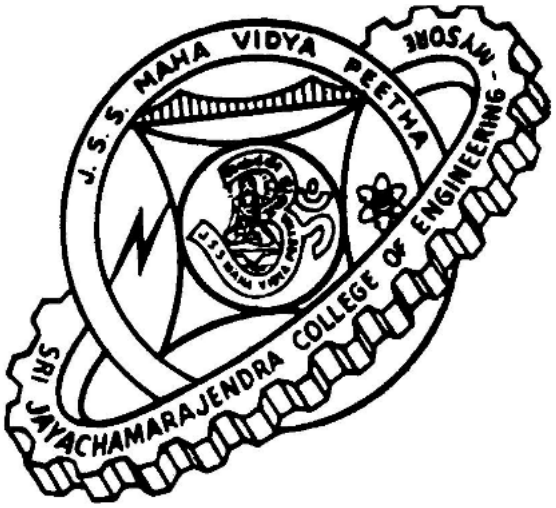
Sri Jayachamarajendra College of Engineering, Mysore - 570006

Department of Computer Science and Engineering



Software Design Specification Document for

Telephone Bill Generation Software for Telecom District

Dec 2009

THE TEAM

Name	Roll Number	USN
Vikram TV	59	4JC07CS120
Prabhakar Gouda	35	4JC07CS070
Sharad D	03	4JC06CS089

*5th Semester 'B' Section

Contents

1 Introduction

1.1 Purpose

This design document provides the design for implementing the **Telephone Bill Generation for Telecom District**. This includes the architectural features of the system and details of each schema used in the design.

1.2 Scope

To design efficient and user friendly online billing system for telecom district.
The dominant design methodology used would be *function oriented* with a *visual interface* for the database.
The billing records that are updated in the database are retrieved and the processing of the retrieved database is done mainly by the Visual interface.
After the processing is done, it is suitably outputted (as per user requirements) by a front-end.
All the customers access the database using the form based approach.
The administrator can access database either by forms or directly using database management systems like MySQL, PHPMyAdmin.

1.3 Glossary

TelBill	Telephone Bill generation System
Users	Administrator, Authorized Staff of Telecommunication Company, Customers
CustomerID	Unique key for identifying the customer in the database.
BillNo	Unique key in identifying each bill.
ReceiptNo	Unique key for identifying each payment of the bill.
logs	Call Logs, it can be received or dialed calls.

1.4 References

Software Engineering by Ian Sommerville, 6th Edition
BSNL Telephone Bill
BSNL Portal for Usage Check
IEEE Documentation Format for SDS

2 Software Lifecycle

The TelBill software lifecycle includes the following phases:

- Requirements
- Design
- Implementation
- Test
- Installation
- Operation and Maintenance

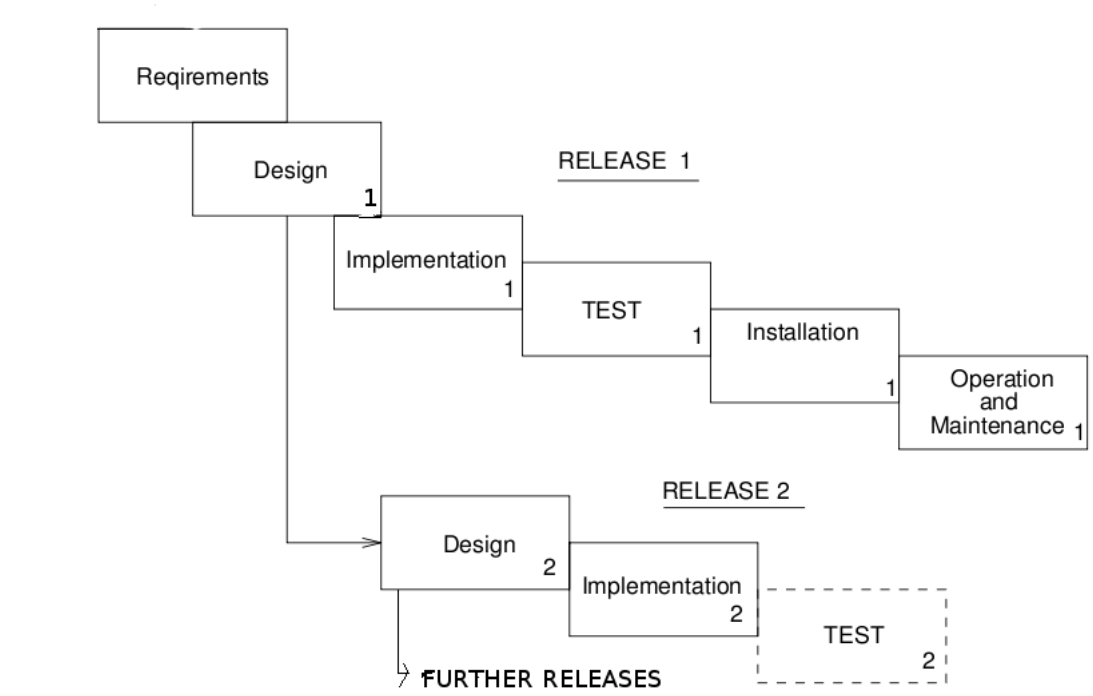


Figure 1: Life Cycle of TelBill

The software is done in multiple releases making use of the *spiral development model*. Each release is updated with new functionalities and features. Development activities starts from the design phase of previously released versions. The act of releasing depends on user suggestions and also on technology updations.

3 Architectural Design

The architectural design of the TelBill consists of identifying the various users and their representation in the database. Maintaining a centralised database for all the users simplifies the task of querying the database. The modules are created for database storage and are linked together by the foreign keys. The task of bill generation also includes the handling of customer information, phone numbers, meter readings, tariffs, call logs and finally the bill details and payment details. A login database also needs to be specified for security.

3.1 Overall System Organisation being Shared Data Repository

The shared centralised data in the database can be accessed by

- Administrator
- Clerks
- Technicians
- Customers
- Application Programs

Application programs that automate the process running in the database.

A centralised database is maintained to store the details of the telephone connections that are provided. It also includes the customer details, call logs, meter logs, bill amounts and payment details. The database access by the above users is as shown in the diagram.

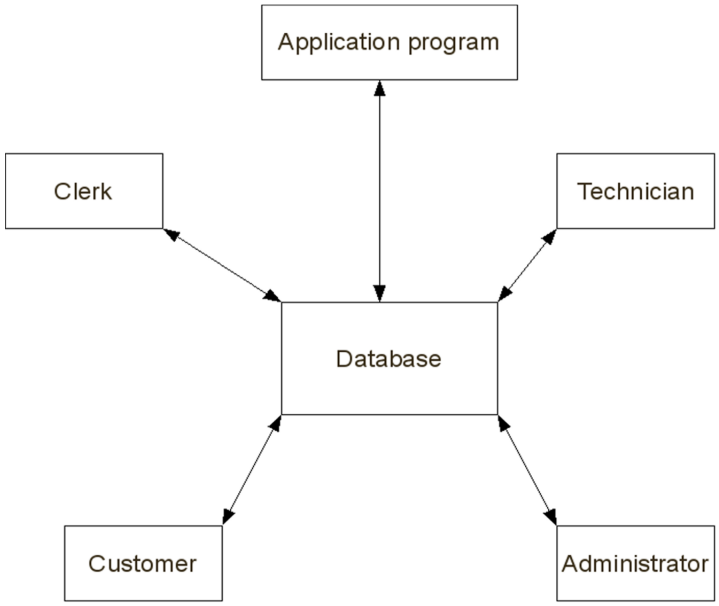


Figure 2: Shared Data Repository

3.2 Creation of Modules

A Unique ID is used to identify a customer in the entire database. Also bills and payment receipts are also identified by Unique Bill Number and Unique Payment Number.

The following nine modules are identified for the implementation of the TelBill:

Customer	Maintains the details of the customer like name and customer ID.
Login	Maintains the login details of the user
Address	Maintains the complete address of the customer.
Tariff	Maintains the plan opted by the customer. Bill is generated as per the tariff.

Phone Number	Maintains the phone numbers for each customer.
Call Log	Maintains the timestamp of the dialled and received calls.
Meter Reading	Maintains the monthly start and end meter readings for each customer.
Bill	Maintains the details of the generated bill like call charges, taxes and the final bill amount.
Payment	Maintains the timestamps of bill payments.

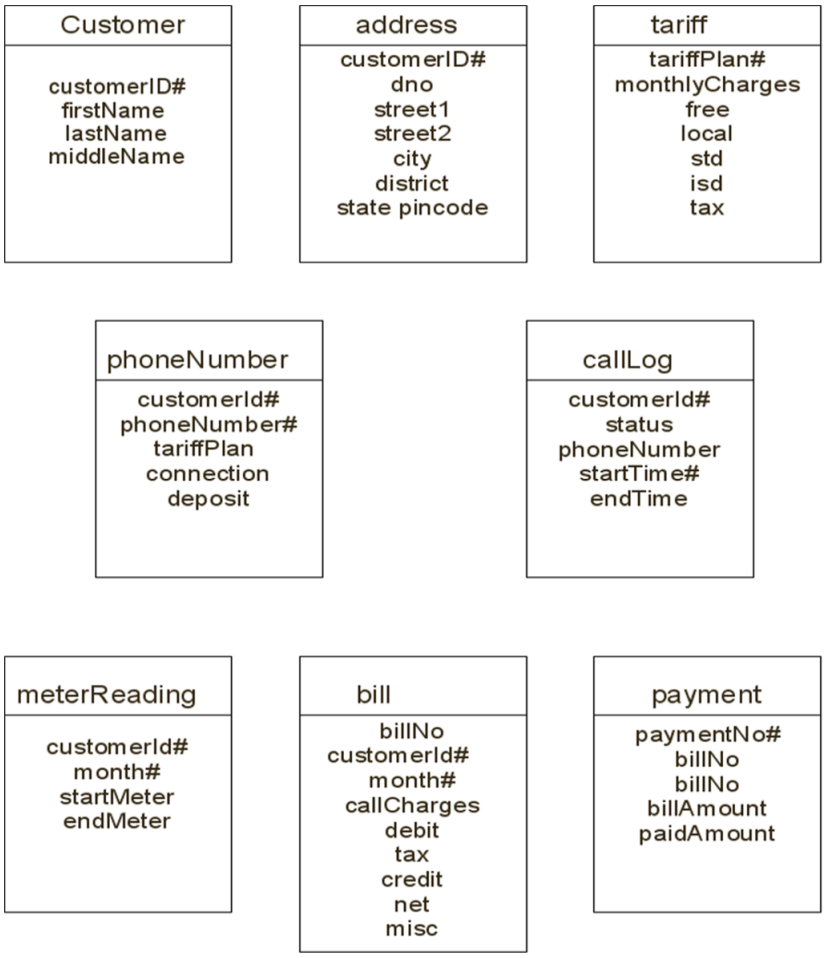


Figure 3: Only Modules No Connections

3.3 Cohesion of Modules

The modules are interconnected to allow interfacing among them. They are binded by refering to *foreign keys*.
The Entity - Relationship Model specifies the key constraints and the relation between tables.

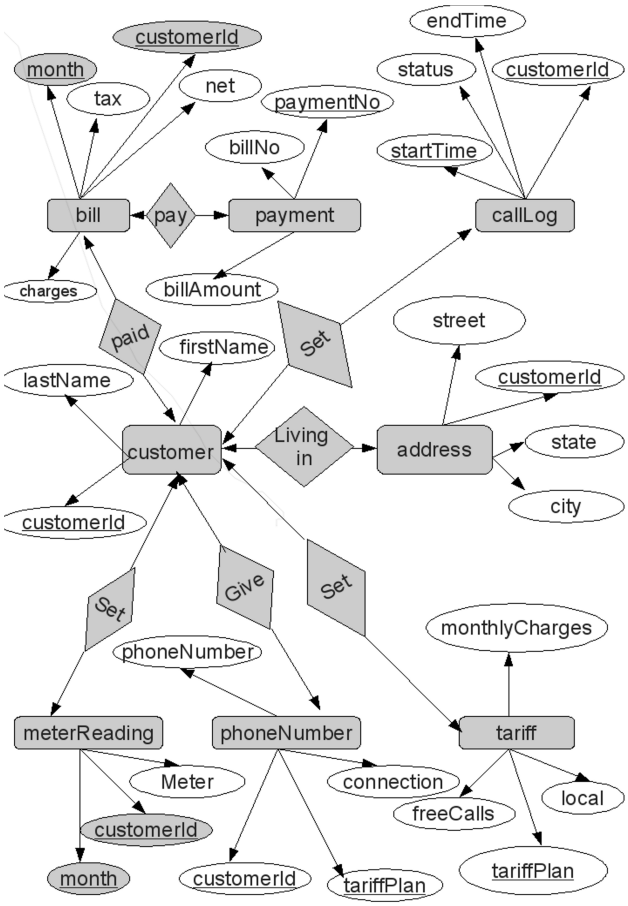


Figure 4: ER Model used in the design

The diagram describes the foreign key referenciations.

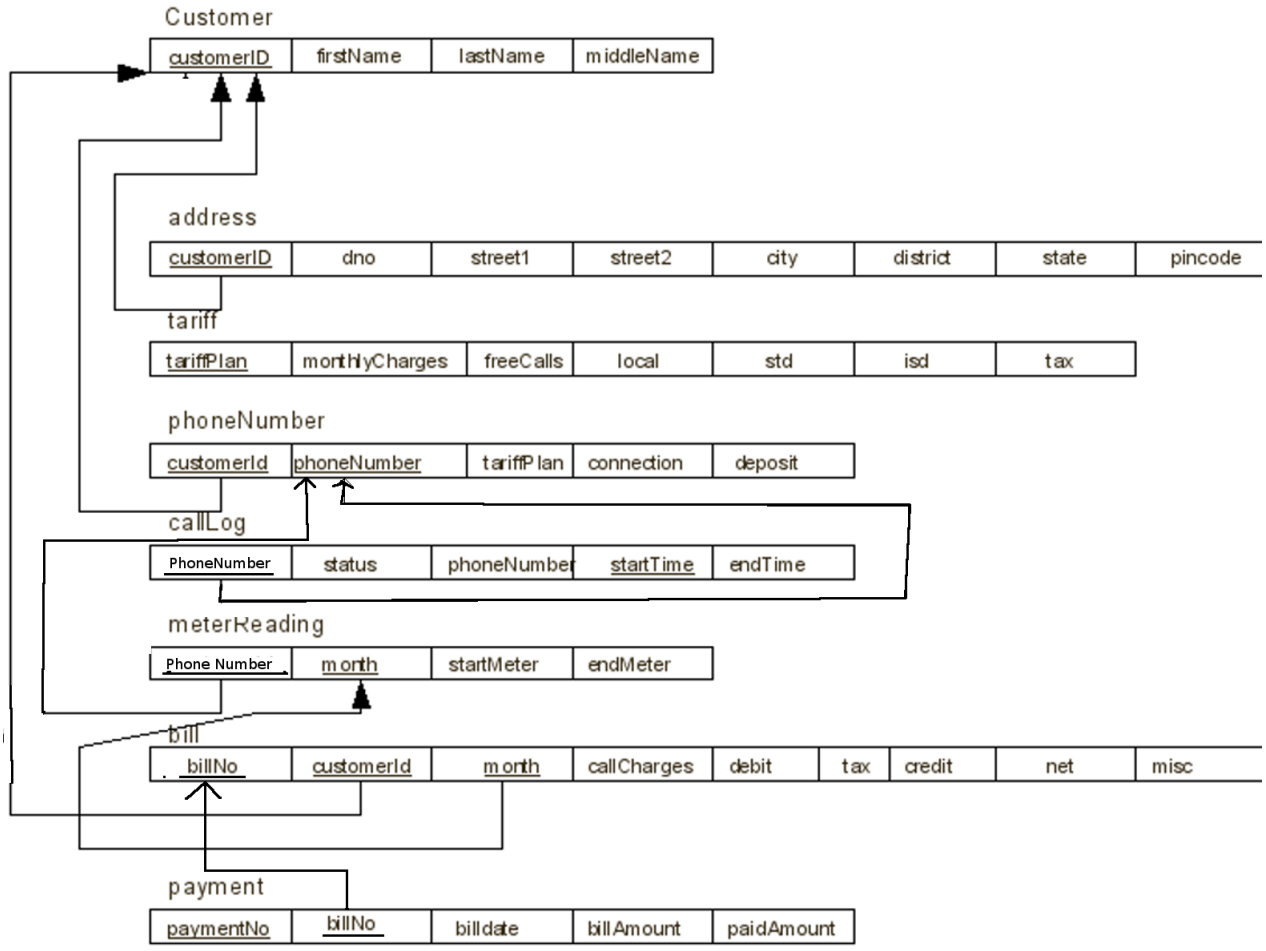


Figure 5: Connecting Modules

4 Detailed Design

This section consists of detailed design of the modules, that is the datatypes used to specify the attributes used in each modules.

4.1 Data types specifying the attributes

The following tables specifies the attributes in each modules. It also specifies the primary keys in each modules and the foreign key referenciations for other modules.

4.1.1 Module Customer

Attributes:

Customer ID	of integer and Auto Increment type, Unique ID for each customer
First Name	of variable string type
Middle Name	of variable string type
Last Name	of variable string type
Primary Key :	Customer ID
Foreign Keys :	NIL

4.1.2 Module Login

Attributes:

Username	of variable string type, length 10
User Password	of string type, length 64 to store passwords in encrypted format
Customer ID	of integer type
Primary Key :	Customer ID
Foreign Keys :	Customer ID → Customer ID of Module Customer

4.1.3 Module Address

Attributes:

Customer ID	of integer type
Door Number	of variable string type
Street 1	of variable string type
Street 2	of variable string type
City	of variable string type
District	of variable string type
State	of variable string type
Pincode	of integer type, length 6
Primary Key :	NIL (Weak entity)
Foreign Keys :	Customer ID → Customer ID of Module Customer

4.1.4 Module Tariff

Attributes:

Tariff Plan	of integer type, describes the type of plan
Monthly Charges	of float type
Free Calls	of integer type
Local Call Rates	of float type
STD Call Rates	of float type
ISD Call Rates	of float type

Tax Percentage	of float type
Primary Keys :	Tariff Plan

4.1.5 Module Phone Number

Attributes:

Customer ID	of integer type
Phone Number	of integer type, Unique ID for each Phone
Tariff Plan	of integer type, describes the type of tariff
Connection Date	of date type
Deposit	of float type, describes the deposit made by the customer
Primary Keys :	Customer ID, Phone Number
Foreign Keys :	Customer ID → Customer ID of Module Customer
	Tariff Plan → Tariff Plan of Module Tariff

4.1.6 Module Call Logs

Attributes:

Customer ID	of integer type
Phone Number	of integer type
Status	of character type, specifies Received Calls from Dialled Calls
Start Time of Call	of integer type
End Time of Call	of integer type

Primary Keys	:	Phone Number, Start Time
Foreign Keys	:	Customer ID → Customer ID of Module Customer
		Phone Number → Phone Number of Module Phone Number

4.1.7 Module Meter Reading

Attributes:

Phone Number	of integer type
Month	of integer type, holds month and year of meter reading
Start Meter Reading	of integer type
End Meter Reading	of integer type
Primary Keys :	Phone Number, Month
Foreign Keys :	Customer ID → Customer ID of Module Customer
	Phone Number → Phone Number of Module Phone Number

4.1.8 Module Bill

Attributes:

Bill Number	of integer type, Unique ID for each Bill that is generated
Customer ID	of integer type
Month	of integer type, holds month and year of bill generation
Call Charges	of float type
Debit	of float type, describes any previous debits from the customer

Tax	of float type, describes taxation for the gross amount of bill generated
Credit	of float type, describes any previous credits from the customer
Net Charges	of float type, contains the net amount of the bill
Miscellaneous Charges	of float type, describes any miscellaneous charges like installation charges, caller tune charges
Primary Keys :	Customer ID, Month
Foreign Keys :	Customer ID → Customer ID of Module Customer

4.1.9 Module Payment

Attributes:

Payment Number	of integer type, Unique ID for each receipts that are made
Bill Number	of integer type
Pay Date	of date type
Bill Amount	of float type
Payment Amount	of float type
Primary Keys :	Payment Number
Foreign Keys :	Bill Number → Bill Number of Module Bill

4.2 Sequence diagram for Data Collection

: The sequence diagram shows the three layers for data flow. They include the Front End, Processing and Database. The following activities takes place in the sequential manner.

- User enters Username and Password to the front end.

- Front end sends the username and password using the send() function to the processing layer and suspends itself.
- The processing layer requests the database to return the password for the requested username using requestPassword() function. After receiving the password from the database using sendPassword() function, it is checked with the password provided by the user using isValid() function.
- If password is valid then further querying is allowed else the error is displayed by the front end.

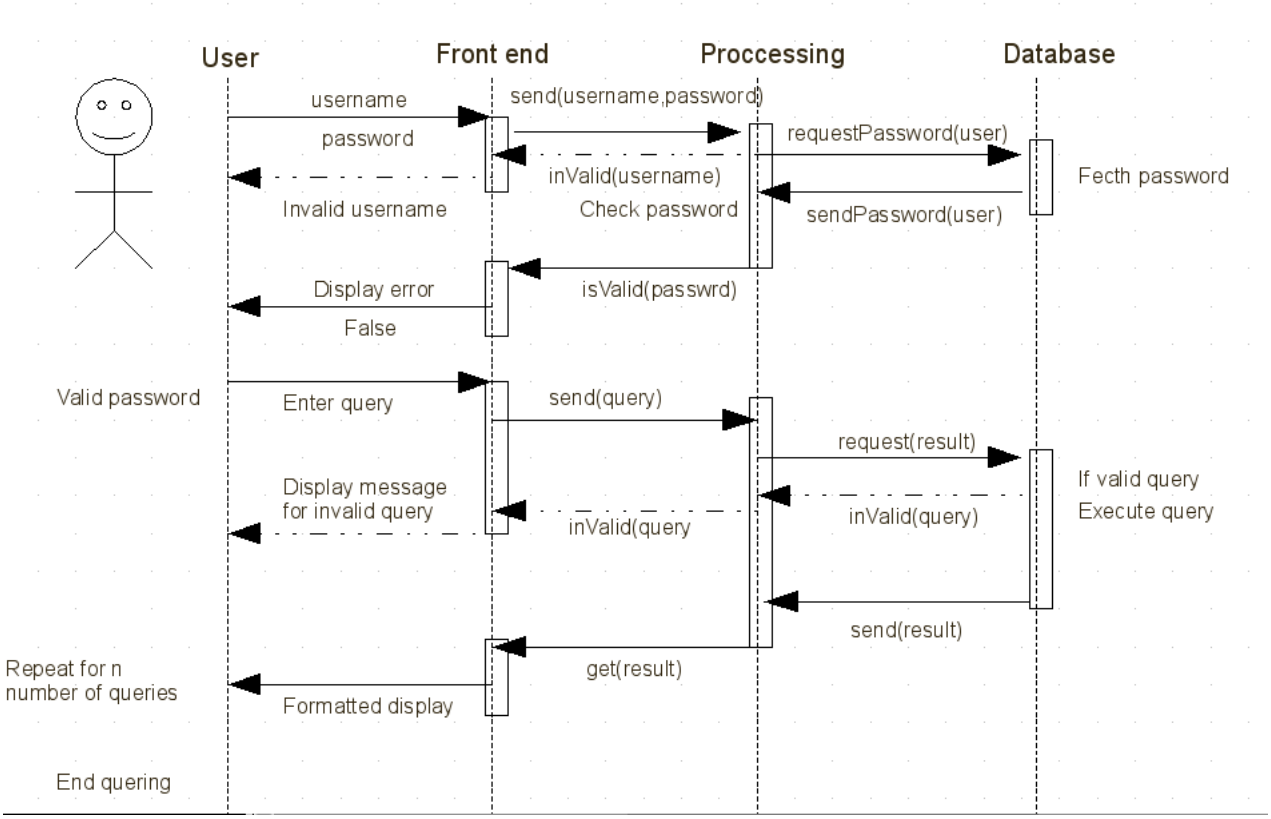


Figure 6: Data Collection Sequence Diagram

- User then queries the database using the front end like requesting the *month bill, profile.*

- These queries are also processed in the processing layer and result is requested from database using the request() function.
- Database returns invalid query if the query is incorrect using the inValid() function that gets displayed on the screen.
- Database executes valid query and returns result using the sendResult() function to the processing layer.
- Processing layer format the results and sends it to the front end using the getResult() function.
- Front end displays the result suitable and waits for further querying by the user.
- Querying gets terminated when the user logs out.

5 User Interface Design

The User Interface or the *frontend* supports for the user to interact with the database. The User Interface is designed to be online so that customers can also access the database. Hence there should be a design that supports both customers and the TelBill staff.

The following are the interfacing designs to be implemented:

- The passwords are encrypted (using the latest encryption algorithms - currently 'message-direct-5' or md5) and stored in the database module login.userpassword and a unique username in login.username.
- A welcome screen should greet all the users where a login prompt is also provided for users to login.

  Administrator login should support for handling the entire database, other TelBill staff login should support for operations and maintenance and Customer login should support for viewing the monthly bills and their profile.
- Administrator login should redirect to admin screen that has options to add user, customer, new connection; update users, customers, new connections; delete users, customers, new connections. Also to perform backups and maintenance facilities. Customize the entire environment.

The Administrator is reminded periodically to make backups. It also reminds to *clear* call logs and older bills.

- TelBill staff login should redirect to staff screen that has options to only handle the entries that gets generated from hardwares. Like they can view the call logs, update and view bill payments, backups, maintenance and also *handle customer queries*.
- Customer login should redirect to customer screen that has options to only view the bill and profile, make online payments. A drop-down boxes are to be provided to select the tariff plans, phone numbers (as a single customer can have multiple phone connections) and the month to view the *bill* that is generated.
- Gross bill is not stored in database and is *calculated dynamically* in the front end.
- A logout option is also provided for each user to safely disconnect from the database.
- If hard copy of the bills are lost, then they can be reproduced again using the TelBill.

6 Limitations

- The current design does not support for *password* retreival of any user, in case the password is lost.
- Billing for Value Added Services or VAS provided by the service provider is not incorporated.

The above features need to be incorporated in future releases.

7 Conclusion

A spiral development model is specified for the design and implementation of the TelBill. A shared data repository holds the entire database of all the interfaced modules that are accessed by all the users. An object oriented approach was found to be suitable to implement the TelBill. Finally, all users are provided with suitable screens to interface with the database.