Group No 1

Riya Vishnoi, Prabhkirat, Shioong, Mitchell

Bill Management SYstem

A system developed to help the user in managing their bills with ease

Contents

[Executive Summary 2](#_Toc131020714)

[Design Analysis Process 3](#_Toc131020715)

[Entity Relationship Diagram 5](#_Toc131020716)

[Table Designs – Data Dictionary 7](#_Toc131020717)

[Contributions 11](#_Toc131020718)

[References 12](#_Toc131020719)

# Executive Summary

BMS is a system that enables consumers to manage all of their bills in one location. Our team worked hard to develop this system, believing it should simplify the monotonous and tedious task of paying multiple bills with a click of a button. Users can view the status of their various bills, including whether or not they have been paid, owing to the system's partnership with bill providers.

The BMS System has collaborated with bill providers to collect information on the user's bills, such as payment due dates and amounts. Users will then see a list of all their bills, which they can filter by provider, due date, or payment status. This makes it simple to identify bills that are approaching or have already passed due dates and take action to pay them before incurring late fees or penalties.

Another upside of the Bill Management Database System is that it gives users an in-depth view of their bill history. Users can view a history of their expenses, including which bills they paid and when. This data can be used to identify spending trends, track changes in bill amounts, and make informed spending decisions in the future.

# Design Analysis Process

**ANALYSIS PROBLEM DOMAIN**

This billing system is automated and efficiently handles clients' information, payment history, and billing. Its purpose is to provide clients with modern technology for quicker and more accurate billing.

Back in the day, billing was done manually. It causes clients and the company to be inconvenienced while the bill is being processed. As a result, there is room for improvement here, and a bill management system could help. Billing Management Systems enable the entry of clients, bill providers, payment information, and bank sources.

The following issues are present in the unprocessed system:

* Inability to modify data.
* Manual operator control
* Lots of paperwork
* Difficult to record information systematically.
* Difficult to retrieve information in time.

Before we begin a new system, we must first study the system that will be improved or replaced (if there is one). We must examine how this system employs hardware, software, network, and human resources to convert data resources, such as transaction data, into information products, such as reports and displays. As a result, we must document how information system activities such as input, processing, output, storage, and control are carried out.

Inability to modify data: Managing large amounts of data effectively and efficiently for efficient results, storing consumer information, etc. in such a way that the database can be modified, which is not possible in the current system. As many past transactions as the user can have, the old transactions may need to be deleted to make more space or to make the system faster.

**And how can bill management solve it?**

Bill generation in a matter of seconds - Because hand billing is no longer required, bill generation becomes simpler and less time-consuming. Furthermore, with a billing system, you can print a bill in the customer's dashboard in seconds, reducing the time customers have to wait.

1. Improve customer satisfaction - A billing system allows you to improve customer satisfaction. Customer delights include not only shorter lines but also accurate and on-time payments.

1. Create reports - When you start using a billing system, you no longer have to keep track of any aspect of your business manually. The system handles everything for you. Review the system's reports.

A billing solution automates the client's data so that they can gain control over all aspects of their payments.

ACTORS IN OUR DATABASE:

* USERS: The users play a significant role in running the system. They can use the system to track all their bills and keep an eye on their bill history.
* BILL PROVIDERS: These companies provide specific bills to every user. They also keep track of the payment status of every user’s bills.
* BILLS: These are the actual physical bills present in the system. They hold all the relative information regarding the bill- for example, the amount, its type, due date, etc.
* BANKS: These are the banks that the user has links with. They help in keeping an acknowledgement of the user’s paid bills.

EVENTS:

*Following are some of the events that will be performed to proceed in the Bill management system. They are categorised broadly into three types external, internal and temporal.*

* EXTERNAL EVENTS: *This generally involves the data which will be inserted into the system by an outer source (user), for ex -: Creating an account  and Adding information like name, address, contact number, and Bank details(debit card or credit one)*
* INTERNAL EVENTS: These are the functions the system will perform on its own. These are storing bills, auto payment (if set automatic) AND keeping payment history.
* TEMPORAL EVENTS:    Under this system, reply to a timer-- Due dates reminder, payment status (paid or due)

WHY DID WE BUILD THIS SYSTEM?

There was much brainstorming that went into building this system. We were looking to build something useful for international students in general. We had ideas like – hotel booking, journey planning, and to-do list maintenance, but suddenly we realised that one of the biggest international struggles. Students are in money management.

We went forward with that idea and came to the design of a bill management system. It would help users manage their bills efficiently and protect them from overspending their budget in the long run.

We approached our project by designing multiple incorrect ERD diagrams until we found one that made sense. We then went ahead and made a data dictionary and finished our proposal report. In our final database, we have added and updated some fields in some entities as we believed it would make it more user-friendly and comprehensive.

Overall, we hope to expand this system to be more helpful towards the bill providers and banks as well. However, our primary focus will always be towards are loyal users.

* From Team BMS

**The database should meet these non-functional requirements:**

* Flexibility: quickly deliver on requirements changes as users demand new reporting data, different metrics, additional reference data, and changes in the level of report detail
* Simplicity: make report SQL as simple as possible
* Reliability: data for reporting is available promptly and is consistent with operational sources
* Security: users have access only to data they need

**The database should meet these functional requirements:**

* Add users, bill providers, bank branches to the database
* Let user check all their bills from the billing entity
* Be able to do basic sum, average, count functions to give the user an analysis of their bill spending
* Be able to update any entity- user or provider or any field- bill amount or user name
* Ba able to delete entities from the database

**SOFTWARE Methodology**

Diagram

Description automatically generated

We thoroughly researched Waterfall and agile software methodologies and decided to go for Agile. These are the reasons for choosing this specific methodology:

1. Agile is predominantly used with software that requires more freedom and relies on customer feedback. In contrast, the waterfall is a more traditional approach that follows a linear path and is more organised and pre-planned.
2. This method breaks down the software-building process into several pieces and focuses on quickly responding to customer feedback. It also leads to collaborative teamwork where we can delegate various authorities to many departments, making software delivery quicker.
3. We wanted a system that would be open to feedback and flexible to the user requirements. Since the requirements and needs of user change quickly, using agile will give us the power to revisit any cycle and fix the problems, unlike in waterfall, where you only ask for user feedback at the end of the development cycle.

USER STORIES:

* An International student who just left their home country says: “I am thrilled to have a system where I can track all my paid and missed bills. As an international student, Bill management is relatively new for me, and I seem to be more organised with a system like this in place.”
* A mother of 5 kids says: “Due to the abundance of bills coming in their home, I find it easier to open this database and check everything there. I also love to check my bill history as it helps me keep track of things.”
* A bill provider company said: “It has been easier to check the number of people connecting to us through this database. It makes it easier to keep track of your clients.”
* One of the interviewed users said: “We have been using this database for quite some time now, and we believe that we have been able to analyse our expenses over the past few months. It has helped us reduce our expenses and track our average monthly expense per bill provider.”
* A bank official said: “I have found it easy to track all the bills linked to our branch. With the help of BMS, I have tracked the bill number, the user linked to the bill and the amount. It has made my life of acknowledging the bills very easy.”

USER CASES

**For the User:**

* Add Personal Information (Name, Bank account number)
* Update their personal information at any time
* Check their current paid bills
* Check if they have any due or missed bills
* Be able to check if there was a fine on any of their missed accounts (if yes, then the fine amount too)
* View their Bill history (all the paid accounts)
* View a Bill analysis:: Average money paid for a specific type of Bill Provider (water: $200.00 for one month)
* View and Update their current Providers along with their bank source
* Update their bank balance
* Be able to update any incorrect payment status (A paid bill showing as due)
* View a comprehensive report of all the types of bills and their basic information.
* Be able to delete the paid bill history.

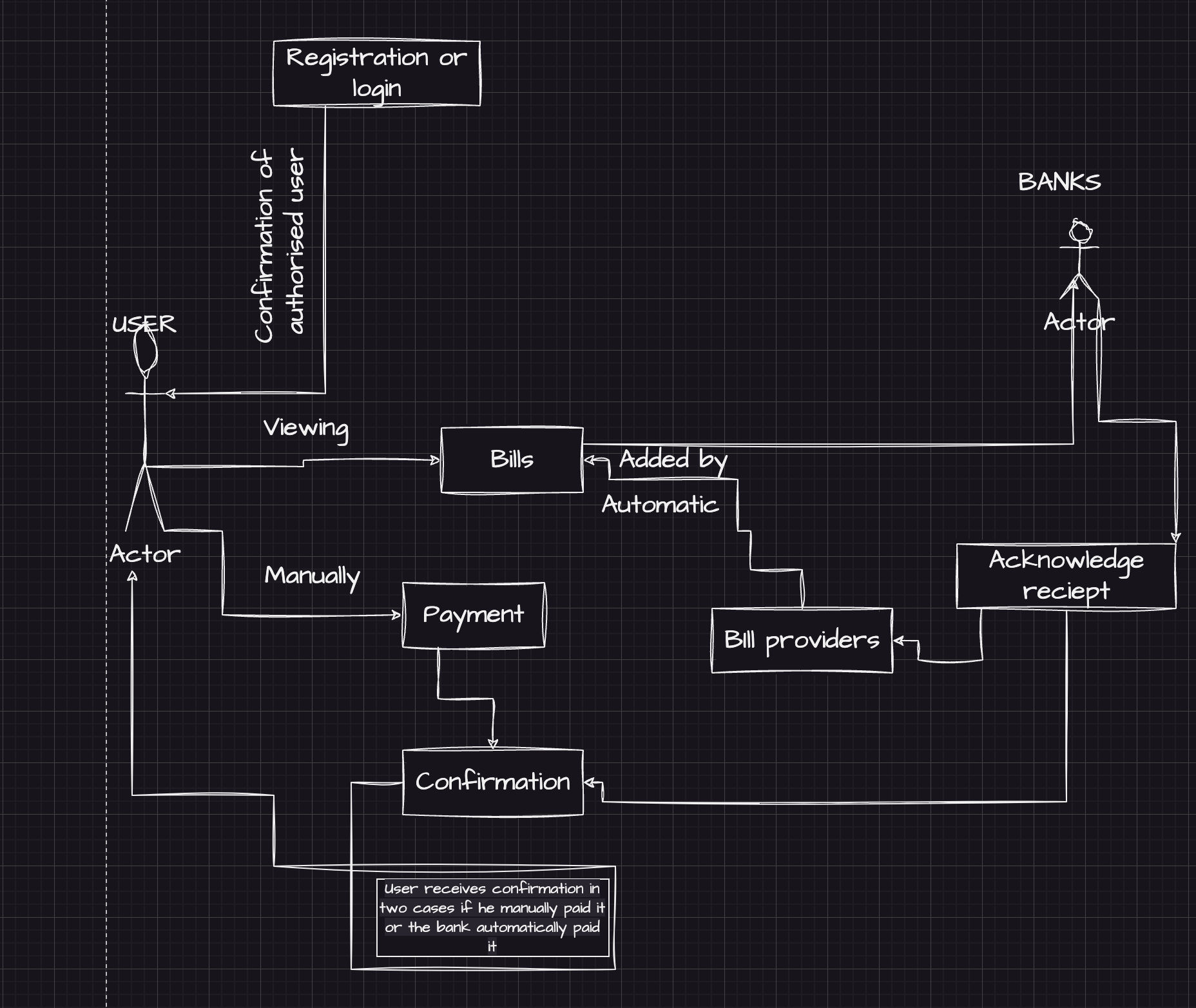
**For the Bill Provider:**

* Check the status of their bills and customers linked to it
* Check if the customer has missed any bill
* If there is, then update the database with the fine amount (ASSUMING THE PROVIDER WANTS TO FINE THE CUSTOMER ON MISSED PAYMENTS BY 2\* THE AMOUNT)
* Delete or view all the paid records of the customers.

**For the Banks:**

* View which bills are linked to their bank and to what source type
* Verify payments through acknowledgement

USE CASE DIAGRAM (ROUGH)



# Entity Relationship Diagram

***An Entity Relationship Diagram is a graphical representation of entities in a database and their relationship to one another. It is a visual tool for designing and modelling database structures.***

We came up with a Physical ERD in one of our Brainstorming sessions and decided that we wanted the following entities in our bill management database:

1. Customer/ User – This would be the entity that uses the system to manage and view their bills
2. Bill Providers – This entity would be the corresponding Bill Company that provides the bills to the user—for Example, The Power Grid Corporation of New Zealand.
3. Bills – This entity would represent the actual bills that the user receives. We would include all the necessary information like the bill type, amount, etc.
4. Bank/Payment Source – This entity would be the payment source used to pay off a specific type of bill.

The relationship between the bank's entity and the bills entity is pretty simple- They have one too many relationships. It is the same case with customers and bills. However, we have a many-to-many relationship with the bill providers and customers, which would cause an issue when implementing the database. This is why we added an associative entity that would solve this issue. This would now be called a Logical ERD diagram.

**PHYSICAL ENTITY RELATIONSHIP DIAGRAM**

*Diagram

Description automatically generated*

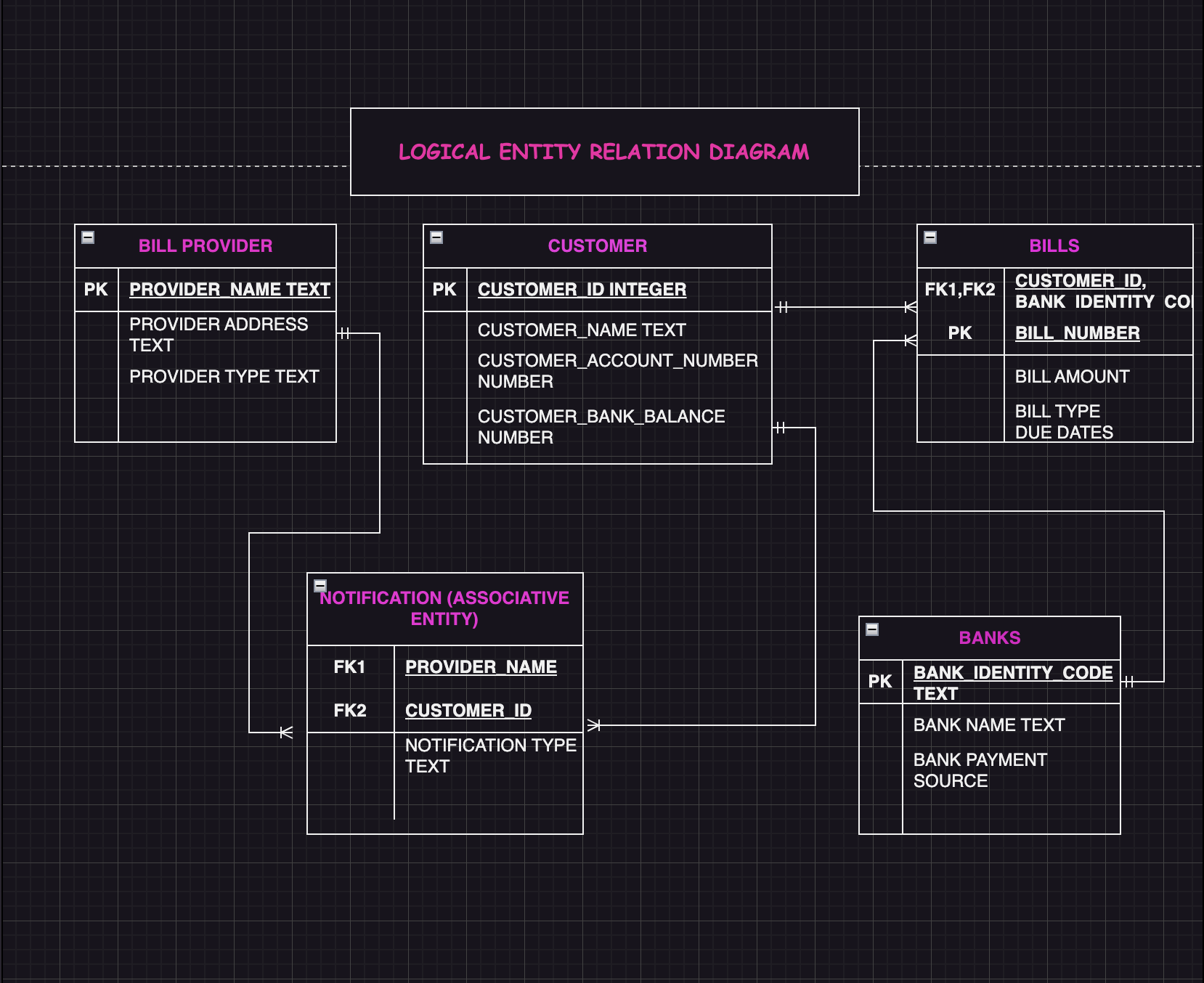
**LOGICAL ENTITY RELATIONSHIP DIAGRAM**

Our Logical ERD will entail the same entities. However, we have added a new entity to make it work in the real world.

1. **Notification Type (ASSOCIATIVE ENTITY) – This entity will have the primary key of both tables with many to many relationships. We have included another field that will show the payment status or the notification type, as we call it. It will show if a bill is “paid”,” due”, or” missed”.**

In the diagram below, we have shown every specific field that every entity has. We have also displayed the associative entity.

**You can also see the primary and foreign keys in the diagram below.**



# Table Designs – Data Dictionary

**This is our comprehensive Data Dictionary:**

**{TABLE 1: BILL PROVIDERS}**

Description: This is our Bill Providers Data Dictionary. This entity provides information regarding the various bill providers the user contacts. We have collaborated with the providers for the user to be able to see where their money is going clearly and to give them the freedom to switch or change their providers as they, please.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name | Entity Description | | | | |
| Bill Providers | **Providers information** | | | | |
| *Field Name* | ***Description*** | ***Data Type*** | ***Key***  ***Field*** | ***Constraints*** | ***Example*** |
| Providers\_  Name | **Providers identity information** | **Text** | **PK** |  | **Loong** |
| Providers\_  Address | **Address information of the provider** | **Text** |  |  | **35 Queen Street** |
| Provider Type | **The Category of Bill this company provides** | **Text** |  |  | **Water** |

**{TABLE 2: CUSTOMER}**

Description: This is our Customers Data Dictionary. This is the first entity created for the database. It holds all the customers currently enrolled with our BMS. We hold information regarding the user’s name, account number, and bank balance. We have asked the user for their account number to make the process of automatic transactions possible easily. The bank balance field lets the user keep track of their incoming and outgoing money. ALL IN ONE PLACE!

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity | Description | | | | |
| CUSTOMER | **This entity provides the customer’s name and their ID(inputs)** | | | | |
| Field Name | **Description** | **Data Type** | **Key**  **Field** | **Constraints** | **Example** |
| CUSTOMER\_ID | **Customer id number is unique number to identity the customer** | **Text** | **PK** |  | **234EYD5** |
| CUSTOMER\_NAME | **Customer name** | **Text** |  | **Text** | **AEH48** |
| CUSTOMER\_ACCOUNT NUM | **Account number is a modified number only known by customers to access their bank account** | **INTEGER** |  | **Integer** | **11110005782827** |
| CUSTOMER\_BANK BALANCE | **Credit score in their account** | **INTEGER** |  | **Integer** | **$2000** |

**{TABLE 3: BILLS }**

Description: This is our Bill's Data Dictionary. It is one of our Primary entities. This represents the actual physical bills in the database. This entity will hold all the information regarding the bills user must pay or has previously paid. This entity is the basis for showing the user's bill history or their current bills.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name | Entity Description | | | | |
| Bill | **Information regarding bills** | | | | |
| Field Name | **Description** | **Data Type** | **Key**  **Field** | **Constraints** | **Example** |
| Bills Number | **Unique bill number provided by the sender** | **Number** | **PK** |  | **234567** |
| Type | **Different bills identification** | **Text** |  |  | **Electronic or water** |
| Amount | **Total cost to be paid** | **Currency** |  | **Currency** | **$1234** |
| Due Dates | **Last date of Payments** | **Standard dates(dd/mm/yy)** |  | **Number** | **12/01/2023** |
| Customer\_ID | **Unique identification** | **Text and number** | **FK** |  | **12A78** |
| Source\_name | **Payment made through an institute** | **Text** | **FK** |  | **BNZ, ASB** |

**{TABLE 4: NOTIFICATIONS (ASSOCIATIVE ENTITY) }**

Description: This is our Associative entity Data Dictionary. This Entity was created due to the presence of a many to many relationship between bill providers and customer. This will also provide the user with a notification/ Status on their bill payment. It would be helpful for the user to see if he has paid or missed a certain bill provider’s invoice

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name | Description | | | | |
| Notifications | **This entity defines the notifications received by the user. This is an ASSOCIATIVE FIELD. It helps the user identify which bill is due, missed, unpaid etc.** | | | | |
| Field Name | **Description** | **Data Type** | **Key**  **Field** | **Constraints** | **Example** |
| Notification Type | **Type of the bill Notification** | **Text** |  | **Text** | **Due on 31st march 2014** |
| Customer\_ID | **Unique ID of the Customer/user** | **Text** | **FK** | **Number** | **1001(Customer Number)** |
| Bill Provider’s Name | **Name to identify the bill provider** | **Text** | **FK** |  | **Water Management of Auckland** |

**{TABLE 4: BANKS}**

Description: This is our Banks entity Data Dictionary. Its primary function is to provide information regarding the individual banks of the users. It also provides the specific bank payment method for each bank. This table is essential due to the existence of our bills entity. The primary key of this table is an essential aspect of the bills table as a foreign key.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Name | Description | | | | |
| BANKS/PAYMENT SOURCE | **This entity displays the various banks that the user has links with and the payment source used- for example “ANZ”,”DEBIT”** | | | | |
| Field Name | **Description** | **Data Type** | **Key**  **Field** | **Constraints** | **Example** |
| Bank\_identity\_code | **Unique code to identify the bank** | **Text** | **PK** | **Text** | **BOTA** |
| Bank\_name | **Name of the bank** | **Text** |  |  | **Bank of America** |
| Bank\_payment\_method | **Method used to pay money from the bank** | **Text** |  |  | **Debit Card** |

# 

# Contributions

**RIYA VISHNOI:**

1. Proposal Document – The primary Design, along with the internal Data
2. Final Project Report – Design along with data (The summary, design Analysis-all parts, 2 \* data dictionary)
3. Use cases along with individual Queries
4. Creating and Inserting data into the database
5. ERD (Physical and logical)

**MITCHELL:**

1. Proposal Document – Summary and internal Data,
2. Final Project Report –data (The summary, design Analysis, 1 \* data dictionary)
3. Use cases along with individual Queries
4. Creating and Inserting data into the database
5. ERD (Physical and logical)

**PRABHKIRAT:**

1. Proposal Document – The Introduction, along with the internal Data
2. Final Project Report –data (The summary, use cases, business-related input)
3. Use cases along with individual Queries
4. Creating and Inserting data into the database
5. Data Dictionary \* 1

**SHILLONG:**

1. Proposal Document – internal Data-(Target user and similar systems)
2. Final Project Report (user stories \* 2, 1 \* data dictionary)
3. Use cases along with individual Queries
4. Creating and Inserting data into the database

Note: \*\*Everybody took part in the data dictionary and also created tables and inserted values for their following data tables