|  |  |
| --- | --- |
|  | **Cognizant Academy**  **ElectricityBill Automation**  **Java Knock Out Challenge**  **Version 1.0** |
| |  |  |  |  | | --- | --- | --- | --- | |  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** | | **Name** |  |  |  | | **Role** |  |  |  | | **Signature** |  |  |  | | **Date** |  |  |  | |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc23440118)

[1.1 Purpose 3](#_Toc23440119)

[1.2 Definitions & Acronyms 3](#_Toc23440120)

[1.3 Project Overview 3](#_Toc23440121)

[1.4 Scope 3](#_Toc23440122)

[1.5 Target Audience 3](#_Toc23440123)

[1.6 Hardware and Software Requirement 4](#_Toc23440124)

[1.6.1 Hardware Requirements 4](#_Toc23440125)

[1.6.2 Software Requirements 4](#_Toc23440126)

[2.0 Functional Requirements 4](#_Toc23440127)

[2.1 Functional Requirements 4](#_Toc23440128)

[2.2 Use case Diagram 5](#_Toc23440129)

[2.3 System Architecture Diagram 6](#_Toc23440130)

[3.0 Design Specification 6](#_Toc23440131)

[3.1 Data Design 6](#_Toc23440134)

[3.2 Component Design for identified Use cases 7](#_Toc23440135)

[3.3 General Design Constraints 11](#_Toc23440136)

[4.0 Submission 13](#_Toc23440138)

[4.1 Code submission instructions 13](#_Toc23440139)

[5.0 Change Log 13](#_Toc23440140)

# Introduction

## Purpose of this document

The State Electricity Board was finding it tough to deal with the billing process of their domestic consumers. Validating the consumer number, making a note of the respective units consumed, calculating the bill amount based on the units consumed, and other processes were done manually which was very time consuming. So, the Electricity Board decides to out source the billing process of their domestic consumers to Global Tek Software Company. Help the Global Tek to automate the above task.

The Electricity Board has the following business processes that must be automated.

1. Parse data and generate a bill for all the customer
2. Store the ElectricityBill of each customer

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| Nill |  |
|  |  |
|  |  |
|  |  |

## Project Overview

This project captures the various concepts, techniques and skills learn and helps to put them into practice using Java with JDBC. Admittedly, this would be at a scaled-down level since the purpose is to let the associate experience the various concepts learned in Java as an individual. The individual associate is expected to carry out the knock out challange and complete it within 4 hours.

## Scope

The scope of the system is explained through its following modules

1. Parse data and generate a bill for all the customer
2. Store the ElectricityBill of each customer

## Target Audience

Learner Level

## Hardware and Software Requirement

### Hardware Requirements

|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
|  |  |  |
|  |  |  |
|  |  |  |

### Software Requirements

|  |  |  |
| --- | --- | --- |
| # | Item | Specification/Version |
| 1. | Java | 8 |
| 2. | MYSQL | 5.1 |

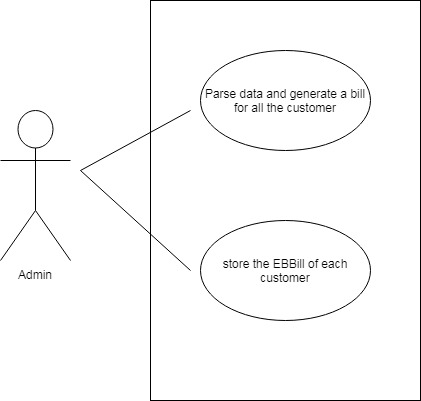
Note: All the required hardware and software isprovided in the TekStac platform

# Functional Requirements

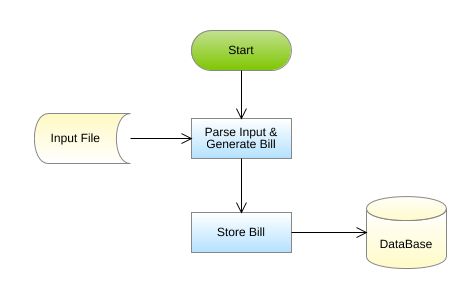
## Functional Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Req. # | Req. Name | Req. Description | Actors / Users | Comments |
| 1 | Parse data and generate bill for all the customer | The units consumed and the other details of the customer are stored in a flat file. Retrieve the data from the file and generate a bill for each customer based on the units consumed | Admin | The admin of the electricity board is reponsible for parsing the data and generating the bill for all the customers |
| 2 | Store the ElectricityBill of each customer | After generating the bill the electricity board will store the ElectricityBill of each customer in a database. | Admin | The admin of the electricity board is responsible for adding the ElectricityBill of each customer into the database |

## Use case Diagram



## System Architecture Diagram



# Design Specification

## Data Design

**Table Structure:**

|  |  |
| --- | --- |
| Table name: ElectricityBill | |
| Column Name | **Data type** |
| consumer\_number | varchar |
| name | varchar |
| address | varchar |
| units\_consumed | int |
| bill\_amount | float |

**Design Constraints:**

* Use MYSQL database to store the data. The database name should be “EBBill” and table name should be ElectricityBill.
* The above table has been created already. To create the table in your local machine, the script is available in “script.sql”, which will be provided as part of the code skeleton.
* The table names and the column names should be the same as specified in the table structure.
* Database connections should be configurable; it should not be hard coded. The database information is specified in the “db.properties” file, which is also provided as part of the code skeleton.

**Note:**The code skeleton is made available in the Tekstac platform. Skeleton incudes the script file. If working with Eclipse IDE,Copy and paste the script inside the script file into MYSQL editor so that the database, table with the required records are created.

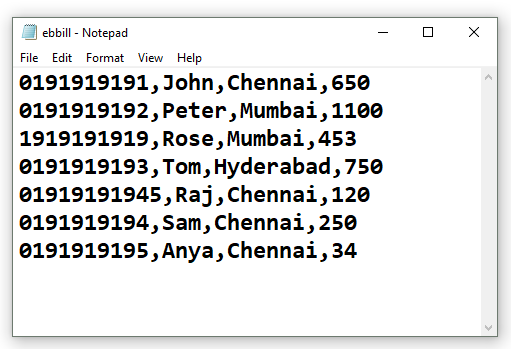
## Component Design for identified Use cases

### 3.2.1 Parse data and generate bill for all the customer

The units consumed and the other details of the customer are stored in a flat file. Retrieve the data from the file and generate bill for each customer based on the units consumed. The details of the customer are stored in a file named ElectricityBill.txt.

Sample File containing customer details. The file is comma delimited.

[ElectricityBillFormat: consumerNumber,Name,address,UnitsConsumed]



From the ElectricityBill.txt file, read the details , parse the data and construct an ElectricityBill object for each record in the file, then calculate the bill amount for each ElectricityBill based on theconditions given below:

|  |  |
| --- | --- |
| Units Consumed | Rate per unit in rupees |
| <=100 | Free(0) |
| >100 and <=300 | 1.5 |
| >300 and <=600 | 3.5 |
| >600 and <=1000 | 5.5 |
| >1000 | 7.5 |
|  |  |

**For example:** If the units Consumed is 650, then the first 100 units are free and for the next 200 units the charges are Rs. 1.50/unit, for next 300 units the charges are Rs.  3.50/unit and for the remaining 50 units the charges are Rs. 5.5 /unit. The total bill amounts to Rs.  1625(first 100 free then 200\*1.50 + 300\*3.50 + 50\*5.5)

After calculating the bill amount store all the EBill object into a list.

**Validation:**

The consumerNumber should start with zero and it should contain 10 digits. If the consumerNumber is valid then parse the data and calculate the bill amount else throw a user defined Exception “InvalidConsumerNumberException” with a message "Invalid Consumer Number".

**Note: This functionality is about only reading the records from the file, parsing each record data, validating the consumerNumber, creation of ElectricityBill object and then storing the ElectricityBill Object into the list and return the list. This functionality does not deal with DB Connectivity.**

**Component Specification: ElectricityBill(model class)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Attributes** | **Methods** | **Responsibilities** |
| Parse data and generate a bill for all the customers | ElectricityBill | String consumerNumber  String consumerName  String consumerAddress  int unitsConsumed  double billAmount | Include getters and setter method for all the attributes. |  |
| Parse data and generate a bill for all the customers | ElectricityBill |  | void calculateBillAmount() | This method should calculate and set the bill amount based on the units consumed. |

**ElectricityBoard(utility class)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Method** | **Responsibilities** | **Exception** |
| Parse data and generate a bill for all the customers | ElectricityBoard | List<ElectricityBill>generateBill(String filePath) | This method takes the file path as argument and it should parse the data stored in the fileand it should validate the consumer number by invoking the validate() method, if valid,construct a ElectricityBill object for each record in the file, then calculate the bill amount by invoking the calculateBillAmount method of ElectricityBill class. After calculating the bill amount, each ElectricityBill should be added into the list and this method should return the list of ElectricityBill. |  |
| Parse data and generate a bill for all the customers | ElectricityBoard | booleanvalidate(String consumerNumber) | This method should validate the consumerNumber, if valid return true else this method should throw an user defined exception | Throw a user defined exception “InvalidConsumerNumberException” if the consumer number is invalid. |

**Note:** The data file will contain both valid and invalid details. Valid ElectricityBill should be added to the list and for the invalid ones user defined exception should bethrown.

### 3.2.2 Store the ElectricityBill of each customer

After generating the bill ,the electricity board will store the ElectricityBill of each customer into the database.

**Note: Script file contains the DDL statement for creating the needed table in the database. This script is needed for implementing this functionality. So, when working with Eclipse, copy and paste the script in your local database and then implement the requirement and test your code.**

**Component Specification:ElectricityBill(model class)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Method** | **Responsibilities** | **Resources** |
| Store the units consumed by each customer | ElectricityBoard | void addBill(List<ElectricityBill>billList) | This method should add all the ElectricityBill details into the ElectricityBill table.  Connect to the database by invoking the establishConnection() method of DBHandler class. | MYSQL database is used. Store the details into ElectricityBill table |

**DBHandler(DAO class)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Method** | **Responsibilities** | **Resources** |
| Store the units consumed by each customer | DBHandler | Connection establishConnection() | This method should connect to the database by reading the database details from the db.properties file and it should return the connection object | MYSQL database is used. Store and retrieve the details into/from ElectricityBill table.  db.properties file is used to store the database configuration details. |

**Note: When working with Eclipse, please change the values of db.classname, db.url, db.username, db.password according to your MYSQL Configuration.**

## General Design Constraints

1. The attribute/method/class name should be correctly specified as given in the document.
2. Do not hardcode the database configuration details in the DBHandler class, read it from the db.properties file.

# Submission

## Code submission instructions

Do not change the code skeleton given, as your code will be auto evaluated.

You can validate your solution against sample test cases during the assessment duration.

Your last submitted solution will be considered for detailed evaluation.

Make sure to submit the solution before the specified time limit. You will not be allowed to submit the solution once the mention time for the assessment is over.

**No Sample Input/Output is provided as part of this document. This means that you will not be evaluated for any of the presentation related Requirements. You are free to write your own code in the main and to invoke the business method to check its correctness. main is not taken for evaluation.**

# Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Changes Made | | | |
| V1.0.0 | Initial baseline created on <dd-Mon-yy> by <Name of Author> | | | |
| Vx.y.z | <Please refer to the configuration control tool / change item status form if the details of changes are maintained separately. If not, the template given below needs to be followed> | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
|  |  |  |  |