will support up to 1000 users initially. As the user base grows, scaling the system will require infrastructure upgrades, such as database optimization and increased server capacity, to ensure continued performance and reliability.

**Compliance with Standards**

The system will comply with relevant legal and industry regulations, including:

- Data Privacy Laws: The system will follow GDPR to ensures that the system is in compliance with data protection regulations, protecting personal data and the privacy rights of users. This includes secure data storage, processing, and consent management to avoid legal violations.

- Payment Security Standards: Ensures the realization of compliance with the Payment Card Industry Data Security Standard (PCI DSS) for the safe handling of credit card information, helping to protect payment data from fraud or breaches during transactions.
- Utility Service Requirements: Ensures that the system meets regional standards and regulations of electricity services provided, regarding billing accuracy, service reliability, and customer protection.

**Dependencies:**

**Integration with Payment Gateways**

The system payment functionality relies on secure and reliable third-party payment services to process transactions efficiently while maintaining industry standards for data protection.

**Accurate Tariff and Meter Data**

The system's billing accuracy is dependent on regular updates of tariff rates and meter readings provided by utility providers. Any delays or inaccuracies in this data will directly affect billing and payment processes.

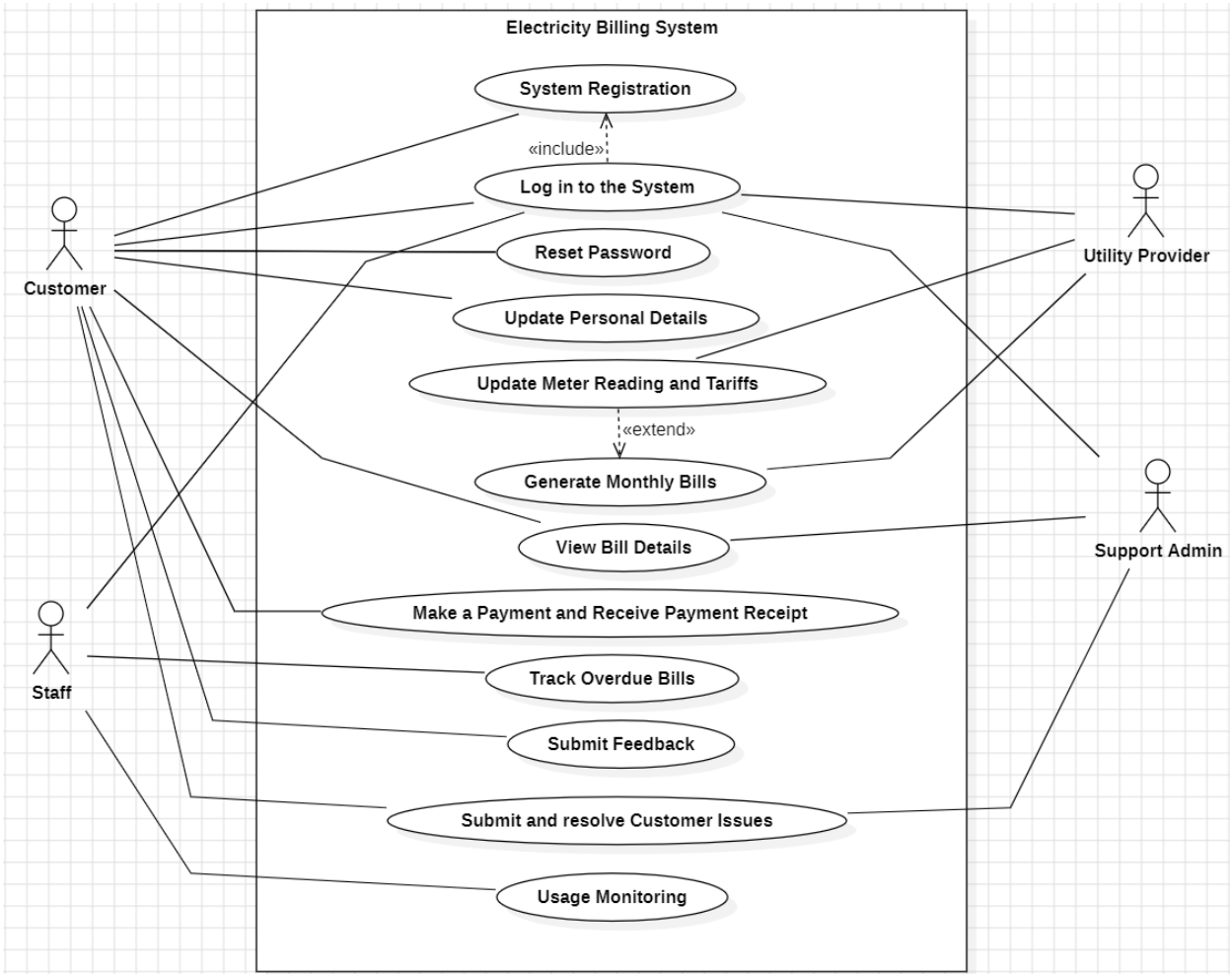
**Database Management System**

MySQL, a robust and scalable relational database system, will manage all system data effectively. It ensures secure storage, fast retrieval, and the capability to scale as the system grows, supporting seamless operations and reliability.

**Ongoing Maintenance and Support**

Continuous system updates, performance monitoring, and regular maintenance will be essential to address emerging needs, improve functionality and support best system performance.

# Use Case Diagram



# Scenario – Based Modeling

**Use Case 1: System Registration**

**Actors:**

- **Primary Actor**: Customer

**Description**:

The system allows new customers to register by providing their meter ID to connect the account with and their personal information. Then the system will verify the provided information by sending a code to the customer's email it also checks with the system database for any duplication (e.g., email, meter number). The Customer account is created and stored in the system database if no duplication has been found. Upon successful registration, the user can easily access the system by logging in to the system.

**Preconditions:** N/A

**Main Flow:**

1. The customer navigates to the homepage and selects the "New Customer Registration" option.
2. The customer enters their meter ID.
3. The customer clicks the "Next" button, so the system validates the provided ID.
4. The customer enters the required details, including:
   - Name
   - Email
   - Phone number
   - Address
   - Username
   - Password
5. The customer clicks the "Register" button, which submits the registration form.
6. The system validates the provided information to ensure they are complete and correctly formatted.
7. The system checks for any duplicate accounts based on the email or meter number.
8. If no duplicate records are found, the system creates a user account and stores the customer information in the database.

**Postconditions**:

- A new customer account is successfully created and securely stored in the system database.

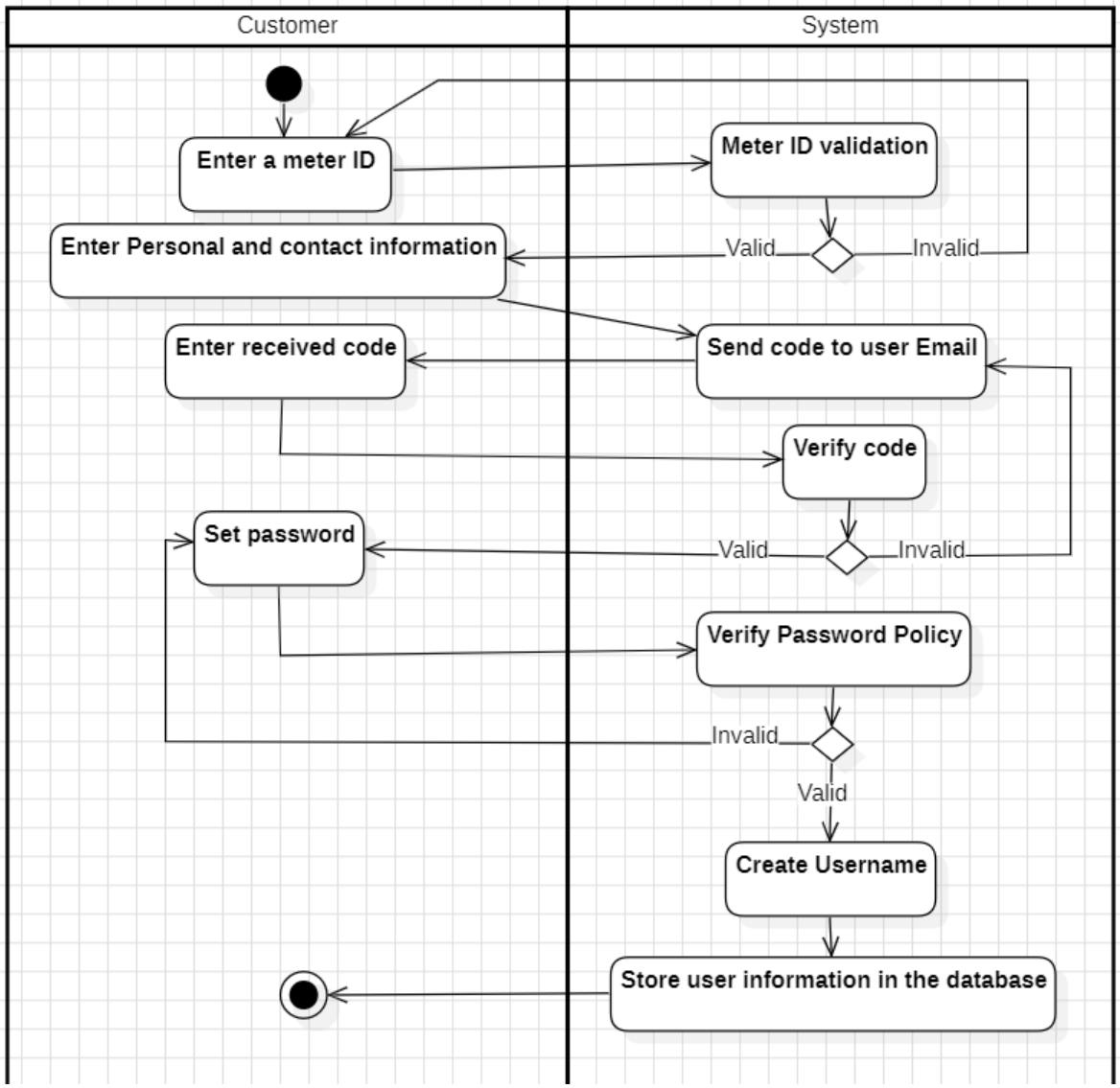
**Alternative Flows:**

- **Incomplete or Invalid Information:**
  - If required fields are incomplete or contain invalid data, the system displays an error message "Please complete all required fields or correct invalid entries".
  - The system marked the fields requiring correction for the customer's attention.
  - The customer updates the information and resubmits the registration form.

- **Duplicate Account Detection:**
  - If a duplicate account is detected based on the provided email or meter number, the system displays a message such as "An account with this email or meter ID already exists".
  - The customer is prompted to provide unique information again or use a different email or meter number to proceed.

**Assumptions:**

- Users have a stable internet connection while attempting to log in.
- The system is operational during the customer's registration attempt.
- The customer provides accurate and verifiable personal or organizational details for registration.




**Use Case 2: Log in to the System**

**Actors:**

- **Primary Actor:** Customer, Staff, Support Admin, Utility Provider.

**Description:**

This use case enables the user to log into the system using valid credentials. The system will open the respective System dashboard based on the user role, ensuring that users interact only with the needed functionalities based on their role.

**Preconditions:**

- The system and its database must be operational.
- The user must possess a registered account.

**Main Flow:**

1. The user accesses the system's login page.
2. The user inputs their registered username and password into the corresponding fields.
3. The user submits the login form by clicking the "Login" button.
4. The system verifies the provided credentials against the database.

5. If the credentials are correct. The user is redirected to their personalized dashboard or homepage, where features and options are tailored to their role.

**Postconditions:**

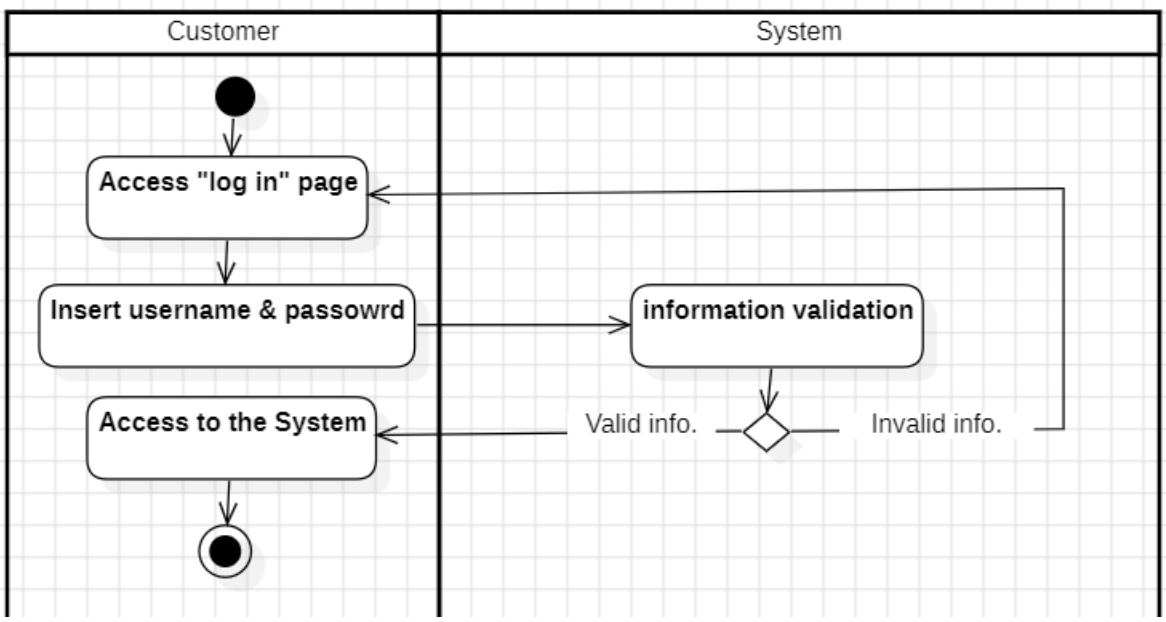
- The user successfully accesses the system with functionality appropriate to their role.
- A record of the login is stored in the system for tracking purposes.

**Alternative Flows:**

- **Invalid Credentials:**
  If the entered username or password is incorrect:
  - The system returns an error “Invalid username or password."
  - The user must re-enter their credentials.
- **Forgotten Password:**
  - Click the "Forgot Password" option.
  - Provide their registered email to receive password reset instructions.
  - Follow the reset process and attempt to log in again.

**Assumptions:**

- Users have a stable internet connection while attempting to log in.
- The authentication system and database are functioning correctly.
- Role-based permissions for users are accurately configured within the system.




**Use Case 3: Reset Password**

**Actors:**

- **Primary Actor:** Customer.

**Description:**

This use case enables the customer to reset their password either if they forget it or prefer to. The system will ask for the customer's email, so it sends the reset instructions to it. Further on the database will be updated so the customer can use their new password.

**Preconditions:**

- The system and its database must be operational.
- The user must possess a registered account.

**Main Flow:**

1- The customer selects the forget password option or navigates to the settings and selects the reset password option.
2- The system will ask the user to provide their email address, so it can send the instructions to it.
3- The customer will follow the instructions and set their new password.
4- The system will update the old password and store the new one in the database.

**Postconditions:**

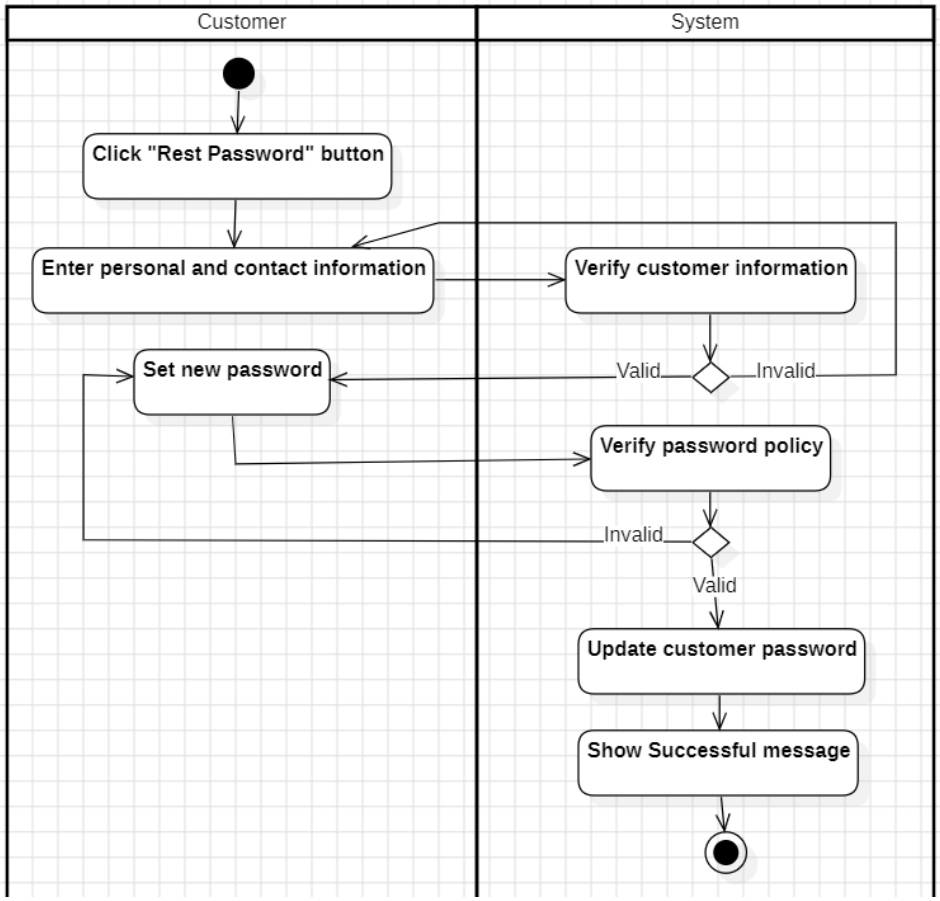
- The user successfully updated their password.
- A record of the operation is stored in the system for tracking purposes.

**Alternative Flows:**

- **If the new password matches the old one:**
  - The system will display an error message "Your new password can't be the same as the old one".
  - The customer must enter another password.

**Assumptions:**

- Customers have a stable internet connection while resetting their password.
- The customer provides a different password than their old one.



**Use Case 4: Update Personal Details**

**Actors:**

- **Primary Actor**: Customer
- **Secondary Actors**: Staff, Utility Provider, Payment Provider

**Description**:

The system allows users to update their own personal details from their own profile page. It then verifies the provided details to check for any errors (e.g. wrong name, email or phone number format, address data uncomplete, etc). If no errors are found, the user data will be updated. If errors are detected during the process, the system prompts the user to correct the identified issues.

**Preconditions:**

- The user must possess a valid customer account.
- The system and its database must be operational.

**Main Flow:**

1. The customer navigates to their profile page and selects the "Update Personal Details" option.

2. The customer enters the required details, including:
   - Meter number
   - Name
   - Email
   - Phone number
   - Address
   - Username
   - Password
3. Click “Save” to submit the changes.
4. The system validates the provided information to ensure they are complete and correctly formatted.
5. The system also checks for any duplicate accounts based on the email or meter number.
6. If successful, the system will show a success message, such as “Your personal details have been updated” and the database will be updated with the new details.
7. An email is sent to the customer to notify them about the successful personal detail update.

**Postconditions**:

- The user’s details are successfully updated and securely stored in the system database.
- The customer receives a confirmation email to notify them about the change.

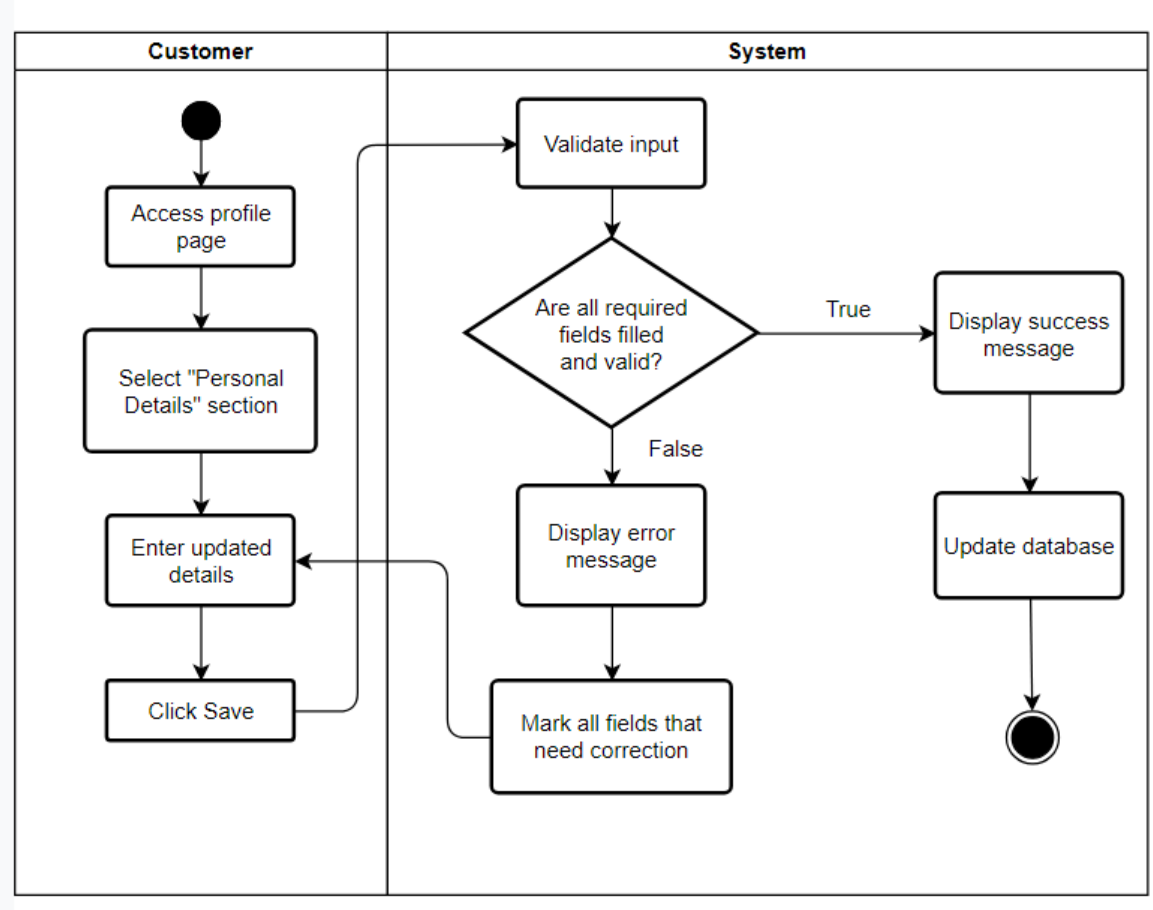
**Alternative Flows:**

**Incomplete or Invalid Information:**

- If required fields are incomplete or contain invalid data, the system displays an error message "Please complete all required fields or correct invalid entries".
- The system marked the fields requiring correction for the customer’s attention.
- The customer updates the information and resubmits the form.

**Assumptions:**

- The customer has a reliable internet connection to complete the registration process without interruptions.
- The system is operational during the customer's registration attempt.
- The customer provides accurate and verifiable personal or organizational details.
- Utility and payment provider services are available and accessible for any necessary validations during the process.




**Use Case 5: Update Meter Readings and Tariffs**

**Actors:**

- Primary Actor: Utility Provider
- Supporting Systems:

**Description:**

This use case enables the utility provider to update meter readings and tariffs. The system then validates the data, ensuring that the data given applies with the rules, then updates the database with the new readings and tariff rates.

**Preconditions:**

- The user must possess a utility provider account.
- The system and its database must be operational.

**Main Flow:**

1. The user accesses the "Meter Readings" or "Tariffs" page section.
2. The user enters updated readings or tariff rates for the specified period.
3. The user submits the updated data.
4. The system verifies the provided data.
5. If the credentials are correct:
   - The system updates database with the new provided data.
   - A success message will be displayed.

**Postconditions:**

- The user successfully updated the database with new reading or rates.

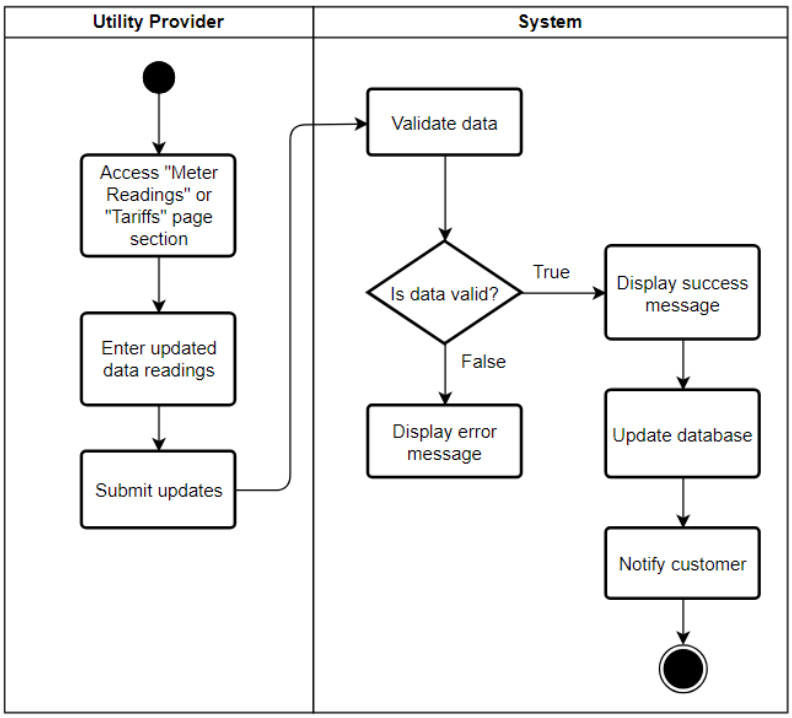
- A record of the update is stored in the system for tracking purposes.

**Alternative Flows:**

- **Invalid Credentials:**
  If the entered data is invalid:
  - The system returns an error message “Invalid data."
  - The user must enter the updated data again.

**Assumptions:**

- Users have a stable internet connection while attempting to log in.
- The authentication system and database are functioning correctly.
- The user possesses a utility provider account.




**Use Case 6: Generate Monthly Bills**

**Actors:**

- Primary Actor: Utility Provider
- Supporting Systems: Customer

**Description:**

This use case generates customer’s monthly bill. This system is automatically scheduled, it will generate a digital copy of the bill and attaches it to the customer’s account, the system then notifies customers via email or SMS.

**Preconditions:**

- The user must possess a valid customer account.
- The system and its database must be operational.

**Main Flow:**

1. On a prescheduled day each month, the system will process each customer’s monthly bill generation.
2. The system will retrieve the customer’s data on meter readings, tariffs, and payment history.
3. The system then calculates the bill amount based on usage and current tariff rates.

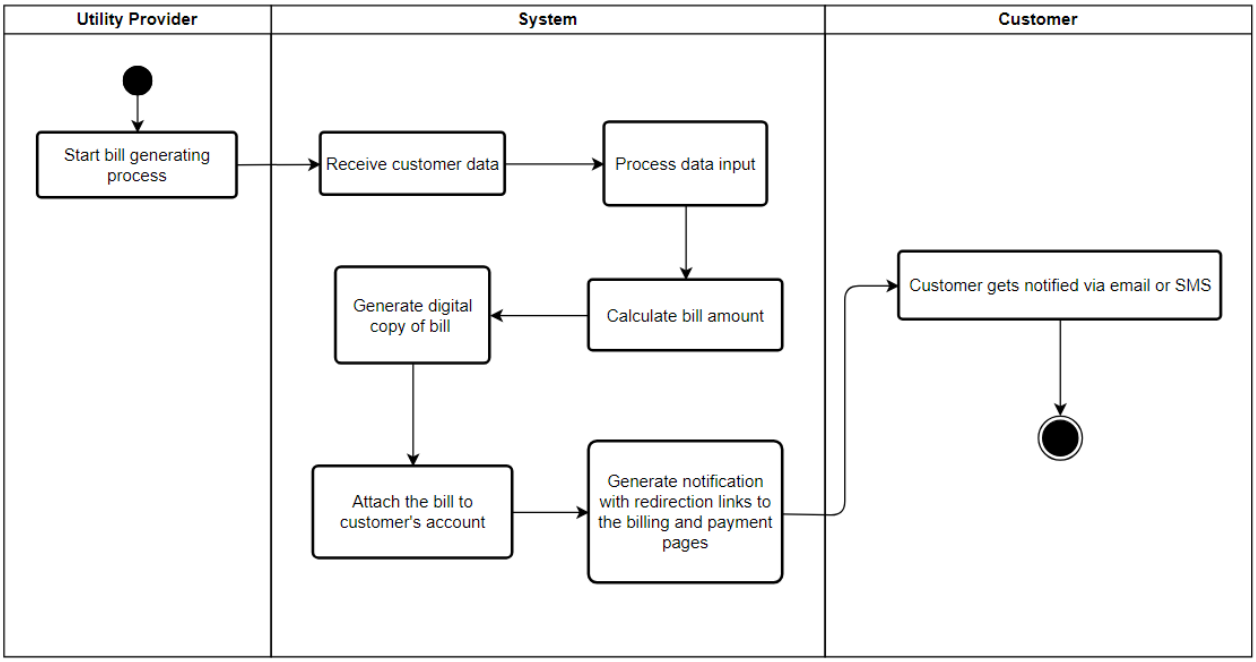
4. The system generates a detailed bill for each customer for the current billing period and saves it in database for record purposes
5. The system generates a digital copy of the bill and attaches it to the customer's account.

**Postconditions:**

- The customers will be notified via email or SMS about their monthly bills.
- The bill will be accessible by customers through the system.

**Assumptions:**

- The system and database are functioning correctly.




**Use Case 7: View Bill Details**

**Actors:**

- **Primary Actor:** Customer
- **Secondary Actor:** Support Admin

**Description:** This use case allow the customer to view detailed electricity billing information, such as outstanding balances, consumption records, and payment history, through a user-friendly interface.

**Preconditions:**

- The customer must have a registered account.
- The customer must be logged into the system.
- Relevant billing data must exist and be accessible in the system.

**Main Flow:**

1- The customer navigates to the "View Bills" section in the system interface.
2- The customer clicks on the specific bill for which they want to view details.
3- The system retrieves billing details based on the selected bill.
4- The customer reviews the displayed billing information:
   - Outstanding balance.
   - Total electricity consumption.
   - Payment due date.

**Postconditions:**

The customer successfully views their bill.

**Alternative Flows:**

**No bill data found**

The system sending the customer a message, "No billing information available." in case of no billing data available at its end.

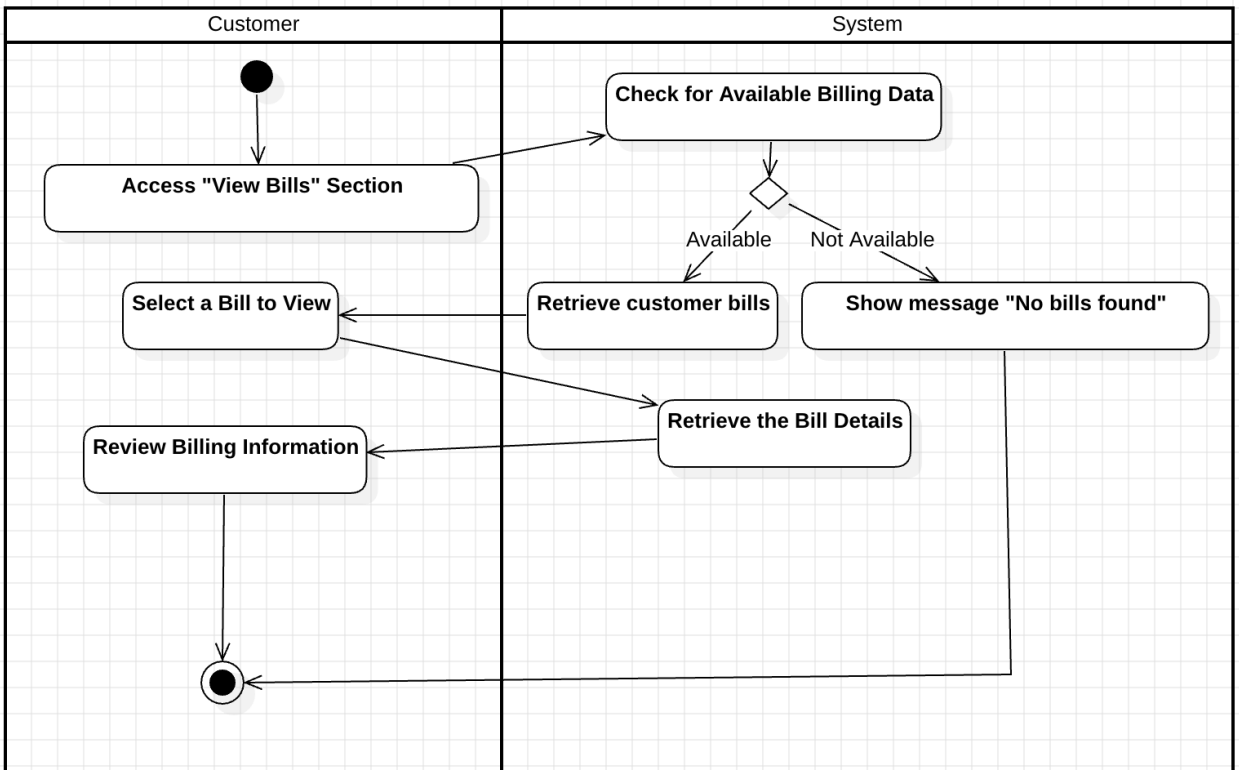
**Error Retrieving Bill from System**

If there is an error during bill retrieve the system displays a message (e.g., "Unable to retrieve bill details. Please try again later.").

The error is logged, and the customer arise issue to the support administrator for resolution.

**Assumptions:**

- The customer is able to access the system with a stable internet connection.
- Billing data is updated in real-time or at regular intervals for accuracy.
- The customer account is active, authorized, and not subject to any restrictions.




**Use Case 8: Make a Payment and Receive Payment Receipt**

**Actors:**

- Primary Actor: Customer

**Description:**

This use case allows the customer to make payments for their bills. Customers can choose which bill to pay, select a payment method, and complete the transaction. Upon successful payment, the system will generate a receipt, updates the database, and notifies the customer.

**Preconditions:**

- The customer must be logged into the system.
- The customer has at least one bill to be paid.
- The system and its database must be operational.

**Main Flow**:

1. The customer accesses the “Make Payment” page.
2. The system will generate a list of unpaid bills.
3. The customer selects a bill to be paid.
4. The customer enters payment details, selects payment method, and confirm transaction.
5. The system will validate the payment and generate a receipt.
6. The system will update the bill status in the database.
7. The system will display a confirmation message as well as the receipt.
8. The customer can view and download the receipt.

**Postconditions:**

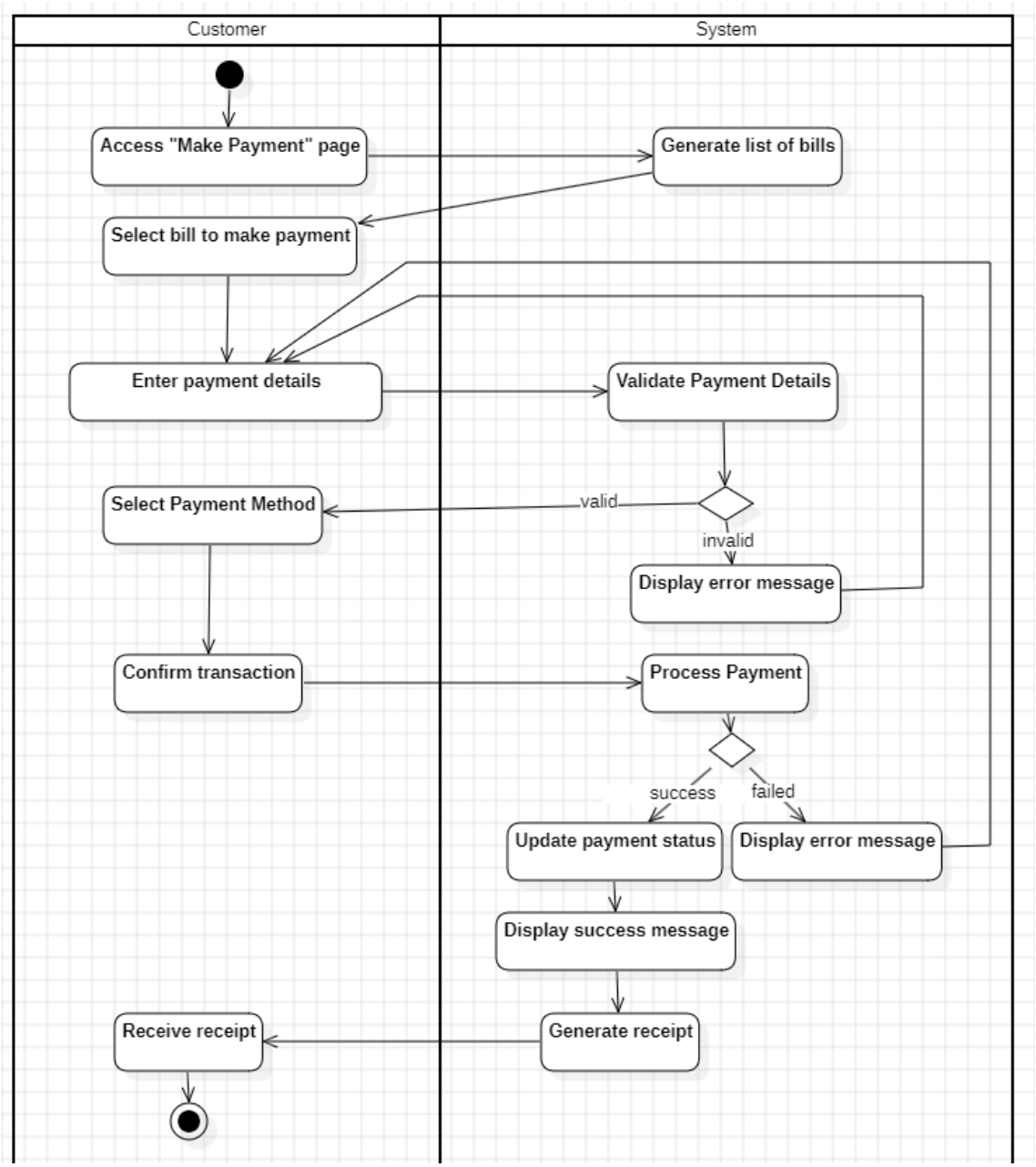
- The bill status is updated in the system.
- The customer receives the payment receipt.

**Alternative Flows:**

- **Invalid Payment Details**
  - If the customer’s payment details are incomplete or invalid, the system will display an error message, such as “Invalid Payment Details”.
  - The customer can re-enter their payment details.
- **Payment Failure**
  - If the customer’s payment fails, the system will display an error message, such as “Payment Failed”.
  - The customer can retry their payment.

**Assumptions:**

- The system is up and running, and capable of securing and validating payments.
- The system's database is functioning properly and is up to date.




**Use Case 9: Track Overdue Bills**

**Actors:**

- Primary Actor: Staff

**Description:**

This use case allows the staff to track overdue bills notified by the system based on the pay date. Staff will review the account details and payment histories to verify the overdue bill and will take necessary actions such as marking a payment as “Overdue” or “Paid”, and set a deadline for accounts with an overdue payment. The customer will be notified of the update by the system.

**Preconditions:**

- The system must have an up-to-date database of accounts and payment records.
- The staff must be logged in to access customers account details.

**Main Flow**:

1. The staff accesses the “Overdue Bills” page.

2. The system generates a list of accounts with overdue bills
3. The staff selects an account to view its details
4. The system displays list of overdue accounts with detailed information such as customer name, overdue amount, and due date.
5. The staff reviews the account and verify its issue.
6. The staff will mark the payment as “Overdue” and sets a deadline for the overdue bill.
7. The system is updated and will notify the customer.

**Postconditions:**

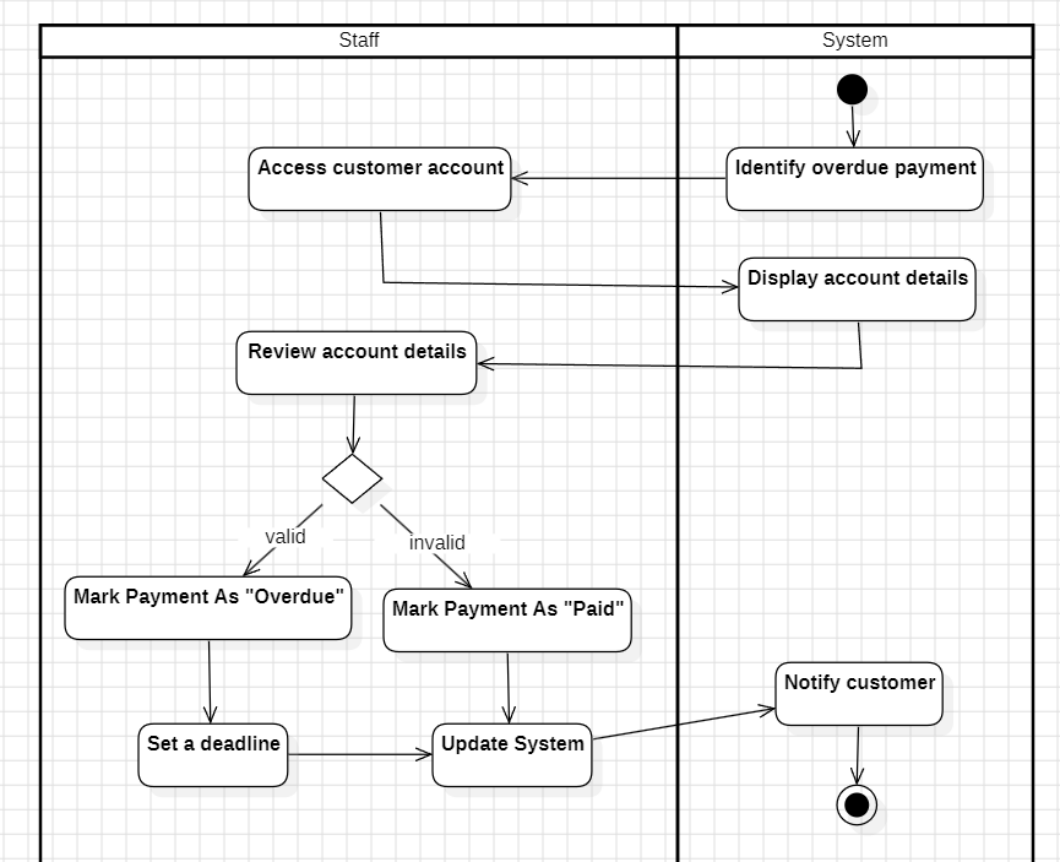
1. The system updates the overdue bill status and the necessary details.
2. Customer is notified about the overdue bill.

**Alternative Flows:**

- **Invalid Overdue Mark on Account**
  - If the account that has been reviewed by the staff is invalid for an overdue bill, the staff will mark the payment as “Paid”.
  - The system will be updated, and the customer will be notified.

**Assumptions:**

- The system is up and running, and capable of generating a list of overdue accounts.
- The system's database is functioning properly and can update new data.



**Use Case 10: Submit Feedback**

**Actors**:
• **Primary Actor**: Customer
**Description**:
This use case allows customers to submit feedback about their experience with the system. Customers can provide a rating and additional comments, which the system validates and stores for future review. After submission, the customer will receive a confirmation message acknowledging their feedback.

**Preconditions**:

- The customer must have a registered account.
- The customer must be logged into the system.
- The system is operational and able to receive feedback.

**Main Flow**:

1. The customer navigates to the "Feedback" section.
2. The customer fills out the feedback form, providing ratings and comments.
3. The customer clicks the "Submit" button to submit the form.
4. The system validates the form to ensure all mandatory fields (rating) are completed.
5. The system stores the feedback in the database for analysis purposes.
6. The system displays a confirmation message to the customer, such as "Thank you for your feedback!".

**Postconditions**:
• The feedback is successfully stored in the database for analysis.
• The customer sees a confirmation message acknowledging their feedback.

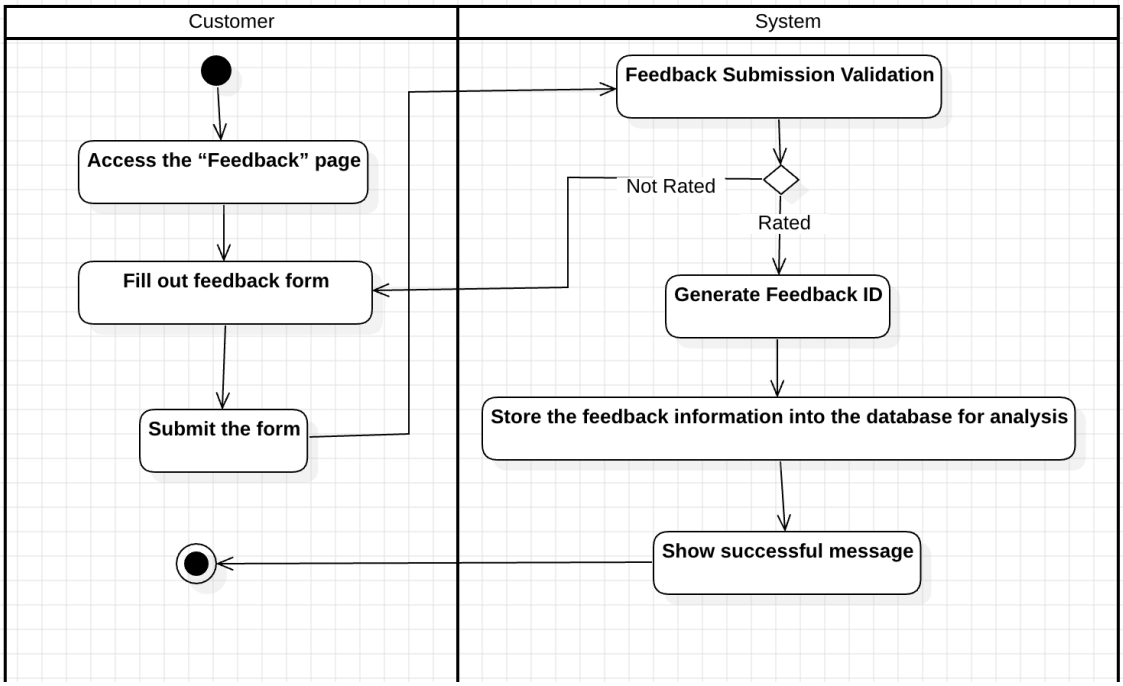
**Alternative Flows**:

**Incomplete Feedback Form:**

- If the user attempts to submit the form without selecting a rating, the system will reject the submission.
- The system will indicate that the rating field is required.
- The customer will be prompted to select a rating and fill in any other necessary fields.
- Once all mandatory fields (including the rating), are completed, the customer can resubmit the form.

**Submission Failure:**
If a system error prevents the feedback from being submitted, the system displays an error message "Please try again later".

**Assumptions**:
• The customer has a registered account and is able to log in.
• The customer has internet access to submit the feedback.
• The system is up and running, and capable of processing feedback submissions.
• The system's database is functioning properly and can store feedback data.




**Use Case 11: Submit and Resolve Customer Issues**

**Actors:**

- **Primary Actor:** Customer
- **Secondary Actor:** Support Admin

**Description:**

Allows customers to report problems related to electricity services and track their resolution progress. Staff members review, manage, and resolve these issues while providing timely updates to customers. The system assigns a unique ticket ID to each issue, ensuring efficient tracking and resolution of all reported concerns.

**Preconditions:**

- The customer must have a registered account.
- The customer must be logged into the system to access the "Support" section.

**Main Flow:**

1. The customer navigates to the "Support" section.
2. The customer completes the form by describing the details of issue.
3. The customer clicks the “Submit” button to submit the form.
4. The system generates a unique ticket ID for the issue.
5. The system sends a confirmation message to the customer, including the ticket ID and assign the issue status “In progress”.
6. The staff reviews the submitted issue.
7. The staff takes the appropriate action to resolve the issue.
8. The staff updates the status of the issue.
9. The customer receives email notifications regarding the issue status.
10. The customer is notified that the issue has been resolved.

**Postconditions:**

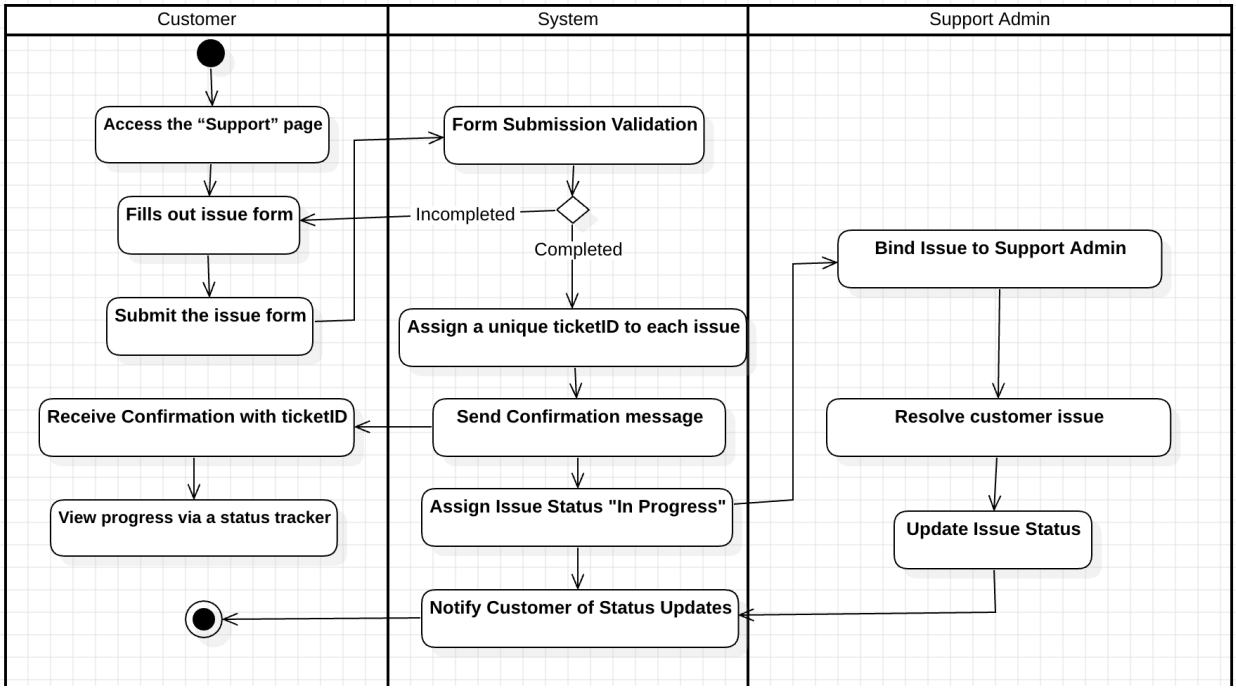
- A unique ticket ID is generated and assigned to each submitted issue.
- The system continuously tracks and updates the status of the issue.
- The customer is notified via email about any updates or resolutions related to their issue.
- The issue is marked as "resolved" once the staff has successfully addressed it.

**Alternative Flows**

- **Incomplete or Invalid Issue Form:**
  If the customer submits an incomplete issue form with some required fields not filled in, the system generates an error message requesting the customer to fill in the required fields. The customer corrects the form and resubmits it.
- **Submission Failure:**
  If a system error prevents the issue from being submitted, the system displays an error message "Unable to submit the issue. Please try again later".

**Assumptions:**

- The customer has internet access and is able to log into the system successfully.
- The customer is able to log in to their account using valid credentials.
- The system is online and functioning during the time the customer issue submission.
- Staff have appropriate permissions and sufficient training to handle and resolve the issues reported by their customers.
- Email services are working, and notifications can be sent successfully to the customer.




**Use Case 12: Usage Monitoring**

**Actors:**

- Primary Actor: Staff

**Description:**

This use case allows the staff to monitor key-metrics related to the system usage such as total number of customers, total number of bills generated, and total number of bills paid. The system will display the trends for staff reference.

**Preconditions:**

- The system must have an up-to-date database with relevant metrics logged.
- The staff must be logged in to access the "Usage Monitoring" page.

**Main Flow**:

1. The staff accesses the “Usage Monitoring” page.
2. The staff chooses which data to be displayed.
3. The system will check for the data availability.
4. The system displays the analytics in a simplified interface.
5. The staff will be able to view and monitor the required data.

**Postconditions:**

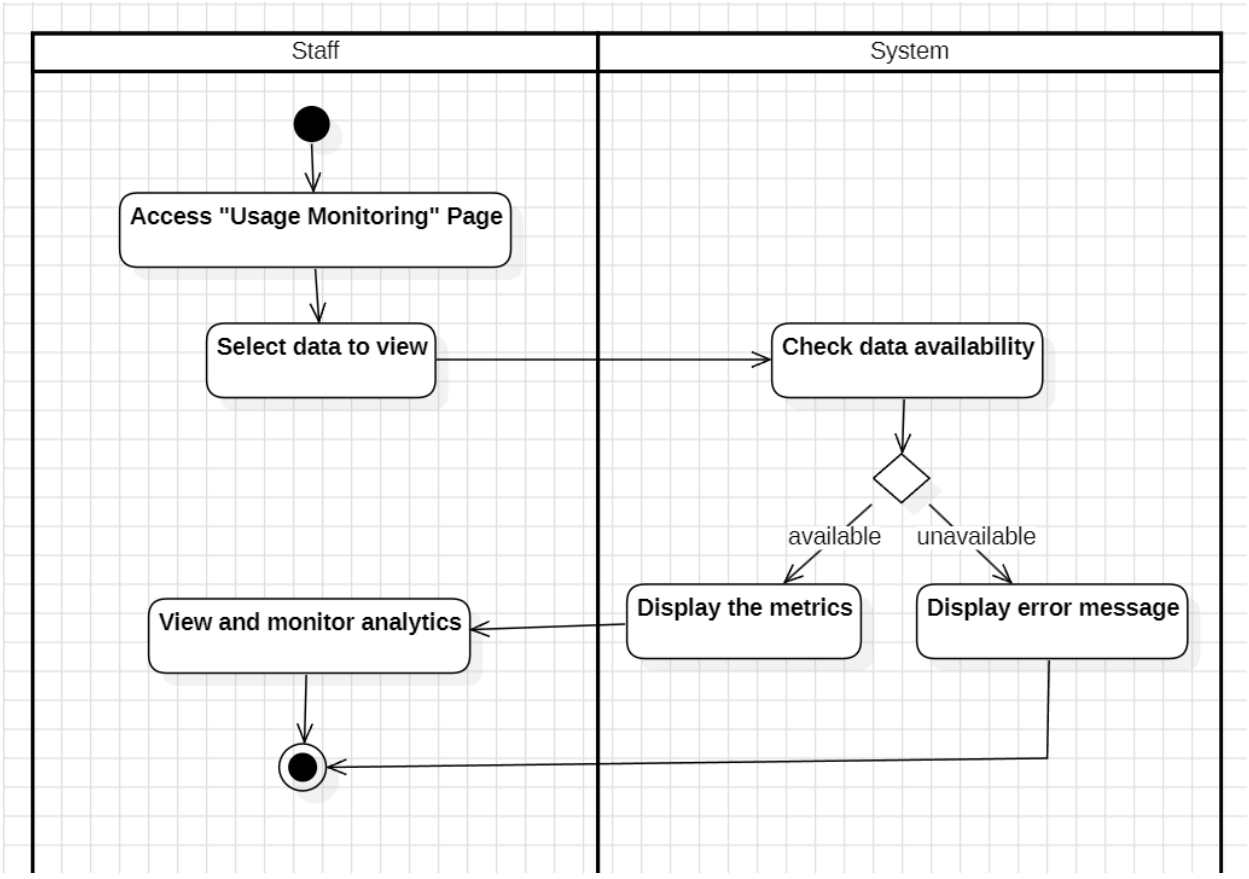
- Staff receives a clear summary of required analytics.
- Data is display accurately and updated.

**Alternative Flows:**

- **No data available**
  - If the data requested by the staff is not available, the system will display an error message, such as “No data is available”.

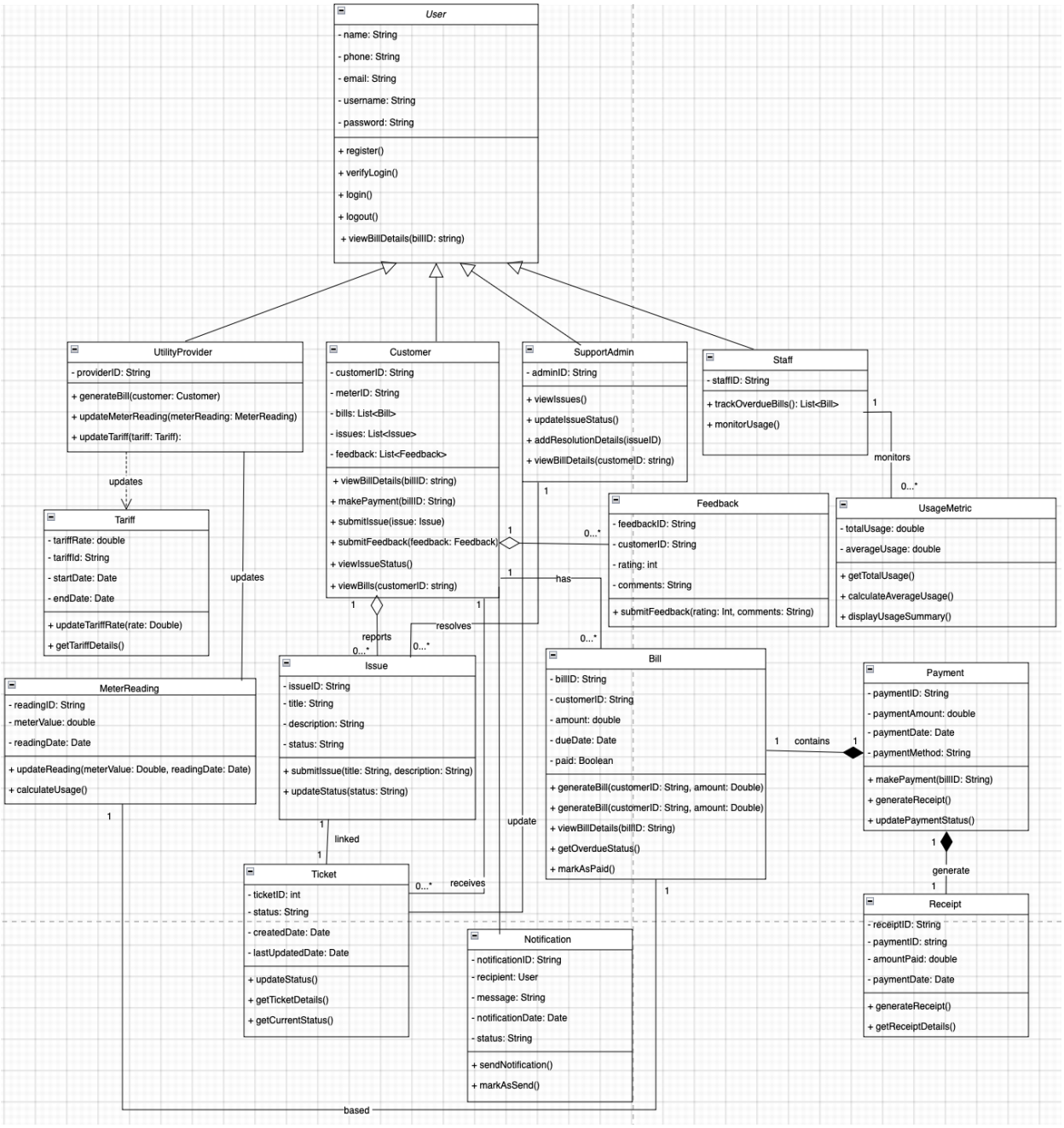
**Assumptions:**

- The system is up and running, and capable of generating key-metrics for the system’s usage.
- The system's database is functioning properly and is up to date.



# Requirements Modeling

## Class Diagrams



## Classes / Entities

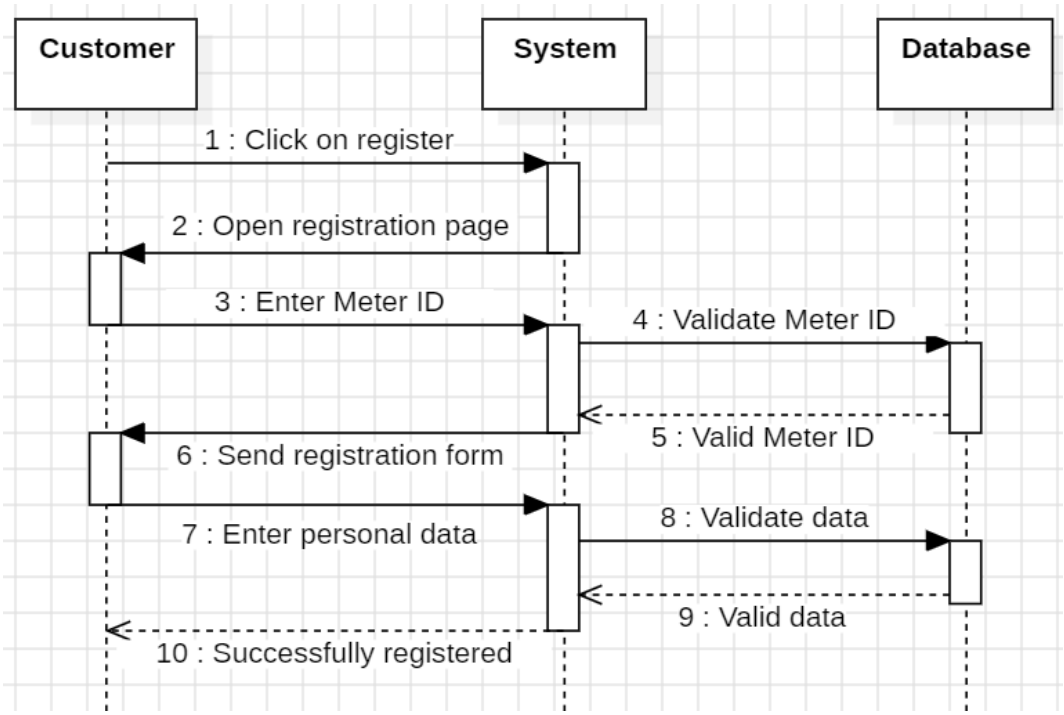
| Class / Entity | Description |
|---|---|
| User | The User class defines the general attributes and functionalities for all user types in the system. It contains key properties like name, phone, email, username, and password, which are important features for identifying and authenticating users. The class also defines core methods such as verifyLogin(), login(), and logout() that handle user session management. This class will be the base class for all other specific user types (classes), such as Staff, Supporting Admin, Utility Provider classes and Customer, which inherit from it and thus can implement similar functionality while allowing customization for each role. |
| Customer | The Customer class inherits from the User class, sharing common attributes and methods while adding specific features to customers. It includes attributes such as meterID, issues-a list of reported issues, and submitted feedback.<br>It has methods like viewing bills through viewBills(), tracking the issues in viewIssues(), making payments through makePayment(), reporting problems via submitIssue(), and submitting feedback via submitFeedback(). This is a specialized class to gives customers a good experience. |
| Support Admin | The SupportAdmin class is specialized for support personnel who oversee customer issues and their resolutions. Inheriting from User, it introduces unique functionality tailored to issue management. Methods such as viewIssues(), updateIssueStatus(), and addResolutionDetails() help the support admin track and resolve customer-reported problems. The support admin also has the responsibility to view customer bills and manage system-wide issues, making this class essential for ensuring that customer concerns are addressed and resolved promptly. |
| UtilityProvider | The UtilityProvider class represents the service provider in the system, responsible for managing the utility services provided to customers. It includes providerID, which identifies the provider, and methods for managing essential data like meter readings and tariffs. The class facilitates generateBill() and updateMeterReading() to calculate charges based on usage and to set tariff rates for services provided. By managing these core operations, the UtilityProvider class ensures that customers receive accurate bills and that the service's pricing structure is updated as needed. |
| Staff | The Staff class inherits from User and defines the roles and responsibilities of employee managing the system. With attributes like staffID, this class helps define who can access various system functionalities. Staff member can monitor customer UsageMetrics and track overdue bills. The ability to monitor usage and track overdue bills ensures that staff can efficiently oversee customer accounts, respond to the payment issues, generate trends reports and maintain operational efficiency within the system. |
| Bill | The Bill class represents a utility bill issued to a customer. It includes attributes such as billID, customerID, amount, dueDate, and paidStatus. Methods like generateBill(), viewBillDetails(),<br>and getOverdueStatus() are used to generate bills, view bill details, and check if the bill is overdue, allowing customers and staff to track payments and statuses and markAsPaid() updates the payment status. |
| Payment | The Payment class records payments made by customers. It holds attributes such as paymentID, paymentAmount, paymentDate, and paymentMethod. Customers can use methods like makePayment() and |

| | |
|---|---|
| | updatePaymentStatus() to process payments and track the customer payment status by the staff. |
| Receipt | The Receipt class provides customers with proof of payment. It holds information like receiptID, paymentID, and amountPaid, and offers methods like generateReceipt() and getReceiptDetails() to generate and retrieve detailed receipt information. The Receipt class is critical for confirming the completion of payments, ensuring that customers have an official record of their transactions, which they can use for future reference or dispute resolution. |
| MeterReading | The MeterReading class holds data about the readings taken from customer meters, with attributes such as readingID, meterValue, and readingDate. It provides the updateReading() method to update meter readings and calculateUsage() to calculate the usage based on the recorded values. |
| Feedback | The Feedback class enables customers to share their experiences with the service by submitting ratings and comments. It includes attributes like feedbackID, customerID, rating, and comments, and the submitFeedback() method allows customers to provide feedback on their service experience. This class helping the support admin identify areas for improvement and allowing them to assess customer satisfaction. |
| Tariff | The Tariff class helps calculate electricity pricing with attributes such as tariffID, tariffName, and rating. It categorizes customers under different plans based on their usage. The applyTariff() method calculates charges for consumption, while updateTariff() allows for modifying rates and conditions. This class ensures transparent and adaptable billing for all customers. |
| Ticket | The Ticket class represents a support request generated when a customer reports an issue. Attributes include ticketID, status, and createdDate. Methods like updateTicketStatus() and getTicketDetails() allow users and support admin to track the ticket's progress and resolve issues. |
| UsageMetric | The UsageMetric class allows for managing and analyzing key metrics of the system efficiently with two key attributes which are totalUsage and averageUsage. It provides functionalities to retrieve, calculate and display the usage with methods such as getTotalUsage(), calculateAverageUsage() and displayUsageSummary(). |
| Notification | The Notification class control the messages send to customer. Attributes include notificationID to uniquely identify each message, recipient, which specifies the intended customer, message containing the notification content, and status, which tracks whether the notification has been sent or not. Methods such as sendNotification() and markAsSend() to ensure that customers remain up-to-date with all critical information. |
| Issue | The issue class tracks problems reported by customers regarding utility services. It contains attributes like issueID, description, status, and resolutionDate. Methods such as submitIssue() and updateStatus() allow customer to report issues and track their resolution. |

# Behavioral & Flow Modeling (optional)

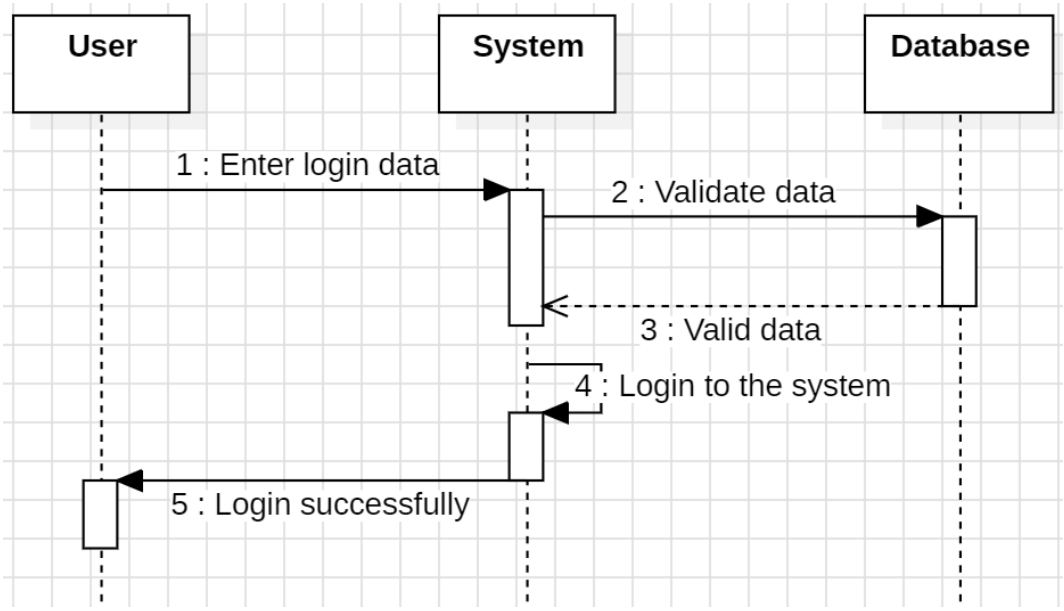
## Sequence Diagrams

### Use Case 1: System Registration



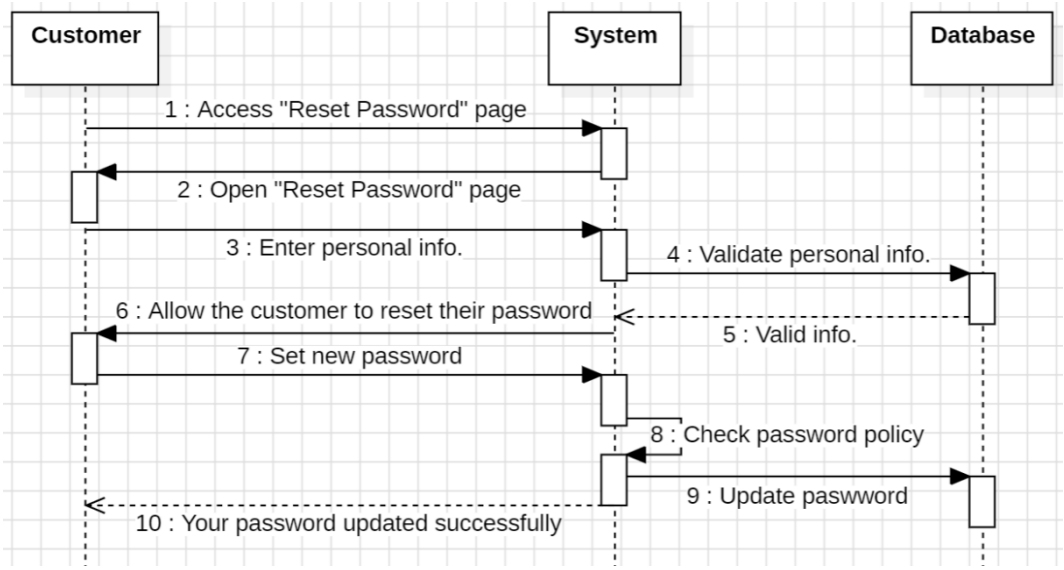

This sequence diagram illustrates the customer registration process for the system. The customer begins by clicking the "Register" button to open the registration page. They then enter their Meter ID, which the system validates by interacting with the database. Upon receiving confirmation of a valid Meter ID, the customer fills in and submits their personal information. The system validates this data with the database, and upon successful validation, the registration process is completed, notifying the customer of successful registration.

**Use Case 2: Log in to the System**




This sequence diagram describes the user logging into the system process. The user enters their username & password, which will be validated by the system by interacting with the database. Upon receiving confirmation of valid data, the system logs into the system to allow the user to use their perspective specialized dashboard.

**Use Case 3: Reset Password**



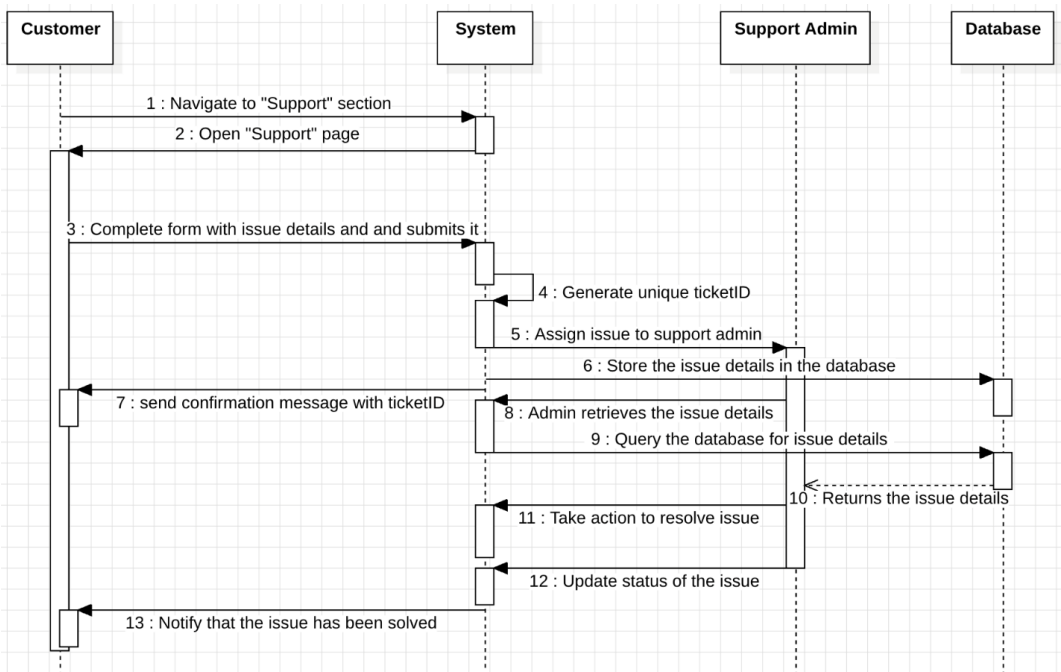

This sequence diagram represents the password reset process. The customer accesses the "Reset Password" page so the system opens it allowing the customer to fill in their personal information. The system verifies that information with the database. After verification, the system allows the customer to create a new password. The system verifies the new password with the password policy and updates it in the database if it's valid. Finally, the customer is notified of the successful update of their password.

### Use Case 4: Update Personal Details

### Use Case 5: Update Meter Readings and Tariffs

### Use Case 6: Generate Monthly Bills

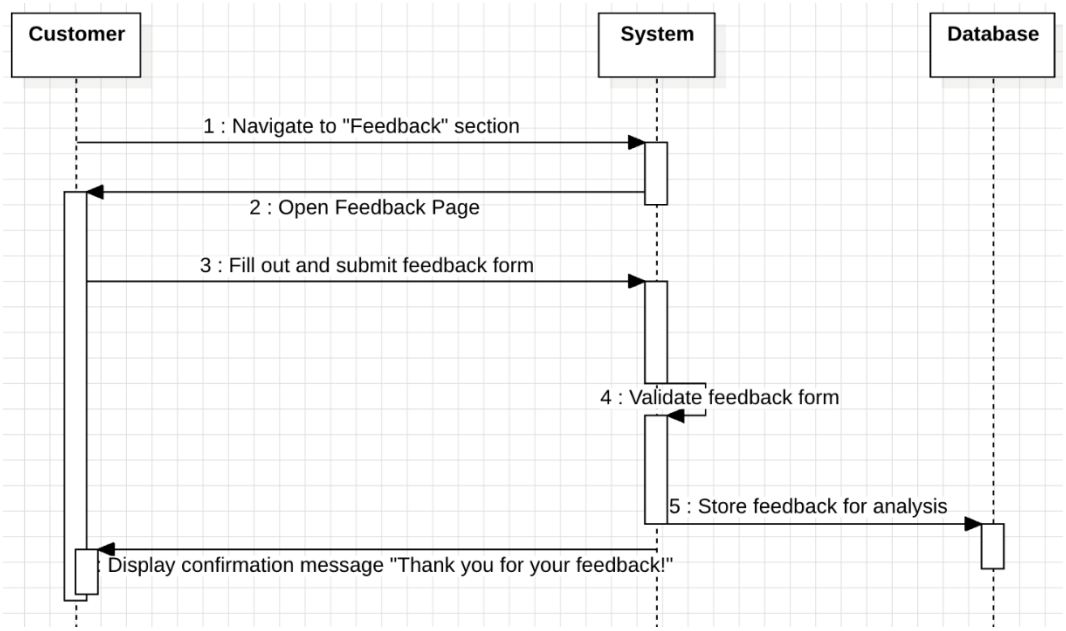
### Use Case 7: Resolve Customer Issues



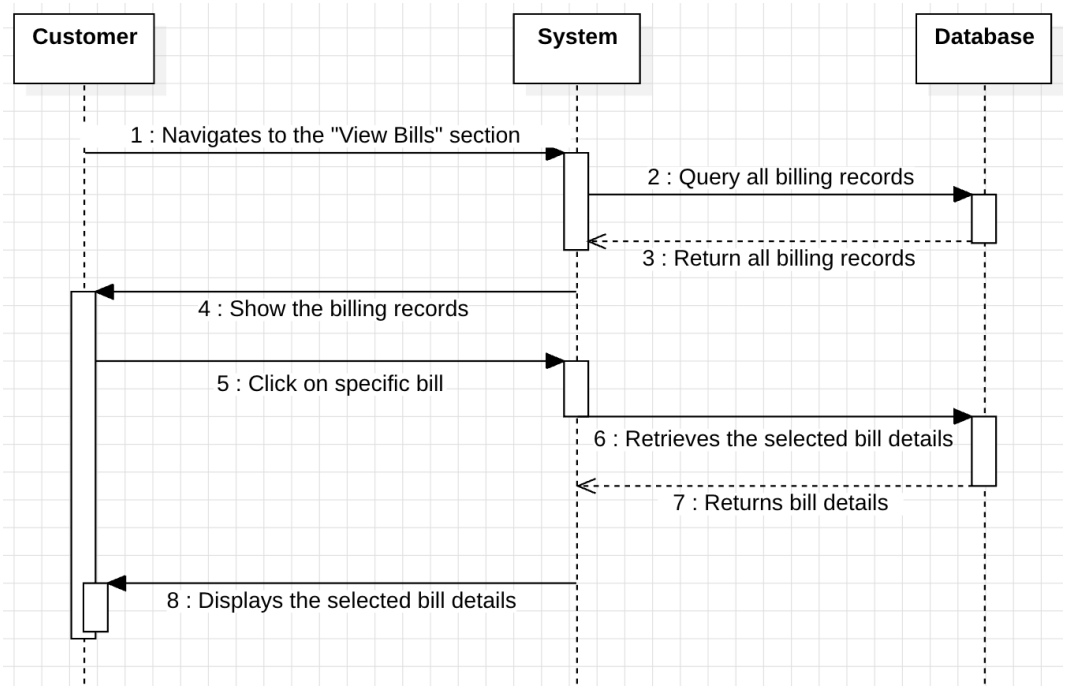

This sequence diagram shows the interaction between the Customer, System, Database, and Support Admin in logging an issue of a customer, processing it, and resolving it in a structured approach. It provides how each actor contributes to issue management, highlighting system notifications and updates throughout the process.

### Use Case 8: Submit feedback

This sequence diagram outlines the process of submitting feedback. The customer navigates to the "Feedback" section to open the feedback page. After filling out and submitting the feedback form, the system validates the form if it either contains anything or is empty. If valid, the feedback is stored in the database for further analysis. Finally, the system displays a confirmation message thanking the customer for their feedback.

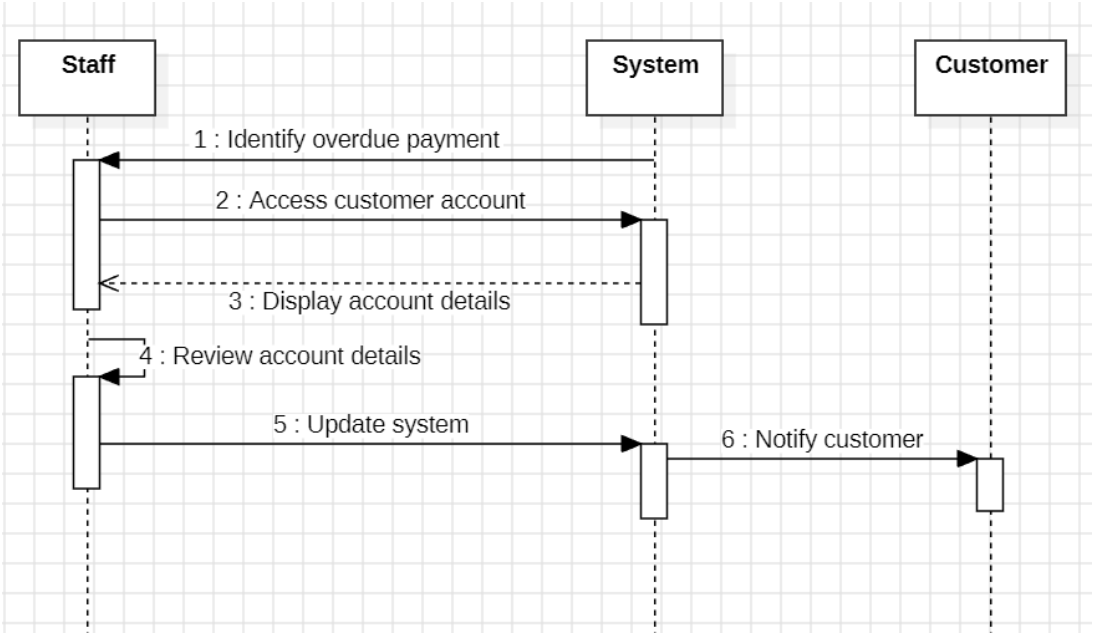



## Use Case 9: Show bill details



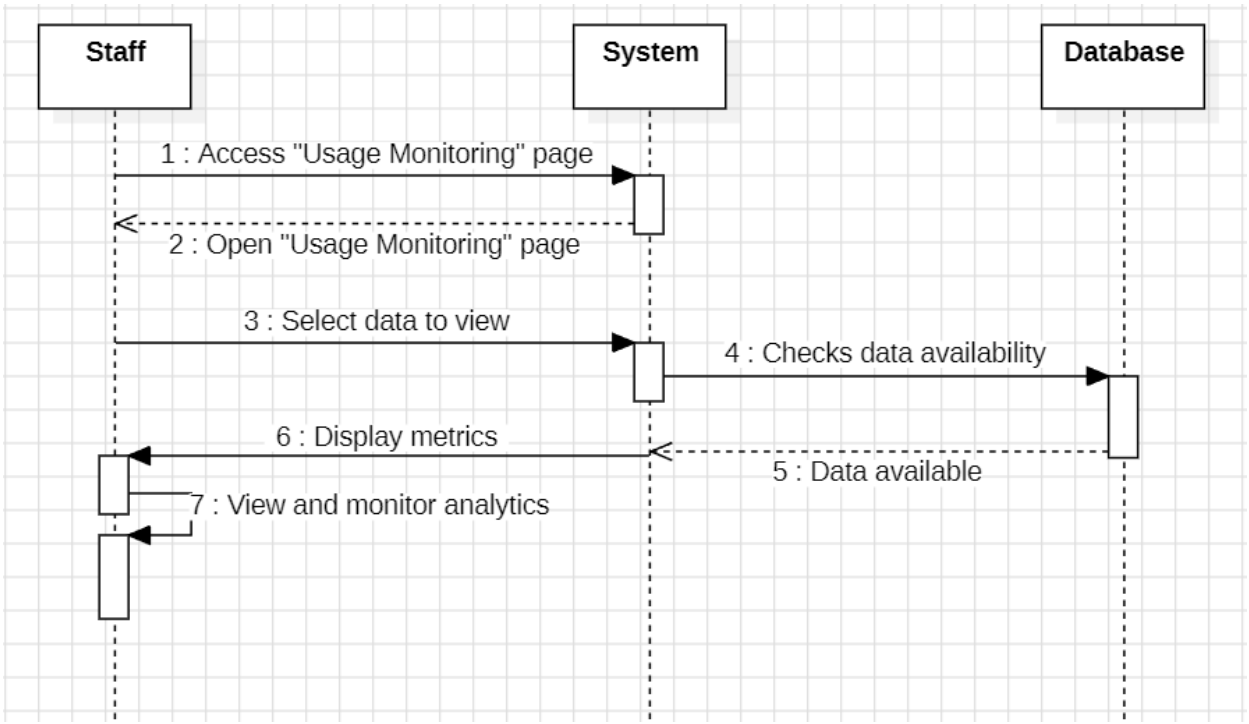

This sequence diagram shows the process for the customer to see their bill details. The customer navigates to the "View Bills" section which causes the system to retrieve all records from the database. After the system gets all bills from the database it will be shown to the customer. Then the customer will be able to choose any active bill to view. Finally, the system displays the selected bill details.

**Use Case 10: Track Overdue Bills**



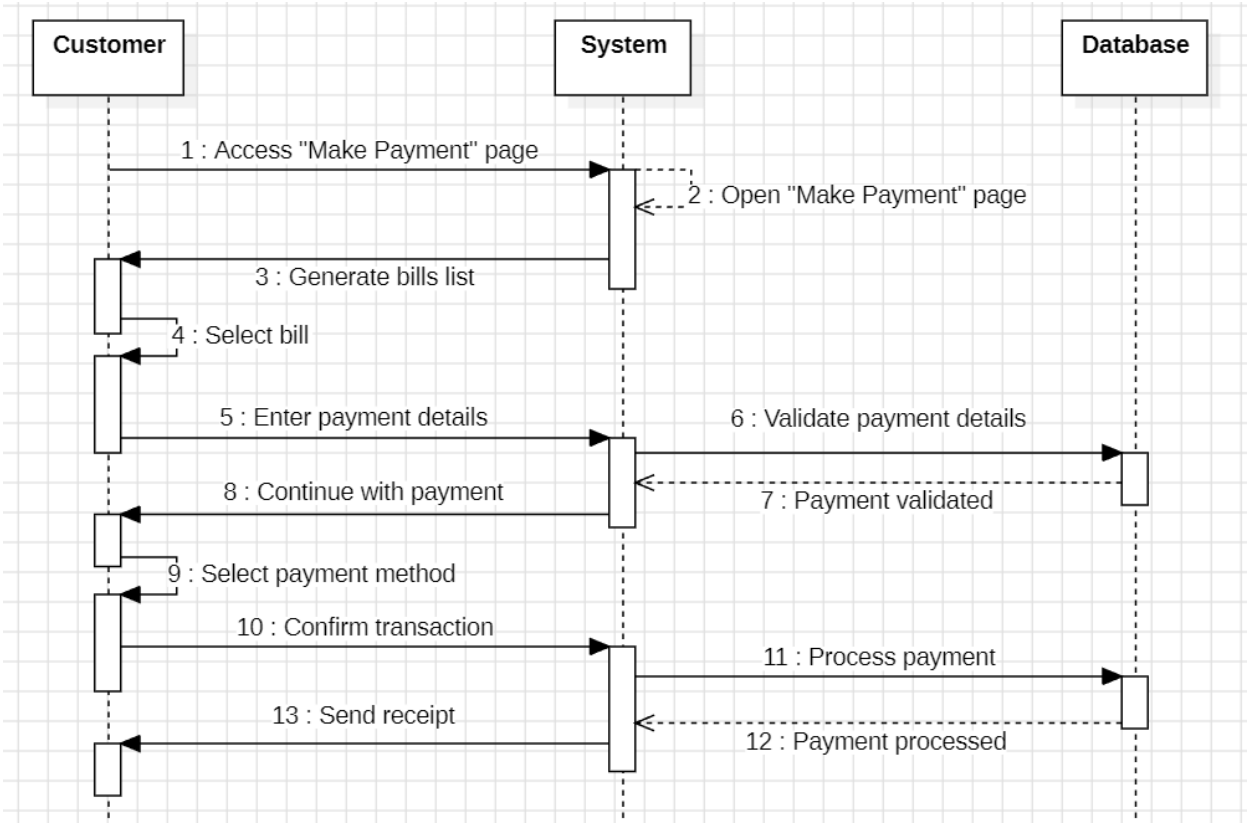

This sequence diagram shows how staff tracks overdue payments. The system will identify overdue payments based on the billing due date. The staff can then access customer accounts to review and verify overdue payments and will update the system accordingly. The system will then notify customers about updates based on their billing status.

**Use Case 11: Usage Monitoring**



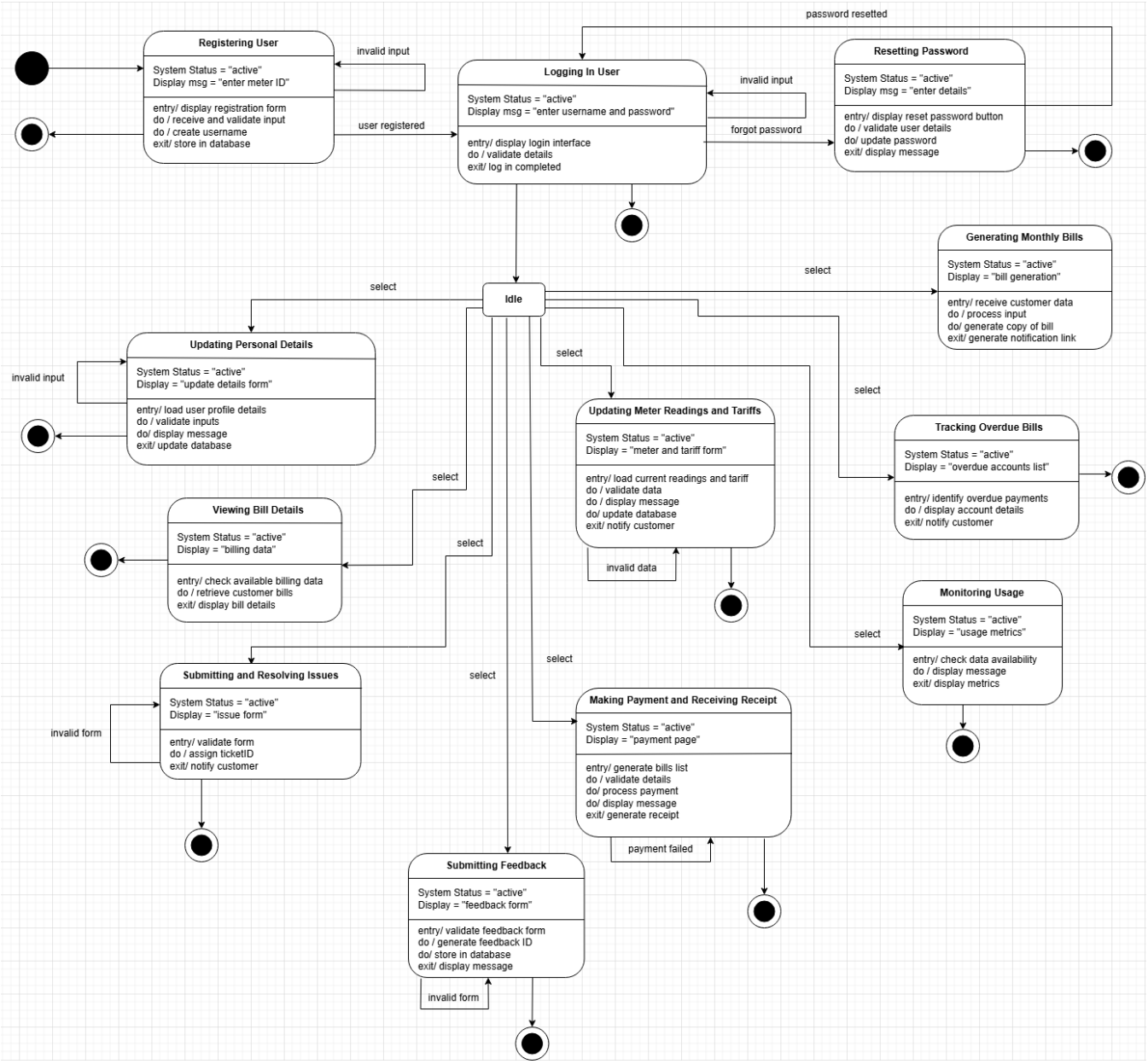

This sequence diagram shows the process of usage monitoring by the staff. The staff will first access the “Usage Monitoring” page and select specific data to view. The system will check the data availability and display them to staff for them to view. The staff can then monitor the analytics and trends of the data.

### Use Case 12: Make a Payment and Receive Payment Receipt




This sequence diagram illustrates how customers can make a payment by firstly accessing the “Make Payment” page, where the system will then show a list of bills. The customer can select a bill to make a payment and enter their payment details. The system will validate the payment details before allowing the customer to continue with payment by selecting a payment method and confirming the transaction. The system will then process the payment and send the customer a copy of the receipt.

# State Diagram




1. **Registering User:**
   - **Description:** This state handles new user registrations by loading the registration form, validates user inputs, and confirms successful registrations. If validation fails, the state transitions back to itself for retrying.

2. **Logging In:**
   - **Description:** In this state, users will input credentials to access system. Successful login will allow users to access the system, where failed attempts will transition for a re-entry.

3. **Resetting Password:**
   - **Description:** Allows users to reset their password if the user forgets and transitions back to the login page.

4. **Updating Personal Details:**
   - **Description:** Users can update their profile information where the system validates the new inputs and confirms successful updates.

**5. Updating Meter Readings and Tariffs:**
- **Description:** This state facilitates updating meter readings and tariff details. The system will ensure inputs are valid before saving them.

**6. Generating Monthly Bills:**
- **Description:** Calculates and generates bills based on user data. The system will receive data from customers, processes input and generate a copy of the bill.

**7. Viewing Bill Details:**
- **Description:** Displays bill details to the user, including usage, and payment histories.

**8. Making a Payment and Receiving Receipt**
- **Description:** Allows user to make payments for their bills. The system will process the transaction and generates a receipt to be sent to the customer.

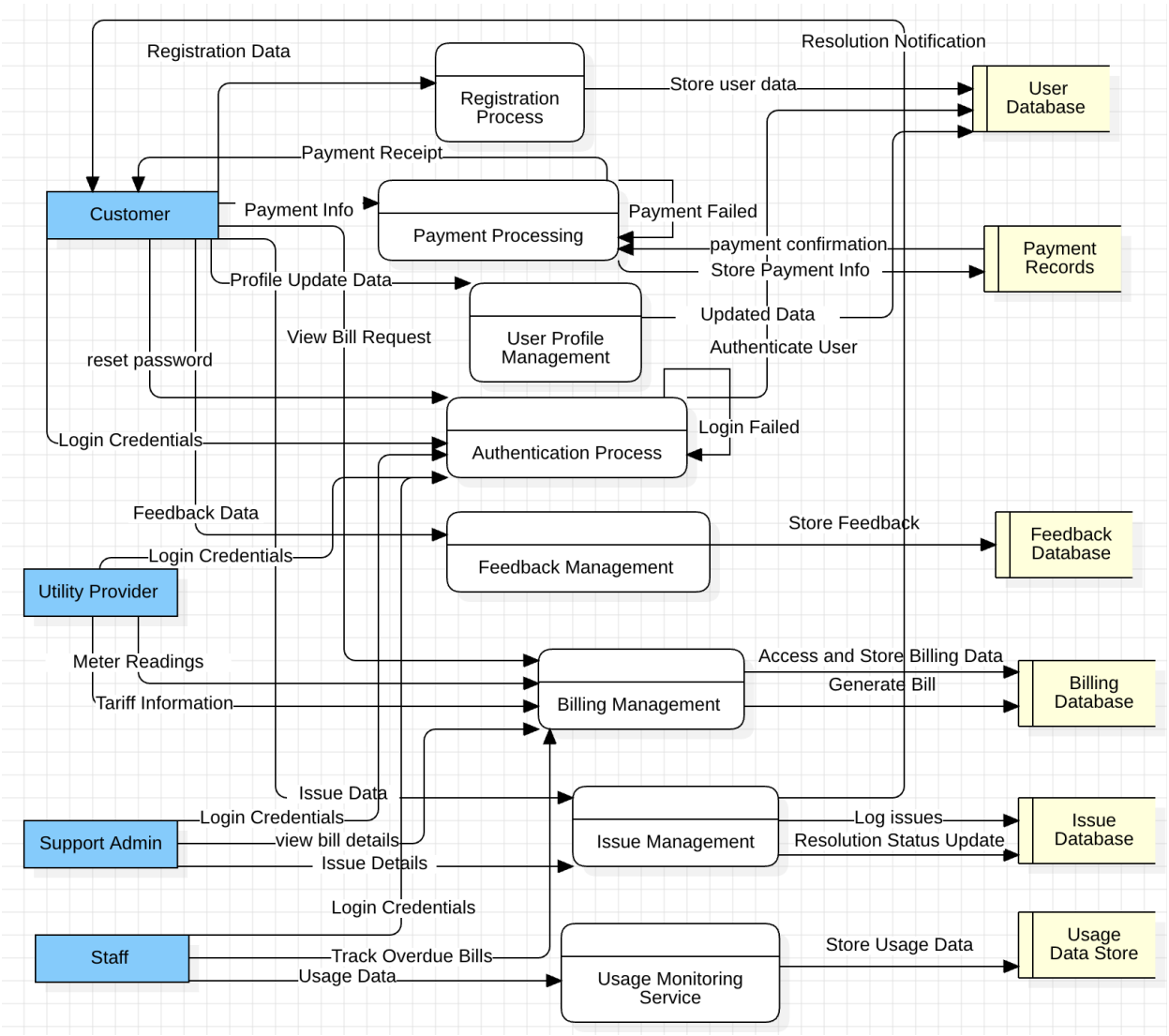
**9. Tracking Overdue Bills:**
- **Description:** Manages overdue accounts by allowing staff to verify and notifies customer on further action.

**10. Submitting and Resolving Issues:**
- **Description:** Users can submit tickets for system-related issues where the system will process and resolve these tickets.

**11. Submitting Feedback:**
- **Description:** Collects feedback from users about the system's functionality and performance.

**12. Monitoring Usage:**
- **Description:** Displays usage statistics and key-metrics, allowing the staff to track and monitor system usage.

# Data Flow Diagram




### Level 0: Context Diagram

The Level 0 Data Flow Diagram provides a high-level overview of the electricity billing system, highlighting interactions with key external entities:

- Customer: Registers, logs in, views bill, makes payments, updates details, resets passwords, submits issues, and provides feedback.
- Support Admin: Logs in, views billing details, and resolves customer issues.
- Staff: Logs in, tracks overdue bills, and monitors electricity usage.
- Utility Provider: Updates meter readings, tariff details, and generates monthly bills.

The system acts as a central core, facilitating secure data flow for user management, payment processing, billing, feedback handling, issue resolution, and usage monitoring, ensuring efficient operations and user satisfaction.

### Level 1 & 2: Process Descriptions and Data Storage

The following processes are the core functionalities of the electricity billing system. Each process is connected to relevant actors and data stores, ensuring a seamless flow of data and reliable system operation:

- **Registration Process:**

Facilitates customer registration by capturing and securely storing user details in the User Database. It validates the registration data provided by customers to ensure accuracy and completeness.

- **Authentication Process:**

Manages the login credentials for all users (Customers, Support Admins, Staff, and Utility Providers), verifying access permissions. It also handles login failures, ensuring that only authorized users can access the system.

- User Profile Management:

Enables customers to update their personal details or reset password. This process ensures that the User Database is kept up-to-date and that customers can easily manage their accounts.

- **Payment Processing:**

Processes customer payments, managing payment confirmations and failures. It generates receipts for successful transactions and updates the Payment Records database accordingly, ensuring accurate financial records.

- Feedback Management:

Allows customers to submit feedback regarding the service. The feedback is recorded and stored in the Feedback Database for further analysis and service improvement.

- **Billing Management:**

Uses meter readings and tariff updates to generate accurate monthly bills for customers. These bills are stored in the Billing Database, ensuring that the billing information is properly maintained and accessible.

- **Issue Management:**

Logs customer-reported issues, tracks their progress, and updates the Issue Database with the resolution statuses. This process ensures that all customer issues are handled efficiently.

- **Usage Monitoring Service:**

Monitors electricity usage and tracks overdue bills. This data is stored in the Usage Data Store and is used by staff for analysis, helping to ensure that overdue bills are addressed in a timely manner.

# Other Requirements

*<This section is **Optional.** Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*