



Key Performance Indicator System For Smart Technology

C. K. Nadeesha Kumari

BIT registration number: R092381

Index number: 0923818

Name of the Supervisor: Mr. D. M. D. Rushan Niroshana

2017



**This dissertation is submitted in partial fulfilment of the requirement of the
Degree of Bachelor of Information Technology (external) of the
University of Colombo School of Computing**

DECLARATION

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a Degree or Diploma in any University and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for inter-library loans, and for the title and summary to be made available to outside organizations.


.....

C. K. Nadeesha kumari
(Name of Candidate)

Date: 2017 November 08

Countersigned by:


.....

Mr. D. M. D. Rushan Niroshana
(Name of Supervisor)

Date: 2017 November 08

ABSTRACT

A Key Performance Indicator (KPI) is a measurable value that demonstrates how effectively a company is achieving key business objectives. PAYable use KPIs to evaluate their success at reaching targets. This project deals with developing Key Performance Indicator System for an existing cashless mobile payment solutions company.

The POS system facilitates for record merchant details and transaction records only. PAYable needs to maintain internal separate system for trace other details. Smart technology provides business solutions for PAYble as their Partner Company. The project objective is to deliver the PAYable Key Performance Indicator System for Smart Technology.

Measuring and monitoring business performance is critical, but focusing on the wrong key performance indicators can be detrimental. So can be poorly structured KPIs, or KPIs that are too difficult, costly to obtain, or to monitor on a regular basis. Proposed system covered the disadvantages of the existing systems and creating the faster and more accurate centralized data collection.

This project is an attempt to provide the information to top management for decision making. It helps Measuring and monitoring business performance.

ACKNOWLEDGEMENT

I take this space to express my gratitude to those who have helped me in a various ways to complete the project successfully

First I wish to express my gratitude to University of Colombo School of Computing for the valuable opportunity they have provided.

I must grateful to my supervisor, Mr. D. M. D. Rushan Niroshana for his enormous support and assistance in all the stages of the project completing with success and he also guide and corporate througho ut the project work.

Then I am indebted to Mr. G. Chamal Roshan Suraweera, the managing director of the Smart Technology and without whom it would not be a success, for the help he has given me in requirement gathering, requirement analysing and for the patience he had with all the meaning of success of my project.

I would like to thank Wide Awake Solutions and its panel of dedicated lectures for the knowledge and wisdom I have gathered while being a student of there.

I honestly thank all my friends who helped me a building this system and supported me in making this project successful.

I owe my deepest gratitude to my beloved family, who encouraging me and everything they have done for the success of the final outcome.

TABLE OF CONTENTS

	Page No
CHAPTER 1 - INTRODUCTION	1
1.1 . Introduction	1
1.2 . Problem Domain And Need of the Project.....	1
1.3 . Objectives of the Project.....	2
1.4 . Scope of the project	2
1.5 . User And System Requirement Specification	3
1.5.1. Administrator's module	3
1.5.2. Manager's module	3
1.5.3. Sales Module.....	3
1.5.4. Back office module	4
1.5.5. Customer Support team module.....	4
1.5.6. Merchants and Transaction module	4
1.6 . Structure of the Dissertation	5
1.6.1. Chapter 01 – Introduction	5
1.6.2. Chapter 02 – Analysis	5
1.6.3. Chapter 03 – Design.....	5
1.6.4. Chapter 04 – Implementation.....	5
1.6.5. Chapter 05 – Evaluation.....	5
1.6.6. Chapter 06 – Conclusion.....	5
1.6.7. Appendixes	5
CHAPTER 2 - ANALYSIS.....	6
2.1. System Analysis	6
2.2. Existing System.....	6
2.3. Requirements Gathering Techniques	8
2.3.1. Sampling of existing documentation	8
2.3.2. Interviews.....	9
2.3.3. Observations of the work environment.....	9
2.4. Functional Requirements.....	9
2.5. Non-Functional Requirements	10
2.5.1. Security	10
2.5.2. Performance	10
2.5.3. Reliability.....	10

2.5.4.	Usability	10
2.5.5.	Maintainability	10
2.5.6.	Extensibility	10
2.6.	Process Model	11
2.6.1.	Inception	11
2.6.2.	Elaboration	11
2.6.3.	Construction	11
2.6.4.	Transition	12
CHAPTER 3 -	DESIGN	13
3.1.	Alternative Solutions	13
3.1.1.	Standalone System	13
3.1.2.	Web Based System	13
3.2.	System Design Process.....	13
3.3.	Architecture of the System.....	14
3.4.	Use Case Diagram	15
3.5.	Use Case Narratives	16
3.6.	Class Diagram	17
CHAPTER 4 -	IMPLEMENTATION	18
4.1.	Introduction	18
4.2.	Implementation Environment.....	18
4.3.	System Developed Tools and Technologies	19
4.4.	Major Code Segments	19
4.4.1.	Database configure page	19
4.4.2.	Session handling page.....	21
4.4.3.	Logout	23
4.5.	Re-usable Components.....	23
CHAPTER 5 -	EVALUATION	24
5.1.	Introduction	24
5.2.	Software Testing	24
5.3.	Software Testing Methods	24
5.3.1.	Black box Testing	24
5.3.2.	White box testing	25
5.4.	Types of Testing.....	25
5.4.1.	Unit Testing	25

5.4.2.	Integration Testing	25
5.4.3.	System Testing	26
5.4.4.	Functional Testing	26
5.4.5.	Non-Functional Testing	26
5.4.6.	Regression Testing	26
5.4.7.	Usability Testing	27
5.5.	System Test Plan	27
5.6.	System Test Cases	29
5.6.1.	Test Cases for Login	29
5.6.2.	Test Cases for User Management	30
5.6.3.	Test Cases for Call Centre View	32
5.6.4.	Test Cases for Merchants	32
5.6.5.	Test Cases for Target Setting	33
5.6.6.	Test Cases for Total Count	33
5.6.7.	Test Cases for Summary	34
5.7.	User Acceptance Testing	35
CHAPTER 6 - CONCLUSION		36
6.1.	Introduction	36
6.2.	Future Improvements	36
6.3.	Lessons Learnt	37
REFERENCES		38
Appendix A - System Documentation		40
	Hardware requirements	40
	Software requirements	41
Appendix B - Design Documentation		43
	User Interface Design	43
	Login Interface	43
	Summary Page Interface	44
	Merchant Details Interface	44
	Transaction Details Interface	45
	Monthly Target Setup Interface	45
	Total Counts Interface	46
	Reports	46
	Graphs	47

Call Center View.....	49
Database Designing for the System	50
Appendix C - User Documentation	52
User Login	52
Add New User.....	53
Add New Permission	53
Add New Role.....	54
Set Monthly Transactions Target.....	54
Set Monthly Deployments Target.....	55
Call Center View.....	55
Total Count	56
Appendix D - Management Reports.....	57
Summary Report	57
Daily Users.....	57
Daily Deployments	58
Daily Transaction Average Graphs.....	58
Daily Transaction Graphs	59
Appendix E - Test Results	60
Test Results for Login.....	60
Test Results for User management	61
Test Results for Call Center View	63
Test Results for Merchants	63
Test Results for Targrt Setting.....	63
Test Results for Total Count	64
Test Results for Summary.....	64
Appendix F - Code Listing	66
Code for login Module.....	66
Code for Summary Module	68
Code for Transactions	71
Code for Merchants.....	71
Code for Monthly Summary	71
Appendix G - Client Certificate.....	72
Index.....	73

LIST OF FIGURES

	Page No
Figure 2.1: Seylan Bank Web Portal.....	6
Figure 2.2: Commercial Bank Web Portal.....	7
Figure 2.3: Zoho CRM.....	7
Figure 2.4: Google Sheets.....	8
Figure 2.5: Rational Unified Process Model.....	11
Figure 2.6: Rational Unified Process Model.....	12
Figure 3.1: Three-tier architecture	14
Figure 3.2: Use Case Diagram	15
Figure 3.3: Class Diagram	17
Figure 5.1: User Evaluation Form	35
Figure B.1: Interface - Login	43
Figure B.2: Interface – Summary Page.....	44
Figure B.3: Interface – Merchant Details	44
Figure B.4: Interface – Transaction Details	45
Figure B.5: Interface – Monthly Target Setup.....	45
Figure B.6: Interface – Total Counts	46
Figure B.7: Interface – Daily Deployments	46
Figure B.8: Interface – Monthly Deployments	47
Figure B.9: Interface – Monthly Target Achievements	47
Figure B.10: Interface – Average Transaction Values.....	48
Figure B.11: Interface – Growth Chart.....	48
Figure B.12: Interface – Call Center View	49
Figure B.13: Database Design for the System	50
Figure B.14: Database Design for User Management	51
Figure C.1: Client Registration	52
Figure C.2: Add New User	53
Figure C.3: Add New Permission	53
Figure C.4: Add New Role	54
Figure C.5: Set Monthly Transactions Target.....	54
Figure C.6: Set Monthly Deployments Target.....	55
Figure C.7: Call Center View	55
Figure C.8: Total Counts.....	56
Figure D.1: Summery Report.....	57
Figure D.2: Daily Users	57
Figure D.3: Daily Deployments	58
Figure D.4: Daily Transaction Average Graphs	58
Figure D.5: Daily Transaction Graphs.....	59

LIST OF TABLES**Page No**

Table 3.1: Use Case Narrative (Login)	16
Table 3.2: Use Case Narrative (Logout)	16
Table 4.1: Implementation Environment	18
Table 5.1: High Level Test Plan	29
Table 5.2: Test Cases - Login	30
Table 5.3: Test Cases - User management	32
Table 5.4: Test Cases - Call Center View	32
Table 5.5: Test Cases - Merchants	32
Table 5.6: Test Cases - Target Settings	33
Table 5.7: Test Cases - Total Count	33
Table 5.8: Test Cases - Summary	34
Table A.1: Hardware Requirements	40
Table A.2: Software Requirements	41
Table E.1: Test Results - Login	60
Table E.2: Test Results – User Management	62
Table E.3: Test Results – Call Center View	63
Table E.4: Test Results – Merchants	63
Table E.5: Test Results – Target Setting	64
Table E.6: Test Results – Total Count	64
Table E.7: Test Results – Summary	65

LIST OF ACRONYMS

CRM	- Customer Relationship Management
KPI	- Key Performance Indicator
POS	- Point of Sales
RUP	- Rational Unified Process
UML	- Unified Modelling Language
WWW	- World Wide Web

CHAPTER 1 - INTRODUCTION

1.1 . INTRODUCTION

PAYable is a free mobile app paired with a free Bluetooth cardreader that enables merchants to accept credit and debit card payments from their customers. A highly reliable, durable and compact card reader, PAYable can be paired with any Android or iOS smart phone or tablet to effectively convert it into a secure, flexible and simple to use mobile Point-of-Sales (POS) solution. The revolutionary PAYable cashless mobile payments platform has gained the trust and support of a diverse cross-section of Micro, Small and Medium Enterprises (MSMEs) and entrepreneurs and has become the leading choice for major banks.

Bank portals support for recording merchants and Transaction related details only. Payable used many data sources (PAYable web portal/ Zoho CRM/ Google sheets/ Reporting portal and corporate portal/ Inventory System and Drop Box) to identified their business performance and they planning to move on new internal system for Centralizes businesses.

Smart Technology provides business solutions for PAYable as their Partner Company. This project is a web based KPI system for a cashless mobile payment solutions company named as PAYable. The project objective is to deliver the PAYable Key Performance Indicator System for Smart Technology.

1.2 . PROBLEM DOMAIN AND NEED OF THE PROJECT

PAYable cashless mobile payments platform has gained the trust and support of a diverse cross-section of Micro, Small and Medium Enterprises (MSMEs) and entrepreneurs and has become the leading choice for major banks. Payable launched its first devices in June 2016, has since grown the number of Sri Lankan merchants accepting card payments by 25% in just over a year, and has directly facilitated transactions in excess of Rs. 750 million.

Due to the exponential growth of Merchants and Transactions they are planning to move on the new internal KPI system for Centralizes businesses. They used many data sources (PAYable web portal/ Zoho CRM/ Google sheets/ Reporting portal and corporate portal/ Inventory System and Drop Box) to identify their business performance. Measuring and monitoring business performance is critical, but focusing on the wrong key performance indicators can be detrimental. So can be poorly structured KPIs, or KPIs that are too difficult, costly to obtain, or to monitor on a regular basis.

In order to solve this, proposed system provides the information to top management for decision making. It helps Measuring and monitoring business performance and Centralizes all data sources.

Using the proposed KPI System, Visualize and comprehend data from a number of KPIs that represent different areas of a business, all in one place. Proposed system covered the disadvantages of the existing system and tracking the information much easier and helping to make decisions fast.

1.3 . OBJECTIVES OF THE PROJECT

The main goal of this project is to provide a PAYable Key Performance Indicator System enables to create, manage and analyse data from KPIs. The objectives of the project are as follows;

- Measuring and monitoring business performance.
- Centralizes businesses data, while simplifying real-time reporting to always give them a competitive edge.
- Visualize and comprehend data from a number of KPIs that represent different areas of a business, all in one place.
- Faster and more accurate centralized data collection.
- Instant reports on performance.
- Increases data visibility.
- Easy to use.

1.4 . SCOPE OF THE PROJECT

The Project Scope pertains to the work necessary to deliver a product. Requirements and deliverables define the project scope. The Proposed system will provide an overall solution to handle a KPI system, helps the users to direct guidance through the total process and which supports the functions of PAYable. The Proposed system will execute the project as follows:

- Develop PAYable Key Performance Indicator system.
- The Proposed system will helps to Measuring and monitoring business performance and centralizes all data sources.
- Bank System has provided to Merchant Details and Transaction details through Secure channel
- Data Entry Dashboard
- Real-time reports
- Retrieve data using existing systems
- Trace Team performance
- Trace Individual Performance
- Define most appropriate KPIs.
- Define User access levels
- Users of each level can be accessed to the system with relevant user privileges
- Daily reports, monthly reports can be generated easily.

1.5 . USER AND SYSTEM REQUIREMENT SPECIFICATION

Given below are the modules of the system.

1. Administrator's module
2. Manager's module
3. Sales Module
4. Back office module
5. Bank User's Module
6. Customer Support team module
7. Merchants and Transaction module (Syncing data from Bank portals)

All the users can modify some of their profile details and change password.

1.5.1. Administrator's module

- Log In
- Log Out
- Create a profile
- Delete profile
- Modify access levels
- Create Roles
- Delete Roles
- Modify user details
- View Transaction details
- View merchant details
- Access lead details
- Access merchant Account Details
- Access Reports
- Create sources

1.5.2. Manager's module

- Log In
- Log Out
- View Reports

1.5.3. Sales Module

- Log In
- Log Out
- Add Leads
- Edit Leads
- Create sources

- View sales related KPIs
- Leads Send to verification
- View Reports

1.5.4. Back office module

- Log In
- Log Out
- Add Leads
- Edit Leads
- Create sources
- Upload required documents
- Lead details verification
- Documents Send to bank
- Add Merchant Account details
- Edit Merchant account details
- View Reports

1.5.5. Customer Support team module

- Log In
- Log Out
- Add Leads
- Edit Leads
- Create sources
- Lead details verification
- Edit Merchant Account details
- View Reports

1.5.6. Merchants and Transaction module

- View Transaction details
- View merchant details
- View Reports

1.6 . STRUCTURE OF THE DISSERTATION

1.6.1. Chapter 01 – Introduction

The Introduction chapter describes about the motivation for the project. Then the project objectives and scope are described. Provides all the background information, which will be useful in understanding the system. Then brief introduction about content of the dissertation report.

1.6.2. Chapter 02 – Analysis

The Analysis chapter describes about the requirement gathering and analyzing techniques that had been used in the project and discussed existing similar systems. Describe about the project goal & system environment. Detailed functional & non-functional requirements are documented in this chapter.

1.6.3. Chapter 03 – Design

The Design Chapter describes about the system designing with reference to certain design diagrams and it further describes the design approach. Used and designing of the data base as well as the user interfaces.

1.6.4. Chapter 04 – Implementation

The Implementation chapter describes the implementation of the project. It described implementation Environment and what are the activities that were carried out during the development and what are the used code and module structures and also described running Environment.

1.6.5. Chapter 05 – Evaluation

The Evaluation chapter describes about the testing strategies and criteria for the project. It also describes the Objectives, Scope and the approaches to the software testing process. It also indicates the personnel who hold responsibility for each task and also specifies the risks associated with the test plan.

1.6.6. Chapter 06 – Conclusion

The Conclusion chapter describes critical evaluation of the system and suggestions for any future work and it also includes lessons learnt during this project.

1.6.7. Appendixes

After the main chapters the Appendixes, it includes further information that is not essential to be included in the main chapters

CHAPTER 2 - ANALYSIS

2.1. SYSTEM ANALYSIS

The system's services, constraints and goals are established with the consultation with the users. This would include the understanding of the information domain for the software, functionality, behavior, performance, interface, security and exceptional requirements. These requirements are then specified in a manner which is understandable by both users and developer.

Systems analysis is an important activity that takes place when new information systems are being built or existing ones are changed. It is concerned with an in-depth study of the existing system to determine how it functions and how well it meets the current needs of the end users. The result of this stage builds a common understanding between the end user and the system analyst concerning the functional requirements of the business. The most crucial role of system analysis is in defining user requirements.

2.2. EXISTING SYSTEM

Bank portals support for recording merchants and Transaction related details only. Payable used many data sources (PAYable web portal/ Zoho CRM/ Google sheets/ Reporting portal and corporate portal/ Inventory System and Drop Box) to collect information and proceed manual reports for identified their business performance.

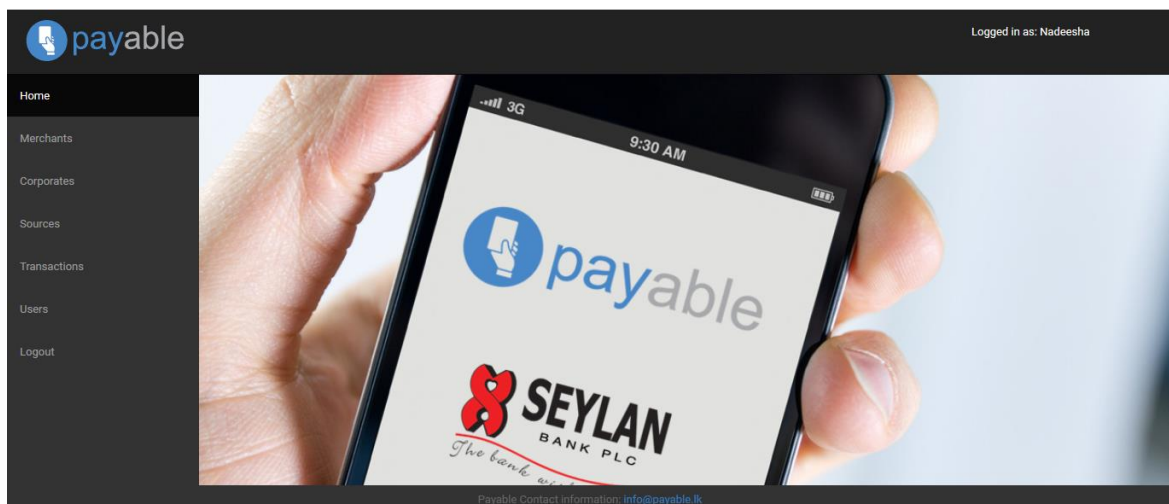


Figure 2.1: Seylan Bank Web Portal

Seylan Bank web portal used to collect Merchant and Transaction details of Seylan Bank users.

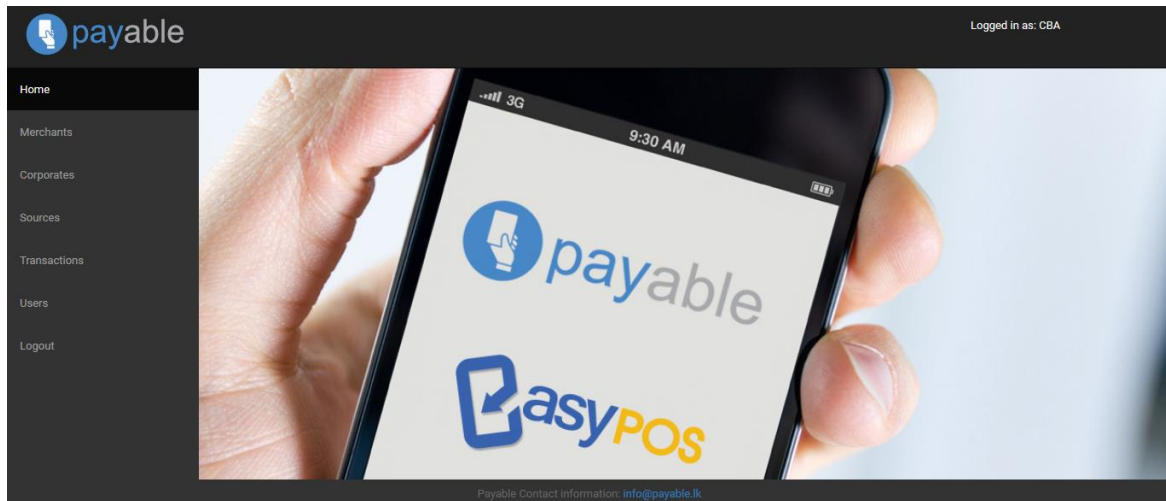


Figure 2.2: Commercial Bank Web Portal

Commercial Bank web portal used to collect Merchant and Transaction details of Commercial Bank users.

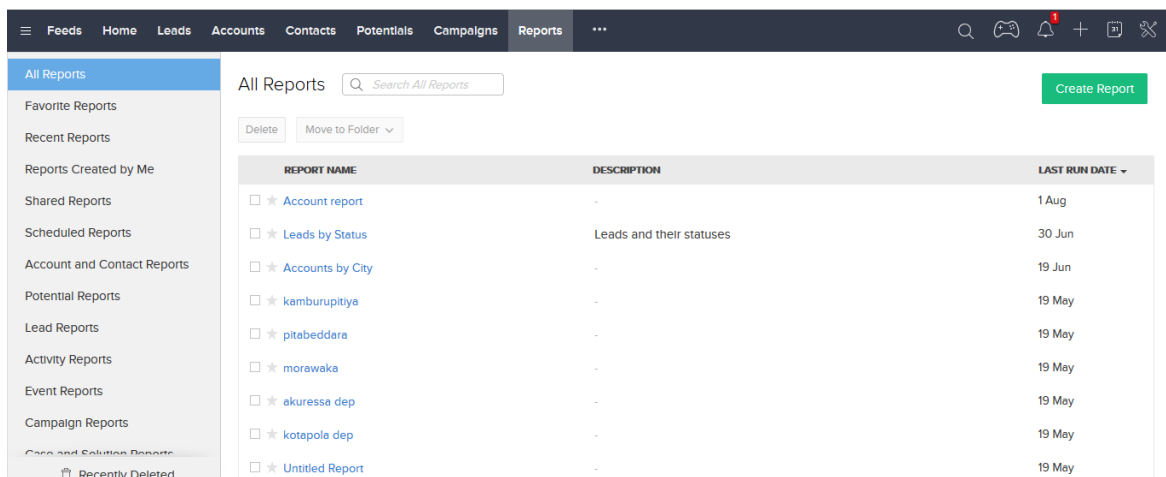


Figure 2.3: Zoho CRM

Zoho CRM used to Enter and collect Promotion Details, Lead and Account details.

Drop Box used to Store Scanned applications and Related Merchants Documents.

TID	MID	MERCHANT NAME	ADDRESS	PHONE NO	STATUS	AGENT	Reader ID	READER ISSUED	Reader Assigned	OS
20000001	100001	PAYable Testing	PAYable		TESTING	PAYable		Yes		
20000001	100001		789, Pannipitiya Road, Pelawatta, Battaramulla	777060477	ACTIVATED	PAYable	19913097216011900006	Yes		Android
20000002	100002	PAYable Testing	PAYable		TESTING	PAYable		Yes		
20000002	100002	ASIAN HOME APPLIANCES (PVT) LTD	NO. 12, Hultsdorf Street, Colombo-12	777353019	DEACTIVATED	PAYable	reader collected	No		Android
20000003	100003		PAYable		TESTING	PAYable		Yes		
20000003	100003	CRYSTAL ENTERPRISES	3/1B, P.T De Silva Mawatha, Dehiwala	722429691	ACTIVATED	PAYable	19313297215121400204	Yes		Android
20000004	100004	IP INTERNATIONAL	NO. 166, GNANENDRA MW, NAWALA	773877451	ACTIVATED	Seylan Bank	19213214416020100013	Yes		iOS
20000005	100005	THAKSALA THOGA VELADHA SELA	NO. 12, SUPERMARKET, MAHARAGAMA	716160744	DEACTIVATED	Seylan Bank		No		
20000006	100006	WASANTHA MOTOR STORES	NO. 41, COLOMBO ROAD, AVISSAWELLA	714173743	ACTIVATED	Seylan Bank	19813134915092900011	Yes		Android
20000007	100007	INDIAN LOOK	NO. 39 A, HILL STREET, DEHIWALA	722925706	ACTIVATED	Seylan Bank	19213214416020100183	Yes		Android
20000008	100008	YASHODHA OPTICIANS	NO. 760/B, PANNIPITIYA ROAD, THALAWATHUGODA	715351691	ACTIVATED	Seylan Bank	19313297215121400188	Yes		Android
20000009	100009	ABHARANA LK	31, Parakrama Road, Mattumagala, Ragama	728827499	ACTIVATED	PAYable	09113309115100900030	Yes		Android
20000010	100010	TAPROBANE ART	30/2, Senanayake Mawatha, Nawala	777268732	ACTIVATED	PAYable	19813134915092900012	Yes		Android
20000011	100011	SANJU SUPERMARKET	15, Gemunu Mawatha, Borupana Road, Moraruwa	717644123	DEACTIVATED	PAYable		No		
20000012	100012	MADISON ENVY	201/B, Havelock Road, Wellawatte, Colombo-06	772515933	ACTIVATED	PAYable	19313297215121400269	Yes		Android

Figure 2.4: Google Sheets

Google sheets used to view Merchant status.

Collect data using above mentioned sources and Prepare manual reports as per need. This is very time consuming work. There is no proper way to find the summarized details of KPIs.

2.3. REQUIREMENTS GATHERING TECHNIQUES

Requirements gathering process is a very important task in requirement analysis. Fact gathering is a formal process to collect information about;

- System requirements
- Problems
- Preferences

Correct system can only be built if analysts know what the user needs and what the system must do. Therefore the most important factor in building correct system is to first clearly define what the system must do and have a better communication with between system users. Using the many requirements gathering techniques as follows;

2.3.1. Sampling of existing documentation

Sampling is the process of collecting a representative sample of documents, forms and records. Studying the existing system use sampling to determine what can happen in the system.

2.3.2. Interviews

Collect requirements from individual users by talking to face to face which helps to identify these individuals and their responsibilities.

2.3.3. Observations of the work environment

Watches a person perform activities to learn about the system and validity of data obtained directly from individuals.

2.4. FUNCTIONAL REQUIREMENTS

Functional requirements describe functionality or services that the system is expected to provide. These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.

- Provide necessary Dash boards to data entry
- Facilitate to enter Leads.
- Facilitate to reject Leads.
- Provide facility to approve Leads.
- Provide user name and password to sign in PAYable Key Performance Indicator System.
- Facilitate to Update Accounts.
- Provide different access level to the users according to the level of authority in order to control the system.
- Provide facility to Uploaded required Documents
- Provide facility to View Uploaded Documents
- Facilitate to update Merchant Accounts
- Facilitate to retrieve actual Transaction and Merchant details from Bank Portals.
- Profile Management.
- Provide necessary reports.
- Sufficient client support with all necessary data and information.

Provide facility to create, delete and change account type for user accounts

2.5. NON-FUNCTIONAL REQUIREMENTS

These are constraints on the services or functions offered by the system. Non-Functional requirements are more critical than functional requirements and Relate to the system as a whole rather than to individual system features. If these are not met, the system is useless.

2.5.1. Security

Security as a condition is the degree of resistance to, or protection from, harm. Establishing or maintaining a sufficient degree of security is the aim of the work, structures, and processes [1].

2.5.2. Performance

Computer performance is characterized by the amount of useful work accomplished by a computer system compared to the time and resources used [2].

2.5.3. Reliability

The ability of a system or component to perform its required functions under stated conditions for a specified period of time [3].

2.5.4. Usability

Usability is the ease of use and learns ability of the system. The system should be usable for the day to day to business work of the Reservations. Software Solution should ease the business process.

2.5.5. Maintainability

Characteristic of design and installation which determines the probability that a failed equipment, machine, or system can be restored to its normal operable state within a given time frame, using the prescribed practices and procedures. Its two main components are serviceability (ease of conducting scheduled inspections and servicing) and reparability (ease of restoring service after a failure) [4].

2.5.6. Extensibility

The system should be designed and developed in such a way that it can be extendable for future requirements. The design should always leave the space for potential requirements and it should be the base of the future developments.

2.6. PROCESS MODEL

Rational Unified Process

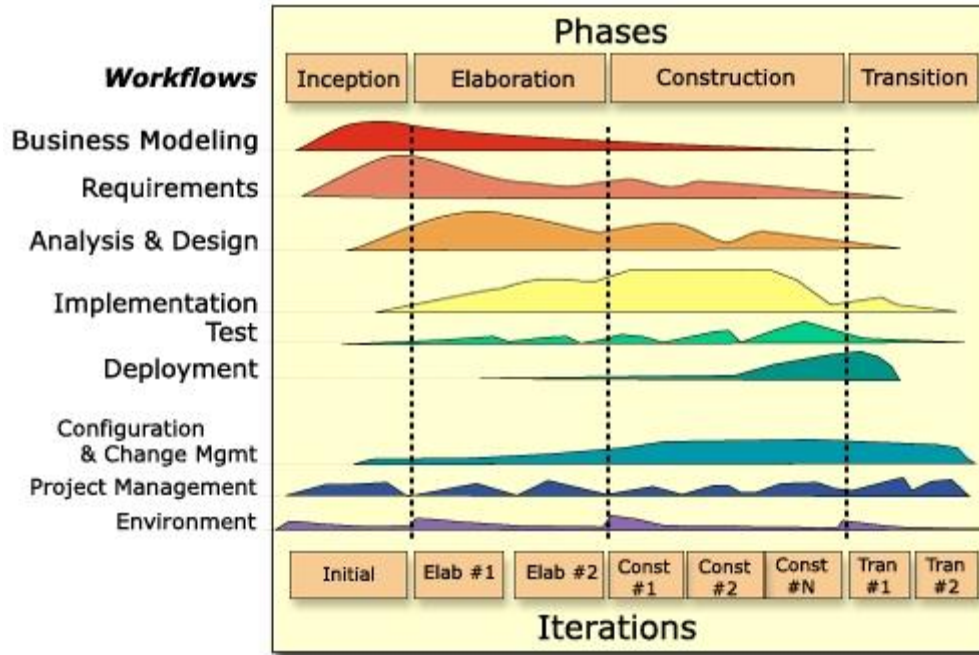


Figure 2.5: Rational Unified Process Model

"Rational Unified Process" [5] (RUP) is a software development process from Rational, a division of IBM. It divides the development process into four distinct phases that each involves business modeling, analysis and design, implementation, testing, and deployment.

The four phases are:

2.6.1. Inception

The idea for the project is stated. The development team determines if the project is worth pursuing and what resources will be needed.

2.6.2. Elaboration

The project's architecture and required resources are further evaluated. Developers consider possible applications of the software and costs associated with the development.

2.6.3. Construction

The project is developed and completed. The software is designed, written, and tested.

2.6.4. Transition

The software is released to the public. Final adjustments or updates are made based on feedback from end users.

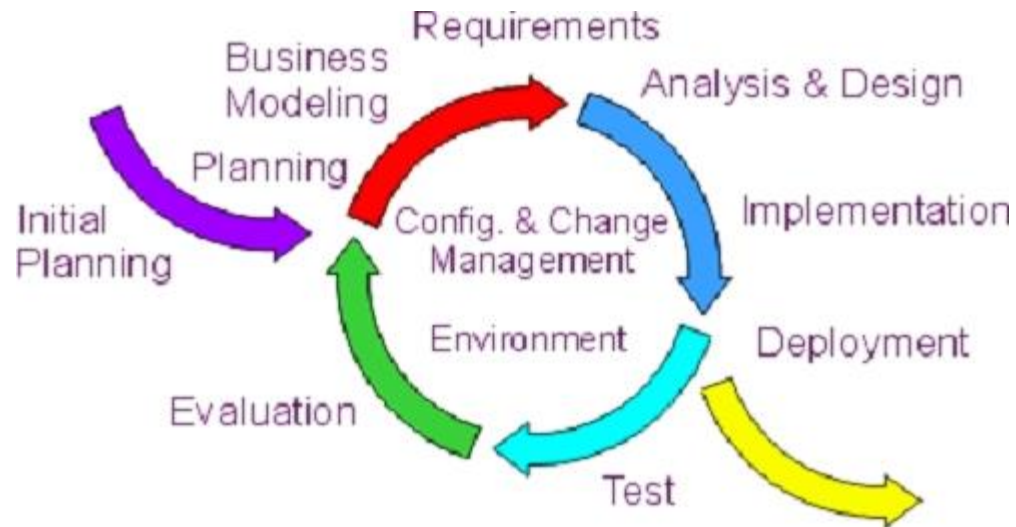


Figure 2.6: Rational Unified Process Model

The RUP development methodology provides a structured way for companies to envision create software programs. Since it provides a specific plan for each step of the development process, it helps prevent resources from being wasted and reduces unexpected development costs [5].

CHAPTER 3 - DESIGN

System designing is comes under the Elaboration phase. Software design is a process of problem-solving and planning for a software solution. After the purpose and specifications of software are determined, software developers will design or employ designers to develop a plan for a solution [6].

3.1. ALTERNATIVE SOLUTIONS

The alternate solutions for the PAYable KPI Indicator System are; developing a standalone system for the KPI Indicator System or developing a web based PAYable KPI Indicator System'

3.1.1. Standalone System

Standalone system can be addressed all the Existing system activities. But Standalone system has many disadvantages.

- Standalone solution which will run only on inside the Smart Technology
- There is no proper way to manage customer or get customer satisfaction.
- System available only permanent customers only.

3.1.2. Web Based System

Web based System can be addressed all the Existing system activities and get more benefits than Standalone system.

- Available for vast number of customers
- KPI Indicator
- System can access any place around the world.

3.2. SYSTEM DESIGN PROCESS

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. The main purpose of the System Design Chapter is to discuss the architecture, components, modules and interface design for the PAYable KPI Indicator System in a clear and concise form. The Unified Modelling Language (UML) is a standardized visual specification language for Object modelling.

3.3. ARCHITECTURE OF THE SYSTEM

The architecture of the PAYable KPI Indicator System is based on the Three-Tier. This three-tier architecture mainly consists of three layers namely,

- Presentation Tier
- Business Tier
- Data Access Tier

The Presentation Tier converts and displays information into a human legible form. This tier displays information related to services such as browsing the website, customer Registration, etc. It communicates with the other tiers by outputting the results to the browser/client tier and all the other tiers. The Business Logic tier is mainly responsible for information exchange between the user interface and the database of the System. The final layer of the three tiered architecture is the Data Access tier, which mainly consists of the Database servers. The information related to the PAYable KPI Indicator System is stored and retrieved from here [7].

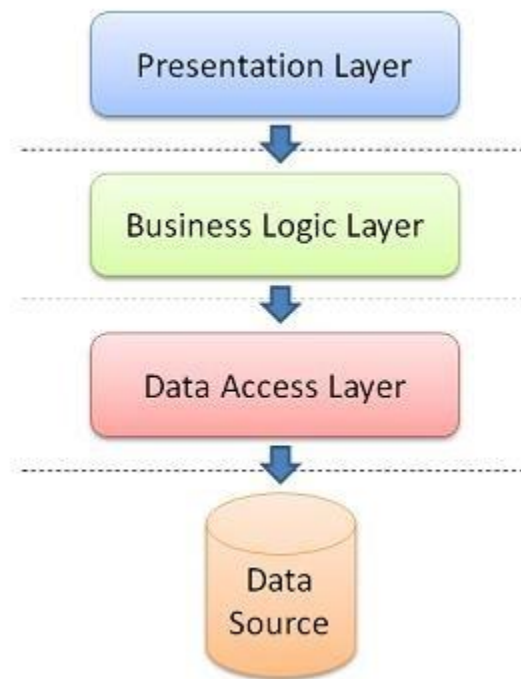


Figure 3.1: Three-tier architecture

3.4. USE CASE DIAGRAM

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well [8].

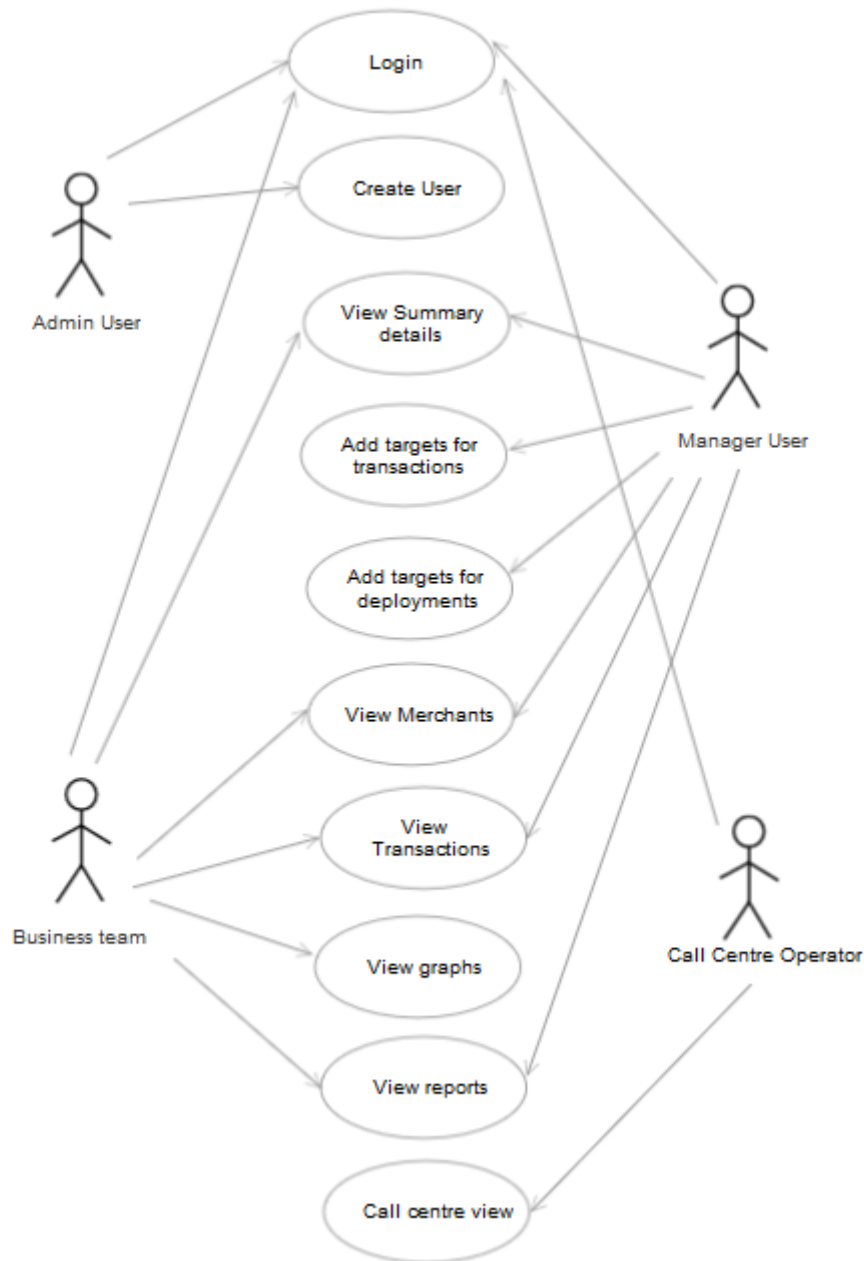


Figure 3.2: Use Case Diagram

3.5. USE CASE NARRATIVES

The following are the use cases and their narratives (Table 3.1 and 3.2) for the refined Use Case (Figure 3.2) drawn during the design phrase.

Use Case Name :	Login
Actors :	Administrator, Manager, Accountant, Client, Team Leader, IPT
Overview :	
This use case describes the scenario where the user logs into the System, with the username and password he has provided after registering the system.	
Precondition :	
Be a Manager, Administrator, Accountant, Client, Team Leader or IPT	
Input :	
Enter the User Name and Pass Word	
Output :	
The application then verifies the authenticity of the Username and Password that the user has provided and allows the user to view the information available on the system, if the username and password are valid.	

Table 3.1: Use Case Narrative (Login)

Use Case Name :	Logout
Actors :	Team Leader, Manager
Overview :	
This use case describes the scenario where the user logs into the system and logs out after the work is done.	
Precondition :	
Login as a Manager, Administrator, Accountant, Client, Team Leader or IPT	
Input :	
Click on logout button	
Output :	
Clear the session and redirect to the home page	

Table 3.2: Use Case Narrative (Logout)

3.6. CLASS DIAGRAM

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

The class diagram is the main building block of object oriented modelling. It is used both for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modelling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed [9].

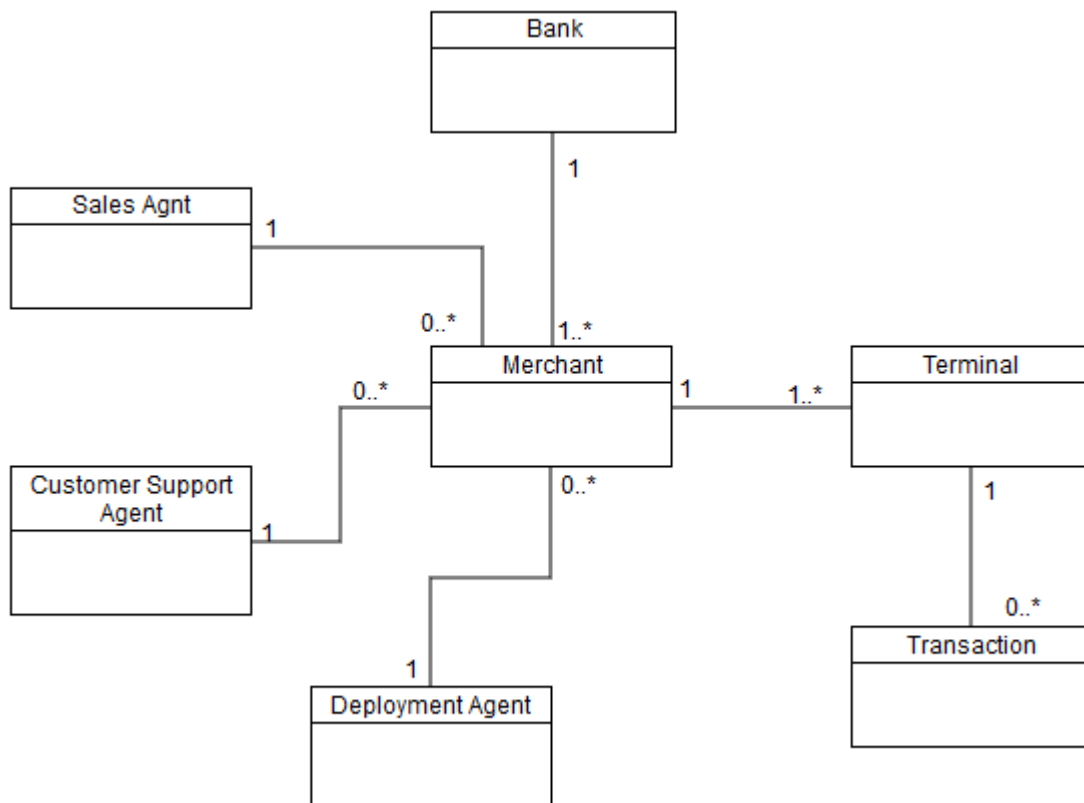


Figure 3.3: Class Diagram

CHAPTER 4 - IMPLEMENTATION

4.1. INTRODUCTION

Implementation is the process of converting the system specification into an executable system. Design and implementation processes transform the specification (as explained in the analysis and design chapters) to an executable program, which are, most of the time interleaved.

A familiarized language and appropriate tools were chosen in the process of development and coding. The codes were written and arranged in a readable and understandable format, along with comments; Hoping to produce a software that will be maintainable in the future.

4.2. IMPLEMENTATION ENVIRONMENT

Some important aspects were taken into consideration when finalizing the implementation environment. When selecting the development software, most of them were free and open source which won't cause much trouble when getting the copyrights of the system. Some of these technologies are targeted at a specific application domain (e.g., Web-site design and implementation); others focus on a technology domain (e.g.: object-oriented systems). The following components were used in the implementation environment.

Hardware	Software
3.0 GHz Intel Processor	Microsoft Windows 7 Professional
4 GB RAM	WAMP Version : 2.4
500 GB Hard	-Apache Version : 2.4.4 -PHP Version : 5.4.16 -MySQL Version : 5.6.12

Table 4.1: Implementation Environment

Although the system was developed under windows 8 and XP operating systems, the IPMS is fully compatible with windows XP as well as with windows Vista.

Also the system was tested under the following resolutions:

- 1366 x 768 (pixels)
- 1280 x 768 (pixels)
- 1024 x 768 (pixels)

4.3. SYSTEM DEVELOPED TOOLS AND TECHNOLOGIES

When developing the system, the following tools and technologies were used and the following bullet points briefly discussed them:

- Adobe Dreamweaver CS5 used for the coding of the system. It contains more supportive background when doing implementation like popup code hints.
- Adobe Photoshop CS5 used for Photo retouching, interface designing had done by using this software.
- PHP (Hypertext Pre Processor) was the main development language used to develop the main system and its logics.
- WAMP Server – As a server and handle database.
- HTML was used to build the base Interfaces of the system.
- CSS was used to make the plain HTML interfaces more attractive and user friendly, which also decided the look and feel of the system.
- JavaScript was used to code all the client-side validation.
- AJAX which is based on JavaScript was used to get data from the server without refreshing it repetitively.
- JQuery which is also based on JavaScript was used to implement the pre-coded time picker module, transition effect for the login and simple password meter.
- Microsoft PDF Plug-in (SaveAsPDFandXPS) – for PDF creation.
- Notepad ++ (Simple Text Editor).

4.4. MAJOR CODE SEGMENTS

The main code modules developed in the system have been mentioned below by briefly describing their functionality. Code modules consist with comments to identify the specific reason of a particular code line.

4.4.1. Database configure page

Before we can access data in the MySQL database, we need to be able to connect to the server.

PDO Fetch Style

By default, database results will be returned as instances of the PHP stdClass object; however, you may desire to retrieve records in an array format for simplicity. Here you can tweak the fetch style.

```
<?php
return ['fetch' => PDO::FETCH_OBJ,
```

Default Database Connection Name

Here you may specify which of the database connections below you wish to use as your default connection for all database work. Of course you may use many connections at once using the Database library.

```
'default' => env('DB_CONNECTION', 'mysql'),
```

Database Connections

Here are each of the database connections setup for your application. Of course, examples of configuring each database platform that is supported by Laravel is shown below to make development simple. All database work in Laravel is done through the PHP PDO facilities so make sure you have the driver for your particular database of choice installed on your machine before you begin development.

```
'connections' => [

    'sqlite' => [
        'driver' => 'sqlite',
        'database' => env('DB_DATABASE',
database_path('database.sqlite')),
        'prefix' => '',
    ],

    'mysql' => [
        'driver' => 'mysql',
        'host' => env('DB_HOST', 'localhost'),
        'port' => env('DB_PORT', '3306'),
        'database' => env('DB_DATABASE', 'forge'),
        'username' => env('DB_USERNAME', 'forge'),
        'password' => env('DB_PASSWORD', ''),
        'charset' => 'utf8',
        'collation' => 'utf8_unicode_ci',
        'prefix' => '',
        'strict' => true,
        'engine' => 'InnoDB',
    ],

    'pgsql' => [
        'driver' => 'pgsql',
        'host' => env('DB_HOST', 'localhost'),
        'port' => env('DB_PORT', '5432'),
        'database' => env('DB_DATABASE', 'forge'),
        'username' => env('DB_USERNAME', 'forge'),
        'password' => env('DB_PASSWORD', ''),
        'charset' => 'utf8',
        'prefix' => '',
        'schema' => 'public',
        'sslmode' => 'prefer',
    ],

],
```

Redis Databases

Redis is an open source, fast, and advanced key-value store that also provides a richer set of commands than a typical key-value systems such as APC or Memcached. Laravel makes it easy to dig right in.

```
'redis' => [

    'cluster' => false,

    'default' => [
        'host' => env('REDIS_HOST', 'localhost'),
        'password' => env('REDIS_PASSWORD', null),
        'port' => env('REDIS_PORT', 6379),
        'database' => 0,
    ],

],

];
```

4.4.2. Session handling page

Code segment given below connects all the required files and start the session and also catches the user input for the login process.

```
<?php

return [

    /*
    | Default Session Driver
    */

    'driver' => env('SESSION_DRIVER', 'file'),

    /*
    | Session Lifetime
    */

    'lifetime' => 120,

    'expire_on_close' => false,

    /*
    | Session Encryption
    */

    'encrypt' => false,
```



```
/*
    | Session File Location
*/

'files' => storage_path('framework/sessions'),

/*
    | Session Database Connection
*/

'connection' => null,

/*
    | Session Database Table
*/

'table' => 'sessions',

/*
    | Session Cache Store
*/

'store' => null,

/*
    | Session Sweeping Lottery
*/

'lottery' => [2, 100],

/*
    | Session Cookie Name
*/

'cookie' => 'laravel_session',

/*
    | Session Cookie Path
*/

'path' => '/',

/*
    | Session Cookie Domain
*/

'domain' => env('SESSION_DOMAIN', null),

/*
    | HTTPS Only Cookies
*/
```

```
'secure' => env('SESSION_SECURE_COOKIE', false),

/*
| HTTP Access Only
*/

'http_only' => true,
];
```

4.4.3. Logout

The function below describes about the logout page. After the user had successfully logged in to the system this function helps the user to terminate his logged session from the system.

```
<?php
    // Initalize session
    session_start();
    //Unset the session and redirect the login page.
    unset($_SESSION['Email']);
    header("Location: login.php" );
?>
```

Other codes can be found in APPENDIX - F.

4.5. RE-USABLE COMPONENTS

Some codes were found by referring internet and used in this project. Extracted codes were customized and well tested. Following codes are extracted from the websites.

opendir():- This code segment used to search folder path in local server/ machine.

readdir() :- This code segment used to read directory names in local server/ machine.

jQuaries and validation codes are written by referring internet.

CHAPTER 5 - EVALUATION

5.1. INTRODUCTION

Evaluating a system is the process of checking that a software system meets specifications and that it fulfils its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Software evaluation makes it possible to determine if the products would be helpful to the client or if some other combination of software products would serve to better advantage.

Two approaches can be followed to ensure software quality. One is focused on a direct specification and evaluation of the quality of software product, while the other is focused on assuring high quality of the process by which the product is developed.

The primary purpose of evaluation, in addition to gaining insight into prior or existing initiatives, is to enable reflection and assist in the identification of future change.

5.2. SOFTWARE TESTING

Testing can be defined in simple words as “Performing Verification and Validation of the Software Product” for its correctness and accuracy of working.

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs [10].

Purpose of the software testing is to determine if the Software is designed as per the requirement of the target stakeholders.

5.3. SOFTWARE TESTING METHODS

There are different methods that can be used for software testing.

5.3.1. Black box Testing

Black box testing is the Software testing method which is used to test the software without knowing the internal structure of code or program.

Most likely this testing method is what most of tester actual perform and used the majority in the practical life.

The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon [11].

5.3.2. White box testing

White box testing is a testing technique that examines the program structure and derives test data from the program logic/code.

It focuses primarily on strengthening security, the flow of inputs and outputs through the application.

The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

5.4. TYPES OF TESTING

5.4.1. Unit Testing

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use [12].

5.4.2. Integration Testing

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit.

In a realistic scenario, many units are combined into components, which are in turn aggregated into even larger parts of the program. The idea is to test combinations of pieces and eventually expand the process to test your modules with those of other groups.

Eventually all the modules making up a process are tested together. Beyond that, if the program is composed of more than one process, they should be tested in pairs rather than all at once [13].

5.4.3. System Testing

In system testing the behavior of whole system/product is tested as defined by the scope of the development project or product.

It may include tests based on risks and/or requirement specifications, business process, use cases, or other high level descriptions of system behavior, interactions with the operating systems, and system resources.

System testing is most often the final test to verify that the system to be delivered meets the specification and its purpose [14].

5.4.4. Functional Testing

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional testing usually describes what the system does [15].

5.4.5. Non-Functional Testing

Non-functional testing is the testing of a software application or system for its non-functional requirements: the way a system operates, rather than specific behaviors of that system. This is contrast to functional testing, which tests against functional requirements that describe the functions of a system and its components. For example, Security and Reliability [16].

5.4.6. Regression Testing

Regression testing is a type of software testing that seeks to uncover new software bugs, or regressions, in existing functional and non-functional areas of a system after changes such as enhancements, patches or configuration changes, have been made to them.

The purpose of regression testing is to ensure that changes such as those mentioned above have not introduced new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software [17].

5.4.7. Usability Testing

The system will be tested with the real users to see if the system satisfies the intended purpose, easy to understand and simple.

Usability testing is the best way to understand how real users experience your website or application.

Unlike interviews or focus groups that attempt to get users to accurately self-report their own behavior or preferences, a well-designed user test measures actual performance on mission-critical tasks.

5.5. SYSTEM TEST PLAN

Test plan is intended to provide solid guidance for the testing, test deliverables etc. It helps to identify, plan, execute and report the testing task of the PAYable KPI Indicator System project. This document allows other stakeholders of the project to understand the software testing approach.

The test plan document is a living document and will be changed based on the project dynamics. But the scope of the testing is fixed and will not be changed.

The test plan will provide baseline for all the software testing activities planned within the scope of the project. Following Table 5.1 shows test plan.

Module Name	Function name	Test Priority
Web Site	Load web site	High
Login	Login into the system	High
	Re-set password in first login attempt	High
	Remember password	High
	Update password	High
	Display validation error messages	Medium
User Management	Create User	High
	Create Roles	High
	Create Permission	High

	Active, deactivate users	High
Summary	Yearly target achievement	High
	Merchant Count	High
	Total Turnover	High
	Total Transaction Count	High
	Active merchants	High
	Ready to deploy	Medium
	Current month Actual	Medium
	Current month Target	Medium
	Daily Reports	High
	Monthly Reports	High
	Average Transaction Value per merchant	High
	Yearly graphs	High
Transactions	View transaction details	Medium
	View receipt	Medium
Merchants	View Merchant Details	Medium
	View Transaction Counts	High
	View Transaction Amounts	High
Total Count	View Total	High
	View total for Test Accounts	High
	View actual total	High
Call Center View	View settle amount	High
	View Settle Count	High

	View Void Amount	Medium
	View Void Count	Medium
Target Setting	Set Monthly Target for Transactions	High
	Set Monthly Target for Deployments	High

Table 5.1: High Level Test Plan

5.6. SYSTEM TEST CASES

A test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

A properly planned test case should have the ability to verify the relevant system component functionality. Therefore, to verify all the system functions there should be properly planned test cases for each and every function.

Following Tables (Table 5.2, and Table 5.3) shows test cases. Please refer Appendix E for test results.

5.6.1. Test Cases for Login

Test Case ID	Test Case
1	Verify user can Login for PAYable-Reports successfully.
2	Verify user enter valid Email name and invalid password
3	Verify user enter invalid email and valid password.
4	Verify sign in attempts with leading spaces for Email
5	Verify user attempts to sign in with leading space for password
6	Verify user trying to sign in with trailing space for Email
7	Verify user trying to login with blank Credentials
8	Verify user trying to Login with blank password.

9	Verify user trying to login with blank email address
10	User trying to sign in to system with trailing spaces for password.
11	Verify functions of the system differ according to the sign in user

Table 5.2: Test Cases - Login

5.6.2. Test Cases for User Management

Test Case ID	Test Case
12	Verify authorized user can access 'user management' tab
13	Verify 'Users' page loads with Existing merchants
14	Verify user can navigate 'Users, Roles and Permissions' tabs
15	Verify records paginations are applied properly.
16	Verify 'user management' page load with 'Users' tab
17	Verify Click on 'Create User'
18	Verify Fill all mandatory fields and Save details.
19	Verify mandatory fields should be indicated by asterisk (*) symbol.
20	Verify cannot save details without filling all mandatory fields
21	Verify select 'Roles' using drop down list.
22	Verify User can 'Active' users
23	Verify User can 'create users' without activation
24	Verify enter invalid Confirm password
25	Verify User can 'Deactivated' users
26	Verify User can edit existing users
27	Verify user cannot enter same 'Email' more than once
28	Verify enter valid Email.

29	Verify User can Delete existing users
30	Verify deactivate functionality for users on page should ask for confirmation
31	Verify confirmation message, before Deleting existing users
32	Verify user can view 'user' details
33	Verify Click on 'Create New Role'
34	Verify Fill all mandatory fields and Save details.
35	Verify mandatory fields should be indicated by asterisk (*) symbol.
36	Verify cannot save details without filling all mandatory fields
37	Verify select 'Permission' using check boxes
38	Verify User can edit existing Roles
39	Verify user cannot enter same 'Name' more than once
40	Verify User can Delete existing user Roles
41	Verify confirmation message, before Deleting existing user role
42	Verify user can view 'user role' details
43	Verify Click on 'Create New Permission'
44	Verify Fill all mandatory fields and Save details.
45	Verify mandatory fields should be indicated by asterisk (*) symbol.
46	Verify cannot save details without filling all mandatory fields
47	Verify user can delete assigned Permissions
48	Verify User can edit existing Permissions
49	Verify user cannot enter same 'Name' more than once
50	Verify User can Delete existing user Permissions
51	Verify confirmation message, before Deleting existing user Permissions

52	Verify user can view Permissions
----	----------------------------------

Table 5.3: Test Cases - User management

5.6.3. Test Cases for Call Centre View

Test Case ID	Test Case
53	Verify authorized user can access 'call Centre View' tab
54	Verify Call centre view fields.
55	Verify user can be filter call centre view data

Table 5.4: Test Cases - Call Center View

5.6.4. Test Cases for Merchants

Test Case ID	Test Case
56	Verify authorized user can access 'Merchant' tab
57	Verify records paginations are applied properly.
58	Verify Merchant tab fields.
59	Verify merchant tab data table.
60	Verify user can be filter merchant data
61	Verify user can be filter transaction data

Table 5.5: Test Cases - Merchants

5.6.5. Test Cases for Target Setting

Test Case ID	Test Case
62	Verify authorized user can access 'Target setting' tab
63	Verify user can select option from bank and year in 'Monthly target' sub tab
64	Verify user can't enter the value in last month or year target value 'Monthly target' sub tab
65	Verify target value 'Monthly target' sub tab
66	Verify Actual value 'Monthly target' sub tab
67	Verify user can add target value 'Monthly target' sub tab
68	Verify user can access select option in 'Monthly deployment' tab
69	Verify user can't enter the value in last month or year target value 'Monthly Deployment' sub tab
70	Verify target value 'Monthly Deployment' sub tab
71	Verify Actual value 'Monthly Deployment' sub tab
72	Verify user can add target value 'Monthly Deployment' sub tab

Table 5.6: Test Cases - Target Settings

5.6.6. Test Cases for Total Count

Test Case ID	Test Case
73	Verify authorized user can access 'Total Count' tab
74	Verify user can select option from bank dropdown
75	Verify the 'Account Total' data base on bank name
76	Verify the 'Test Accounts/Other' data base on bank name
77	Verify the 'Actual Count' data base on bank name

Table 5.7: Test Cases - Total Count

5.6.7. Test Cases for Summary

Test Case ID	Test Case
78	Verify Summary page loads properly
79	Verify user can select bank
80	Verify filtered data changed according to the bank
81	Verify Summary tab loads properly
82	Verify Daily Reports/Year loads properly
83	Verify user can be select option field in 'Daily Reports/Year tab'
84	Verify Daily Users data
85	Verify target data
86	Verify authorized user can access 'Monthly Report' sub tab
87	Verify user can select option from dropdown in 'Monthly Report' sub tab
88	Verify 'Monthly Report' for every months base on selected year
89	Verify authorized user can access 'Average Transactions Value Per Merchant' sub tab
90	Verify user can select option from dropdown in 'Average Transactions Value Per Merchant' sub tab
91	Verify 'Average Transaction Value' for every months base on selected year
92	Verify authorized user can access 'Yearly Graphs' tab
93	Verify user can access details in 'Yearly Graphs' tab
94	Verify user can select option from dropdown
95	Verify 'Transactions' data base on selected year
96	Verify 'Active Users' data base on selected year

Table 5.8: Test Cases - Summary

5.7. USER ACCEPTANCE TESTING

User acceptance is a type of testing performed by the Client to certify the system with respect to the requirements that was agreed upon. This testing happens in the final phase of testing before moving the software application to Market or Production environment.

User acceptance test cases shall be based on the scenarios given by the customer for accepting the system.

USER EVALUATION FORM
For
Key Performance Indicator System

Name of User: D. Namal Jayasinghe

Role of User : Software Engineer

Evaluating Item	Very Good	Good	Average	Poor	Very Poor
Overall reaction	✓				
Character readability		✓			
Color scheme	✓				
System navigation	✓				
Ease of usage	✓				
Functionalities	✓				
Interfaces		✓			
Ease of learning	✓				
Response time	✓				
Comments: <u>This product is very good & helpful to us. Reporting System is quite helpful for to get decisions. please do the modification in the future.</u>					

Date: 08.11.2017 Signature: all

Figure 5.1: User Evaluation Form

CHAPTER 6 - CONCLUSION

6.1. INTRODUCTION

PAYable is a free mobile app paired with a free Bluetooth cardreader that enables merchants to accept credit and debit card payments from their customers. A highly reliable, durable and compact card reader, PAYable can be paired with any Android or iOS smart phone or tablet to effectively convert it into a secure, flexible and simple to use mobile Point-of-Sales (POS) solution. The revolutionary PAYable cashless mobile payments platform has gained the trust and support of a diverse cross-section of Micro, Small and Medium Enterprises (MSMEs) and entrepreneurs and has become the leading choice for major banks. They need to enhance their business with new technology.

Smart Technology provides business solutions for PAYable as their Partner Company. This project is a web based KPI system for a cashless mobile payment solutions company named as PAYable. The project objective is to deliver the PAYable Key Performance Indicator System for Smart Technology.

Implementation of PAYable KPI System was helping to increase the productivity and direct guidance through the total process and improving the business process to be more effective and efficient.

By comparing the user feedback, test results, system functionality with the existing system; it was identified as a system which can satisfy the client requirements up to a satisfactory level.

6.2. FUTURE IMPROVEMENTS

Developing a commercial level system is a huge task. Some suggestions for improvements in the future are as follows:

Providing Online Payment Facility by obtaining SSL

By implementing this suggestion, the internal user as well as the external user can do their transactions via the system using a payment gateway.

Add customized profile facility

By implementing this suggestion, administrator can customize the profile as he wants and can set privileges according to the user type.

Add more reports

By implementing this suggestion, system can add more reports according to the client's current need. It can enhance feature by adding bar charts and pie charts.

6.3. LESSONS LEARNT

This developed system does not merely fulfil the requirement of the final year of the Degree program; however it assists me to practically apply the knowledge learnt throughout the past three years. When assigning the project proposal, I did not have much of an idea on how to carry out the project. When progressing through step by step according to the guideline provided by the university, I gained a valuable knowledge on how to do a successful professional system development project. By doing the development process according to a schedule, I learnt how to do my day-to-day activities by managing time efficiently.

The implementation phase was the toughest and most interesting phase of the project, as it allowed me to try out practically the academic knowledge that I have gained on programming languages such as PHP, Laravel, Java Scripts, jQuery, CSS and many more development tools and techniques. Further I learnt configuring E-mail server according to the system. Writing the dissertation was another interesting task of the project. It provided me with lessons on how to write a report in a professional manner. It helped me to develop my skills on writing and designing technical reports.

REFERENCES

- [1] Security, Wikipedia, the free encyclopaedia.
[Online] Available: <http://en.wikipedia.org/wiki/Security>
[Accessed: 16 Feb, 2018].
- [2] Performance, Wikipedia, the free encyclopaedia.
[Online] Available: http://en.wikipedia.org/wiki/Computer_performance
[Accessed: 16 Feb, 2018].
- [3] Reliability, Wikipedia, the free encyclopaedia.
[Online] Available: http://en.wikipedia.org/wiki/Reliability_engineering
[Accessed: 16 Feb, 2018].
- [4] Maintainability, Online business dictionary.
[Online] Available: <http://www.businessdictionary.com/definition/maintainability.html>
[Accessed: 16 Feb, 2018].
- [5] RUP, RUP Fundamentals Presentation.
[Online] Available: https://era.nih.gov/docs/rup_fundamentals.htm
[Accessed: 16 Feb, 2018].
- [6] Design, Wikipedia, the free encyclopaedia.
[Online] Available: http://en.wikipedia.org/wiki/Software_design
[Accessed: 16 Feb, 2018].
- [7] Three-tier architecture, stack overflow
[Online] Available: <https://stackoverflow.com/questions/2011698/3-tier-architecture-in-need-of-an-example>
[Accessed: 16 Feb, 2018].
- [8] Use Case Diagram, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Use_case_diagram
[Accessed: 16 Feb, 2018].
- [9] Class Diagram, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Use_case_diagram
[Accessed: 16 Feb, 2018].
- [10] Software testing, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Software_testing
[Accessed: 16 Feb, 2018].

- [11] Black-box testing, Tutorials point.
[Online] Available: http://www.tutorialspoint.com/software_testing/software_testing_methods.htm
[Accessed: 16 Feb, 2018].
- [12] Unit Testing, Software testing stuff
[Online] Available: <http://www.softwaretestingstuff.com/2010/09/unit-testing-best-practices-techniques.html>
[Accessed: 16 Feb, 2018].
- [13] Integration Testing, Techno func.
[Online] Available: <http://www.technofunc.com/index.php/erp/178-what-is-integration-testing>
[Accessed: 16 Feb, 2018].
- [14] System Testing, ISTQB Exam Certification
[Online] Available: <http://istqbexamcertification.com/what-is-system-testing/>
[Accessed: 16 Feb, 2018].
- [15] Functional Testing, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Functional_testing
[Accessed: 16 Feb, 2018].
- [16] Non-functional testing, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Non-functional_testing
[Accessed: 16 Feb, 2018].
- [17] Regression testing, Wikipedia, the free encyclopaedia.
[Online] Available: https://en.wikipedia.org/wiki/Regression_testing
[Accessed: 16 Feb, 2018].

APPENDIX A - SYSTEM DOCUMENTATION

This documentation consists of a set of steps to show, how to install this PAYable KPI Indicator System. These steps explain about the hardware and software environment which needs to be installed. When installing the system, this documentation can be followed by the interested parties.

In order to install the system, the Device chosen for installation should meet the following prerequisites of Hardware and Software.

HARDWARE REQUIREMENTS

Hardware	Recommended Minimum Requirements
Processor	3.0 GHz Intel Processor
Memory	4 GB RAM
Hard disk space	500 GB Hard
Display	1024*768 Resolution Monitor or resolutions above High.
Printer	Inkjet Printer or LaserJet Printer
Internet	ADSL Connection (Minimum Speed 512Kbps)

Table A.1: Hardware Requirements

SOFTWARE REQUIREMENTS

Software	Recommended Minimum Requirements
Operating System	Windows 7, Windows 8, Windows 10 O/S
WAMP	Version : 2.4 -Apache Version : 2.4.4 -PHP Version : 5.4.16 -MySQL Version : 5.6.12
Web Browser	Firefox (Version 39.0) Web browser.
Code Editor	NetBeans IDE 8.2
Framework	Laravel 5.4

Table A.2: Software Requirements

Installing WAMP

Download and install WAMP for Windows (refer Table A.2 for the Minimum Version) from <http://www.wampserver.com>. Give installation path to C:\wamp of the computer.

Installing Web Browsers

Install Browsers (refer Table A.2 for the Version and Recommended Browsers).

Files Extraction

Open the CD and copy the PAYable KPI folder and paste it to the directory path "C:\wamp\www"

Database Installation

Open the web browser and type the URL <http://localhost:8080/phpmyadmin/> and enter Username and Password (if you set username and password).

Create empty database by providing name as "payable_reports" and navigate to the "Import" tab and click "choose file" button. Then browse the CD and select the "payble_reports.sql" file by opening Database folder.

Then Press "GO" button located in the bottom of the page.

Launching System

Verify the WAMP is running, go to the “C:\wamp\” and open the control panel and verify whether Apache, PHP and MySQL are running.

Open the installed web browser and type the URL
http://localhost:8080/payable_reports/public/ and press “Enter” button to access the system.

Please refer Appendix-C User Documentation to get the idea about how to operate the system.

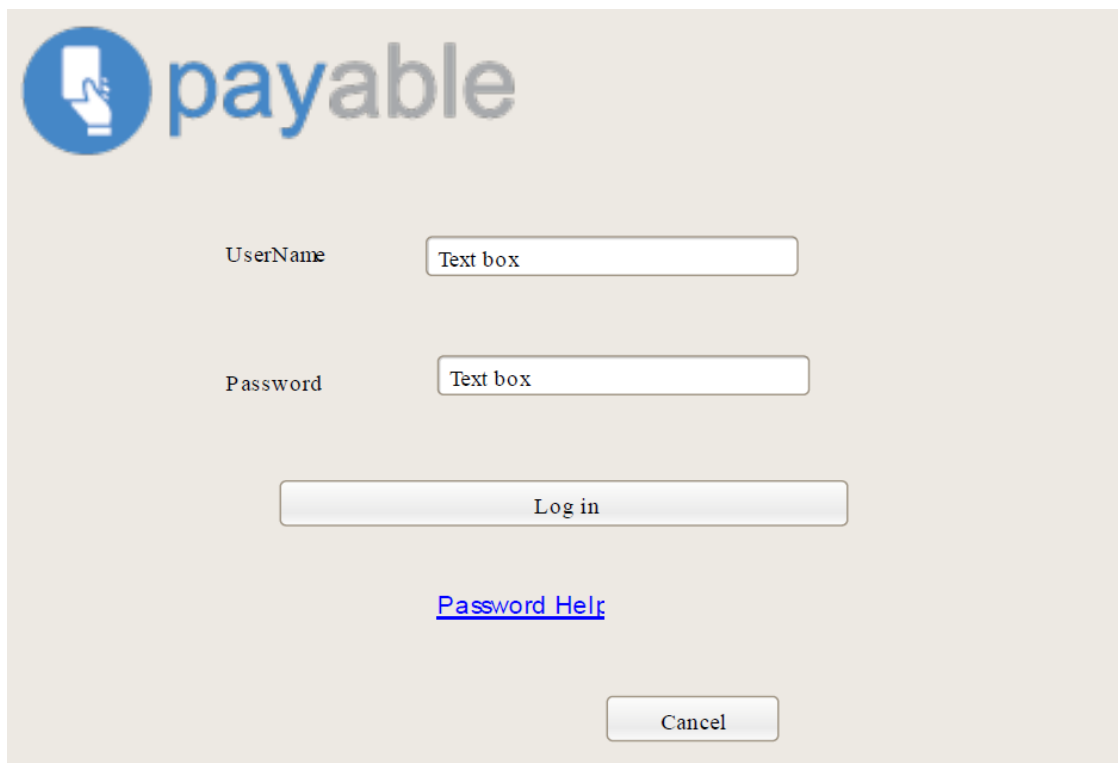
APPENDIX B - DESIGN

DOCUMENTATION

USER INTERFACE DESIGN

Login Interface

Following Figure B.1 shows User Login. Authorized person can log into the system. All users can log into the system using valid credentials.



The screenshot displays a login interface for a system. At the top left, there is a blue circular icon containing a white hand cursor pointing at a document, followed by the word "payable" in a large, blue, sans-serif font. Below this header, the form consists of two rows of labels and text boxes. The first row has the label "UserName" in a small, black, sans-serif font, followed by a white rectangular text box with a thin grey border and the placeholder text "Text box". The second row has the label "Password" in the same font, followed by a similar white rectangular text box with the placeholder text "Text box". Centered below these fields is a wide, light grey button with rounded corners and the text "Log in" in a small, black, sans-serif font. Below the button is a blue, underlined hyperlink that reads "Password Help". At the bottom right of the form area is a smaller, light grey button with rounded corners and the text "Cancel" in a small, black, sans-serif font. The entire interface is set against a light beige background.

Figure B.1: Interface - Login

Summary Page Interface

Figure B.2 shows total achievements of payable app according to bank.

SummaryPage

The screenshot displays the 'SummaryPage' interface. On the left is a vertical navigation menu with links: [Summan](#), [Transaction:](#), [Merchants](#), [Total Count](#), [Call Center Vie](#), and [Monthly Target Seti](#). The main content area is titled 'Total Turnover'. It features a 'Bank' dropdown menu labeled 'Combo Box'. Below this are five tabs: 'Summary', 'DailyReports/Month', 'MonthlyReports/Year', 'Average Transactions Value Per Merchant', and 'Yearly Graphs'. The 'Summary' tab is active, showing 'Yearly Target Achievement %'. It includes a table with columns 'Current Month Targe' and 'Current Month Actua'. Below the table is a link 'Deployments Graph to be shov' and the text 'Yet to be deployed number'. At the bottom, there are five 'Year' dropdown menus and two links: [Transaction:](#) and [Active Users](#).

Figure B.2: Interface – Summary Page

Merchant Details Interface

Figure B.3 shows merchant details according to bank.

Merchant Details

The screenshot displays the 'Merchant Details' interface. On the left is a vertical navigation menu with links: [Summan](#), [Call Center Vie](#), [Transaction:](#), [Merchants](#), [Total Count](#), and [Monthly Target Seti](#). The main content area is titled 'Merchant Details'. It features a 'Bank' dropdown menu labeled 'Combo Box'. Below this is a 'Merchant Name' dropdown menu. The 'Transaction Details' section includes a 'Year' dropdown menu.

Figure B.3: Interface – Merchant Details

Transaction Details Interface

Figure B.4 shows transaction details according to bank.

Transactions

All Transactions

Bank

Year Month Date

Card Type Transaction Type

TXN ID	TXN Date	Amount	Card Type	TXN Type	ID	Card No.	Rece
--------	----------	--------	-----------	----------	----	----------	------

Figure B.4: Interface – Transaction Details

Monthly Target Setup Interface

Following Figure B.5 shows Monthly target form which allows only Admin user. User cannot access past months.

Monthly Target Set Up

Monthly Target Set Up

Year Bank

	Target	Actual
January	<input type="text"/>	
February	<input type="text"/>	
March	<input type="text"/>	
April	<input type="text"/>	
May	<input type="text"/>	
June	<input type="text"/>	
July	<input type="text"/>	
August	<input type="text"/>	
September	<input type="text"/>	
October	<input type="text"/>	
November	<input type="text"/>	
December	<input type="text"/>	

Save Clear

Figure B.5: Interface – Monthly Target Setup

Total Counts Interface

Following Figure B.6 shows total counts and amounts of total, test and actual transaction values.

Total Counts

Total Counts

Bank:

Total	
Total	#REF!
Total Transactions	#REF!
Merchant count	935
Transaction Count	16882
Count of Close Transactions	14993
Count of Void Transactions	1888
Total Void	848,508.69

Test Accounts/ Other	
Total Transactions	661.31
Merchant count	9
Transaction Count	683
Count of Close Transactions	280
Count of Void Transactions	403
Total Void	4,928.49

ACTUAL COUNT	
Total Transactions	#REF!
Merchant count	926
Transaction Count	16199
Count of Close Transactions	14713

Figure B.6: Interface – Total Counts

Reports

Daily Deployments Interface

Following Figure B.7 shows authorized user able to generate daily deployment counts according to deployment agents

TOTAL - January																																	
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	Total	
Bunty	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Kasun	0	0	2	2	3	0	1	0	6	13	2	0	3	0	0	0	1	18	4	4	0	0	1	0	5	7	5	0	0	0	0	3	77
Arshad	0	1	2	4	3	0	0	0	8	13	2	0	0	0	0	2	2	18	0	4	0	0	1	0	0	0	0	0	0	0	0	0	60
Ranga	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Sashika	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Priyankara	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
Nabeel	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ayeshi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Prabu	0	0	0	0	0	0	0	0	2	2	1	0	0	0	0	0	1	0	1	1	1	0	1	0	0	0	2	0	1	0	0	0	13
Anurada	0	0	0	0	0	0	1	0	1	2	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	9
Romesh	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	3	0	0	1	0	0	0	0	1	0	0	0	0	8
	0	1	4	6	6	0	2	0	19	31	5	0	4	0	0	2	4	37	9	10	1	1	3	0	5	7	10	0	1	0	3	168	

Figure B.7: Interface – Daily Deployments

Monthly Deployments Interface

Following Figure B.8 shows authorized user able to generate daily deployment counts according to month.

Deployments-2017												1218
Name	JANUARY 1	FEBRUARY 2	MARCH 3	APRIL 4	MAY 5	JUNE 6	JULY 7	AUGUST 8	SEPTEMBER 9	OCTOBER 10	NOVEMBER 11	DECEMBER 12
Bunty	0	22	0	1	0	0	0	0	0	0	0	0
Kasun	80	93	52	43	41	23	32	55	39	25	0	0
Arshad	60	54	122	40	44	18	28	63	13	0	0	0
Ranga	0	0	0	0	0	0	0	0	0	0	0	0
Sashika	0	4	0	0	0	0	0	1	0	0	0	0
Priyankara	1	24	8	3	1	1	1	10	2	2	0	0
Nabeel	0	12	47	6	9	0	0	0	0	0	0	0
Rohan	0	9	6	9	3	0	2	5	1	2	0	0
Ayeshi	0	41	9	0	3	0	0	0	0	0	0	0
Vincent	0	0	32	0	0	0	0	0	0	0	0	0
Adil	0	0	4	43	70	59	35	75	0	0	0	0
Prabu	13	9	30	12	13	17	3	86	64	36	0	0
Anurada	9	22	14	13	6	19	11	21	13	16	0	0
Romesh	8	8	11	23	12	21	13	22	20	7	0	0
Rizmiya	0	0	0	3	16	30	24	37	25	37	0	0

Figure B.8: Interface – Monthly Deployments

Graphs

Relevant graphs should be loaded, when user select relevant functions.

Monthly Target Achievements

Following Figure B.9 shows monthly target achievement graph.

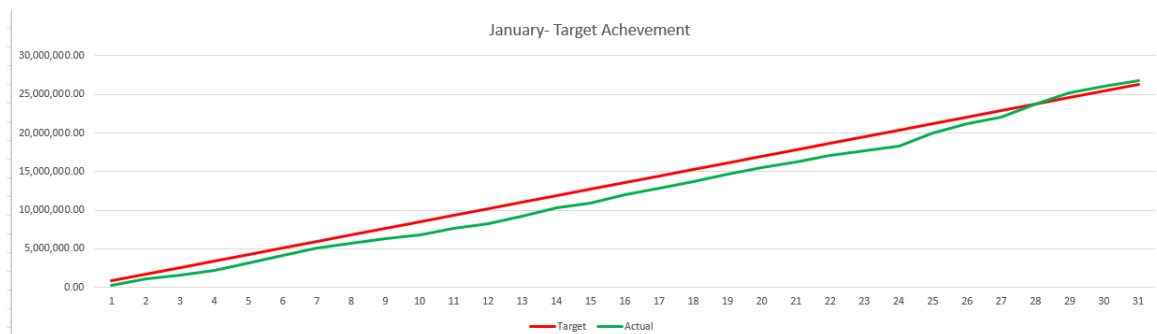


Figure B.9: Interface – Monthly Target Achievements

Average Transaction Values

Following Figure B.10 shows average transaction value graph.

- Daily average transaction value.
- Daily average transaction value for active merchants.
- Daily average transaction value for all merchants

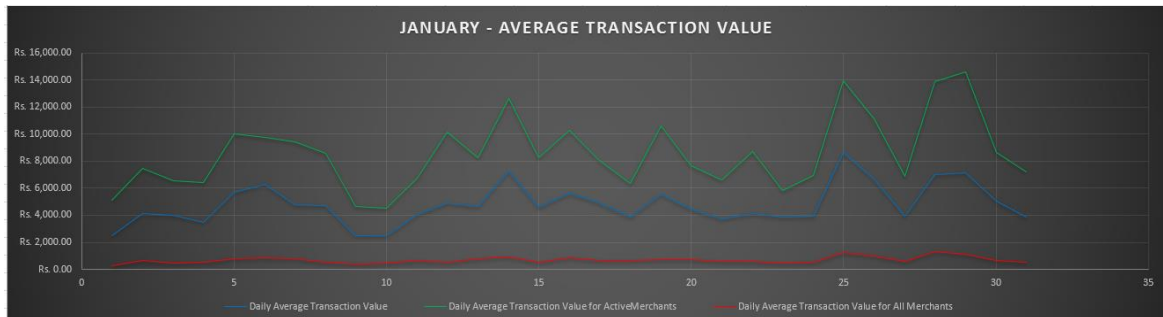


Figure B.10: Interface – Average Transaction Values

Growth Chart

Following Figure B.11 shows Growth Chart.



Figure B.11: Interface – Growth Chart

Call Center View

Following Figure B.12 shows Call centre view. Call centre agents can access call centre view and they can view total counts and amounts of settled and void transactions.

Call Center View

The screenshot displays the 'Call Center Dashboard' interface. On the left is a vertical sidebar with navigation links: [Summan](#), [Call Center View](#), [Transaction](#), [Merchant](#), [Total Count](#), and [Monthly Target Set](#). The main content area is titled 'Call Center Dashboard' and contains a form with the following elements: a 'Bank' label next to a 'Combo Box' dropdown; a 'Merchant Name' label next to a dropdown menu; and two 'Text box' input fields. Below the form is a table with four columns: 'Settled Amour', 'Settled Coun', 'Void Coun', and 'Void Amoun'. The table body is currently empty.

Figure B.12: Interface – Call Center View

DATABASE DESIGNING FOR THE SYSTEM

Valuable data can be kept in a proper order without losing them by a better database design. The above mentioned goal can be achieved by the database normalization method. Reduction of data redundancy and keeping consistency of the database is helped by it.

Normalization consists with several normal form stages with different goals. They are First Normal Form (1NF), Second Normal Form (2NF) and Third Normal Form (3NF). Database for the system has normalized up to the Third Normal Form and the diagram is depicted under the figure B.13.

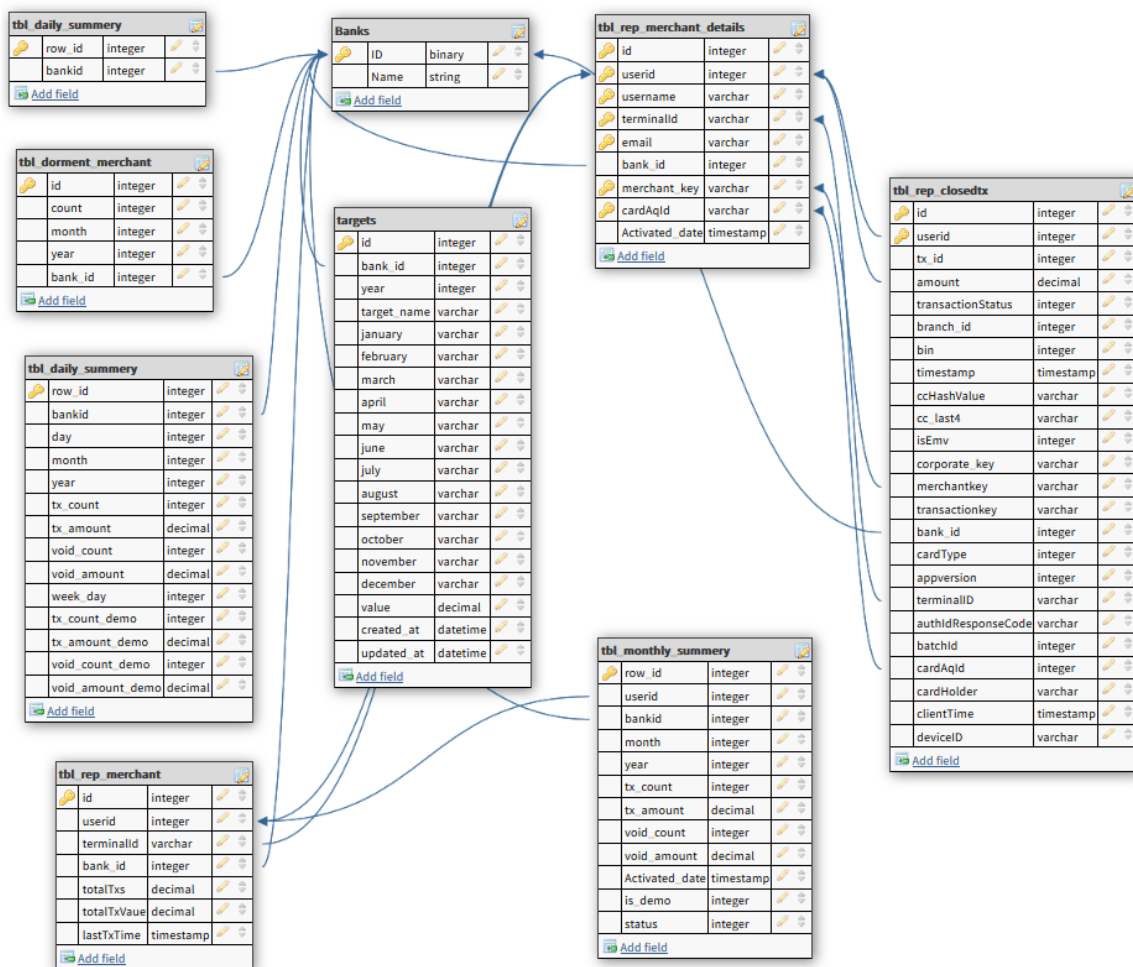


Figure B.13: Database Design for the System

User management data base diagram displayed under the figure B.14.

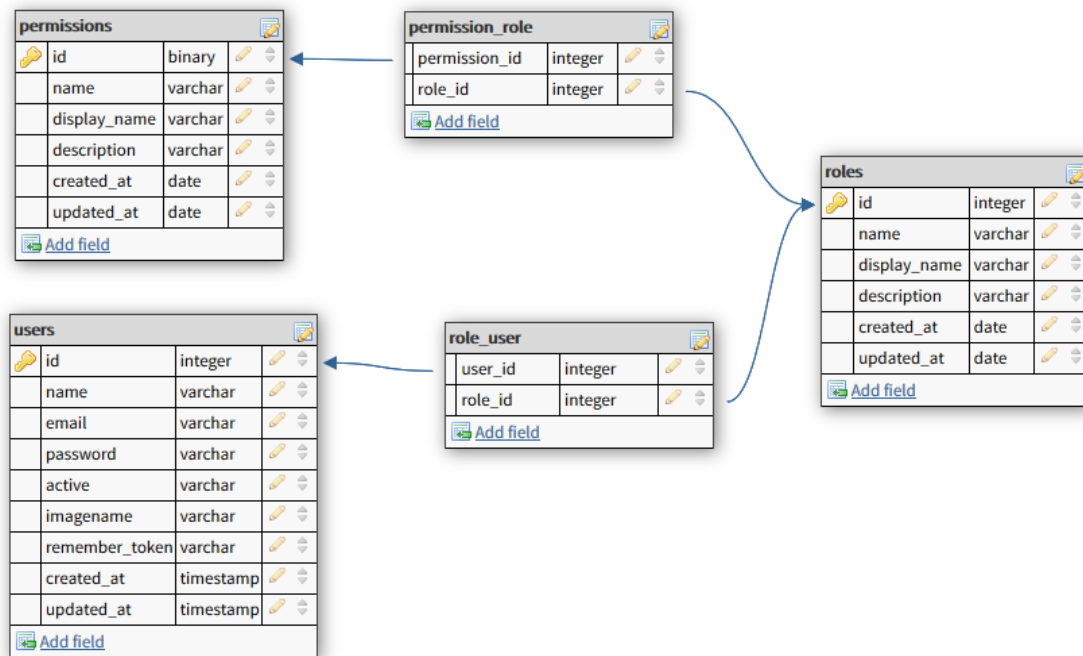



Figure B.14: Database Design for User Management

APPENDIX C - USER DOCUMENTATION

USER LOGIN

Following Figure C.1 shows User Login. Authorized person can log into the system. All users can log into the system using valid credentials. When user try to log into the system check whether this user is valid user or not otherwise system display error message.

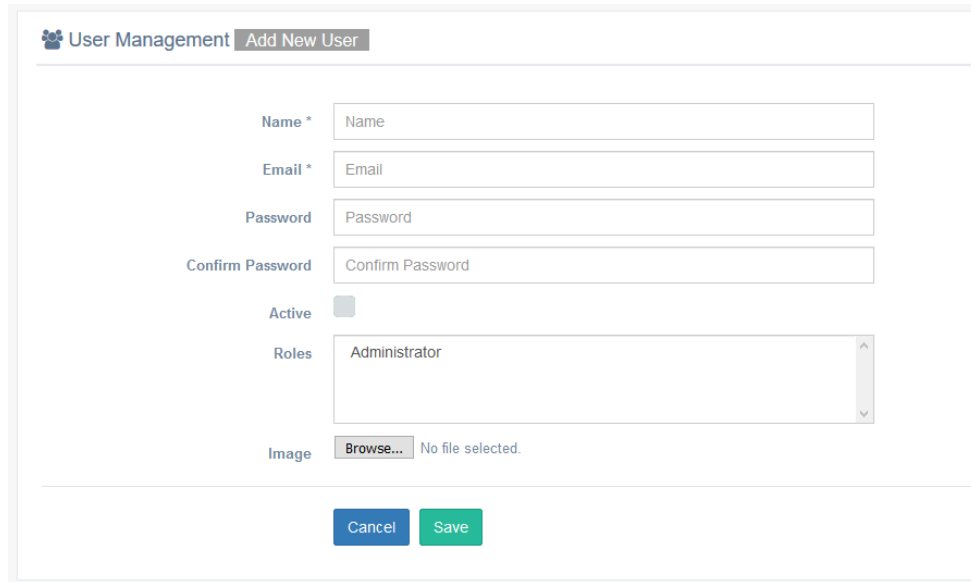


The image shows a user login interface for a system named 'payable'. The interface is centered on a light gray card against a teal background. At the top of the card is the 'payable' logo, which consists of a blue circle containing a white icon of a hand holding a document, followed by the word 'payable' in a blue sans-serif font. Below the logo is the text 'Log In' in a gray font. There are two input fields: 'Email' and 'Password', both with light gray borders and placeholder text. Below the password field is a checkbox labeled 'Remember Me'. A 'Login' button is positioned below the checkbox. At the bottom of the card, there is a copyright notice: '©2016 All Rights Reserved. PAYable.'

Figure C.1: Client Registration

ADD NEW USER

Following Figure C.2 shows User Creation form which allows only Admin user. User needs to fill all mandatory fields and click on Save to create new user.



The screenshot shows a web application interface for 'User Management' with a sub-tab 'Add New User'. The form contains the following fields and controls:

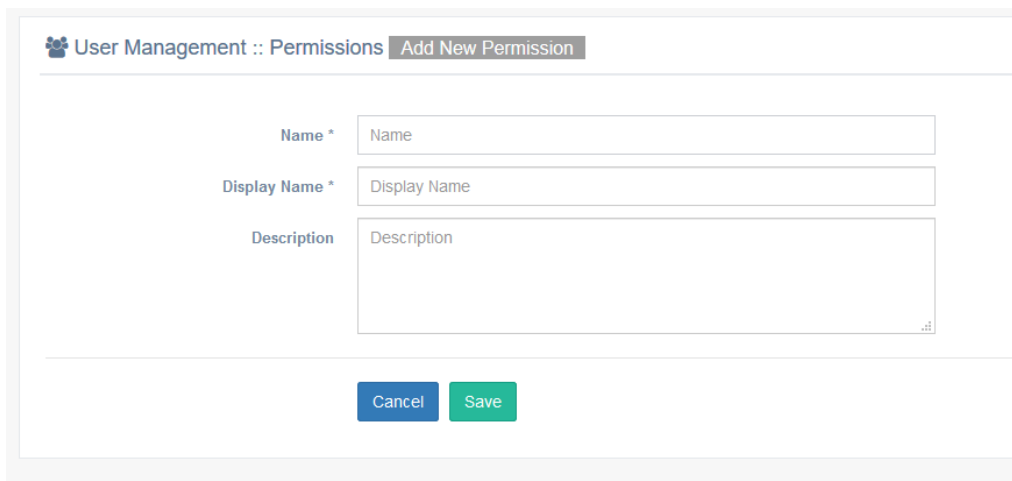
- Name ***: A text input field with the placeholder text 'Name'.
- Email ***: A text input field with the placeholder text 'Email'.
- Password**: A text input field with the placeholder text 'Password'.
- Confirm Password**: A text input field with the placeholder text 'Confirm Password'.
- Active**: A checkbox, currently unchecked.
- Roles**: A dropdown menu showing 'Administrator' as the selected option.
- Image**: A file upload section with a 'Browse...' button and the text 'No file selected.'

At the bottom of the form are two buttons: 'Cancel' (blue) and 'Save' (green).

Figure C.2: Add New User

ADD NEW PERMISSION

Following Figure C.3 shows Add Permission form which allows only Admin user. User needs to fill all mandatory fields and click on Save to create new permission.



The screenshot shows a web application interface for 'User Management :: Permissions' with a sub-tab 'Add New Permission'. The form contains the following fields and controls:

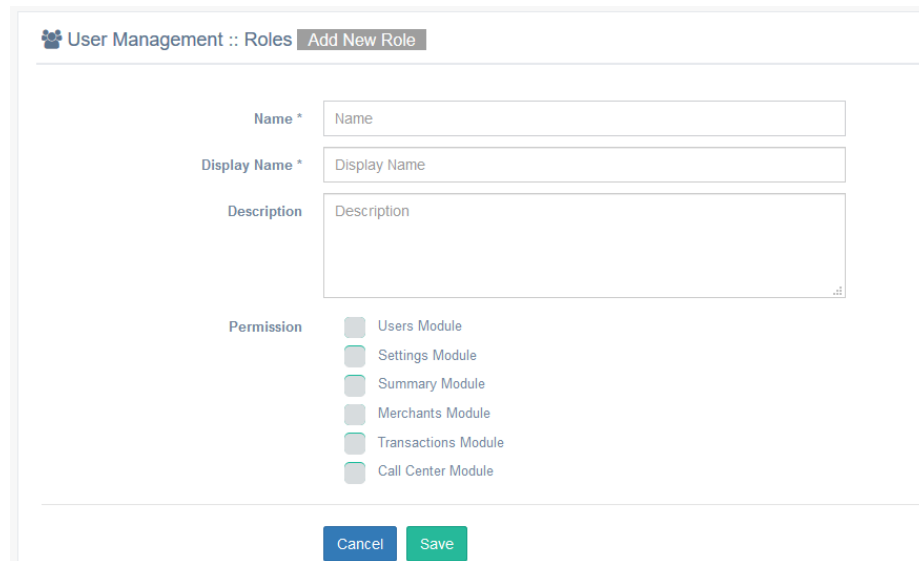
- Name ***: A text input field with the placeholder text 'Name'.
- Display Name ***: A text input field with the placeholder text 'Display Name'.
- Description**: A larger text input field with the placeholder text 'Description'.

At the bottom of the form are two buttons: 'Cancel' (blue) and 'Save' (green).

Figure C.3: Add New Permission

ADD NEW ROLE

Following Figure C.4 shows Add Role form which allows only Admin user. User needs to fill all mandatory fields and click on Save to create new Role.



User Management :: Roles Add New Role

Name *

Display Name *

Description

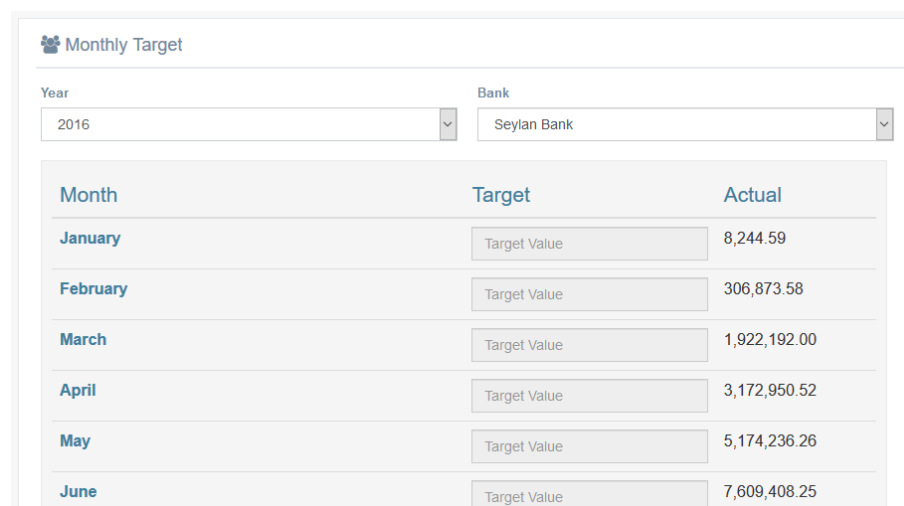
Permission

- ☐ Users Module
- ☐ Settings Module
- ☐ Summary Module
- ☐ Merchants Module
- ☐ Transactions Module
- ☐ Call Center Module

Figure C.4: Add New Role

SET MONTHLY TRANSACTIONS TARGET

Following Figure C.5 shows Monthly target form which allows only Admin user. User needs to fill target field and click on Save to set Target. User cannot access past months.



Monthly Target

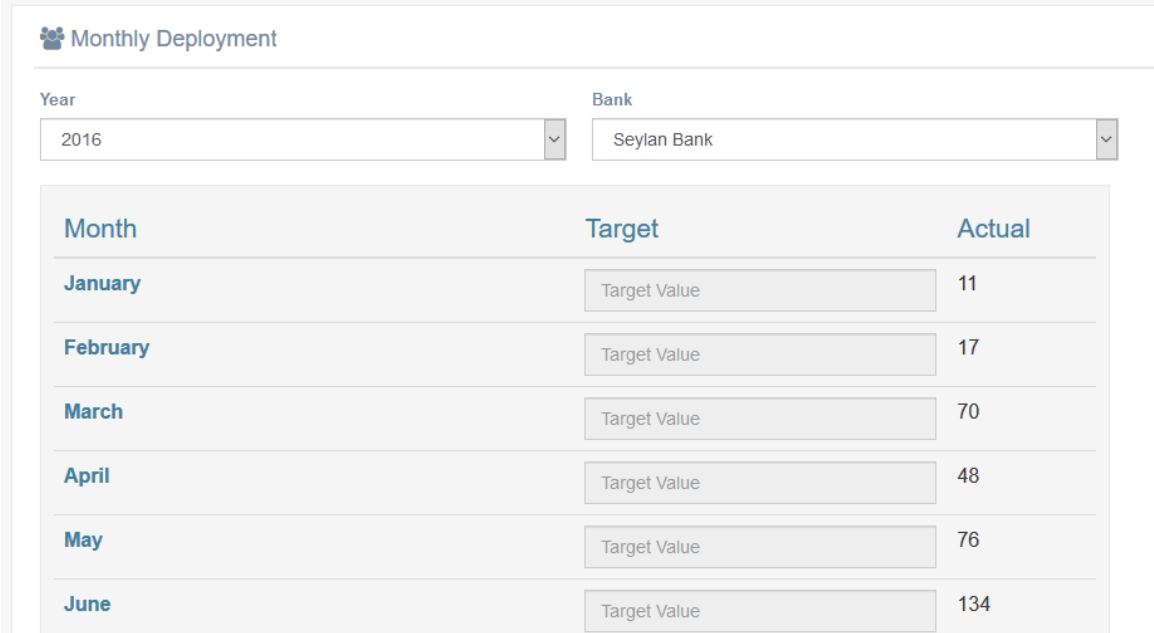
Year Bank

Month	Target	Actual
January	<input type="text" value="Target Value"/>	8,244.59
February	<input type="text" value="Target Value"/>	306,873.58
March	<input type="text" value="Target Value"/>	1,922,192.00
April	<input type="text" value="Target Value"/>	3,172,950.52
May	<input type="text" value="Target Value"/>	5,174,236.26
June	<input type="text" value="Target Value"/>	7,609,408.25

Figure C.5: Set Monthly Transactions Target

SET MONTHLY DEPLOYMENTS TARGET

Following Figure C.6 shows Monthly target form which allows only Admin user. User needs to fill target field and click on Save to set Target. User cannot access past months.

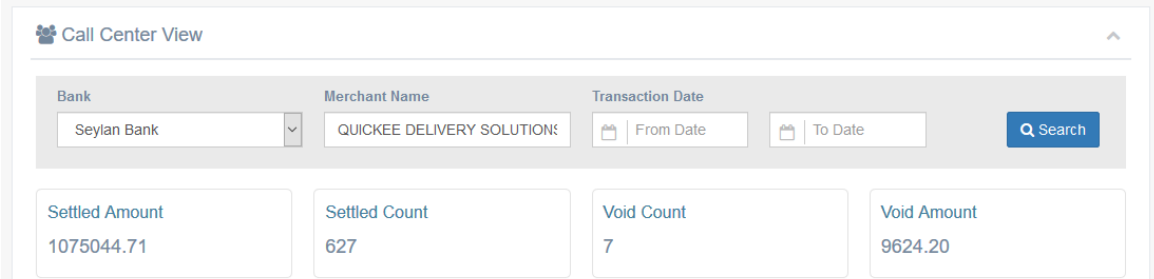


Month	Target	Actual
January	Target Value	11
February	Target Value	17
March	Target Value	70
April	Target Value	48
May	Target Value	76
June	Target Value	134

Figure C.6: Set Monthly Deployments Target

CALL CENTER VIEW

Following Figure C.7 shows call center view which allows Customer support users. User needs to select bank and enter merchant name to view transaction details according to each merchant. Can view transaction details by selecting date range.



Bank	Merchant Name	Transaction Date
Seylan Bank	QUICKEE DELIVERY SOLUTIONS	From Date To Date

Search

Settled Amount 1075044.71	Settled Count 627	Void Count 7	Void Amount 9624.20
------------------------------	----------------------	-----------------	------------------------

Figure C.7: Call Center View

TOTAL COUNT

Following Figure C.8 shows Transaction counts with 3 main categories. User needs to select bank using drop down list to view total counts.

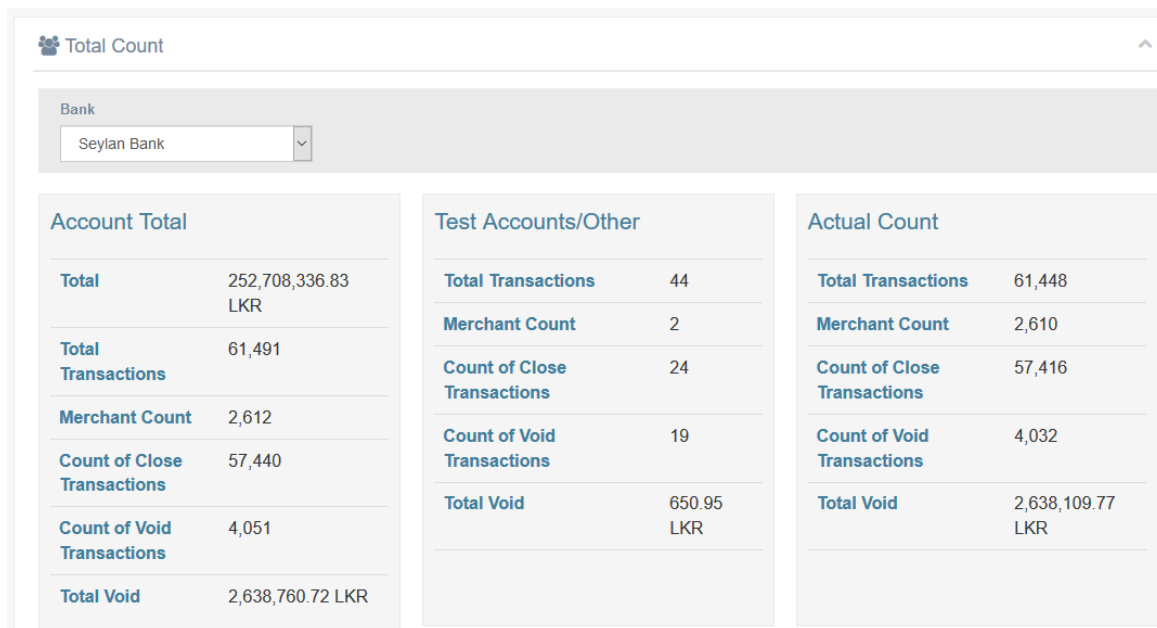


Figure C.8: Total Counts

APPENDIX D - MANAGEMENT REPORTS

SUMMARY REPORT

Figure D.1 shows total achievements of payable app according to bank.

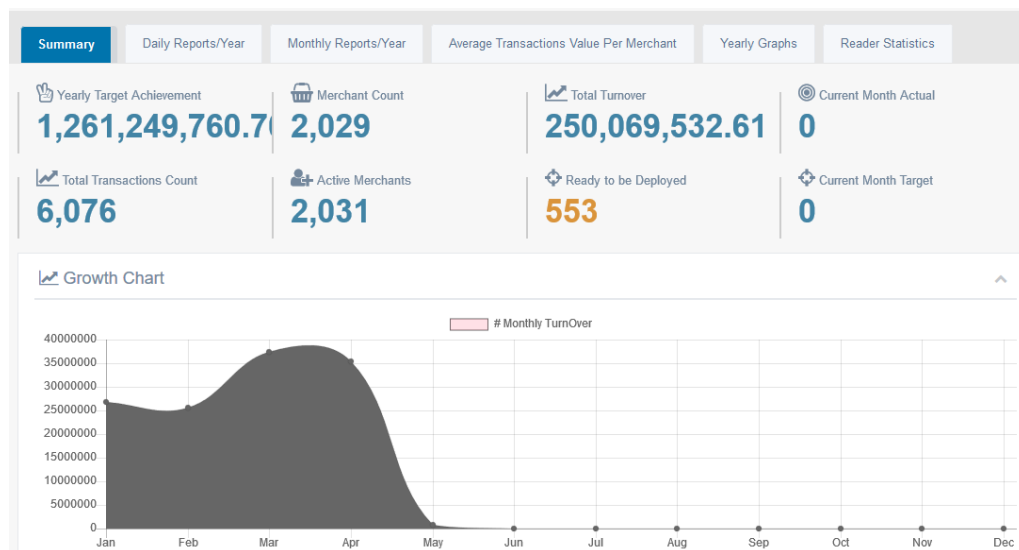


Figure D.1: Summery Report

DAILY USERS

Figure D.2 shows daily usage.

Daily Users								
	1 Sun	2 Mon	3 Tue	4 Wed	5 Thu	6 Fri	7 Sat	8 Sun
Daily Transaction Amount	299,803.11	744,153.74	528,400.59	647,968.49	933,230.92	978,683.21	974,344.01	642,356.76
Daily Active Merchants	59	95	77	95	88	98	103	75
New Merchants	1	0	1	1	2	0	2	1
Daily Transaction Count	119	173	123	179	155	149	204	134
Total Merchants of the Day	1005	1005	1006	1007	1009	1009	1011	1012
Daily Average Transaction Value	2,519.35	4,301.47	4,295.94	3,619.94	6,020.84	6,568.34	4,776.20	4,793.71
Daily Average Transaction Value for Active Merchants	5,081.41	7,833.20	6,862.35	6,820.72	10,604.90	9,986.56	9,459.65	8,564.76
Daily Average Transaction Value for All Merchants	298.31	740.45	525.25	643.46	924.91	969.95	963.74	634.74
Daily Average Transaction value without Dorment Merchants	573.24	1,422.86	1,008.40	1,234.23	1,770.84	1,857.08	1,841.86	1,211.99

Figure D.2: Daily Users

DAILY DEPLOYMENTS

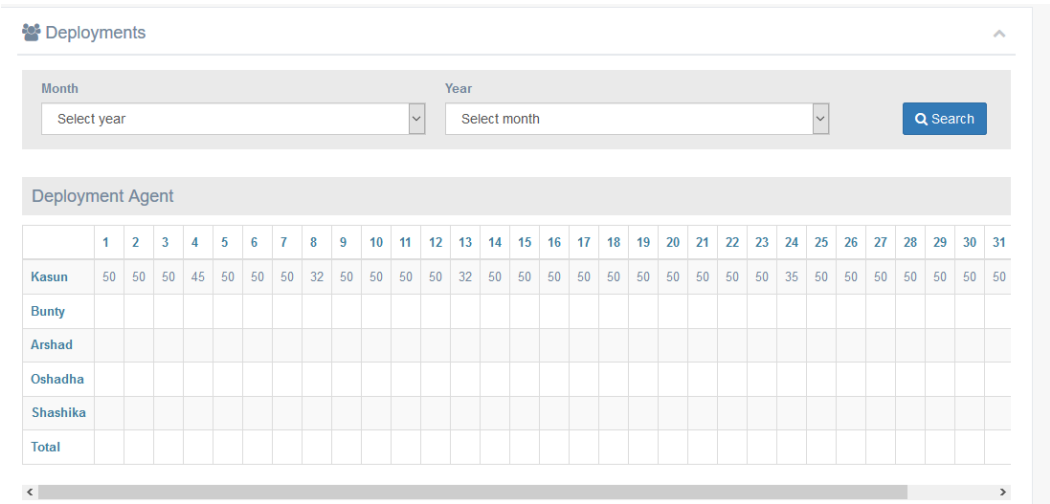


Figure D.3: Daily Deployments

DAILY TRANSACTION AVERAGE GRAPHS

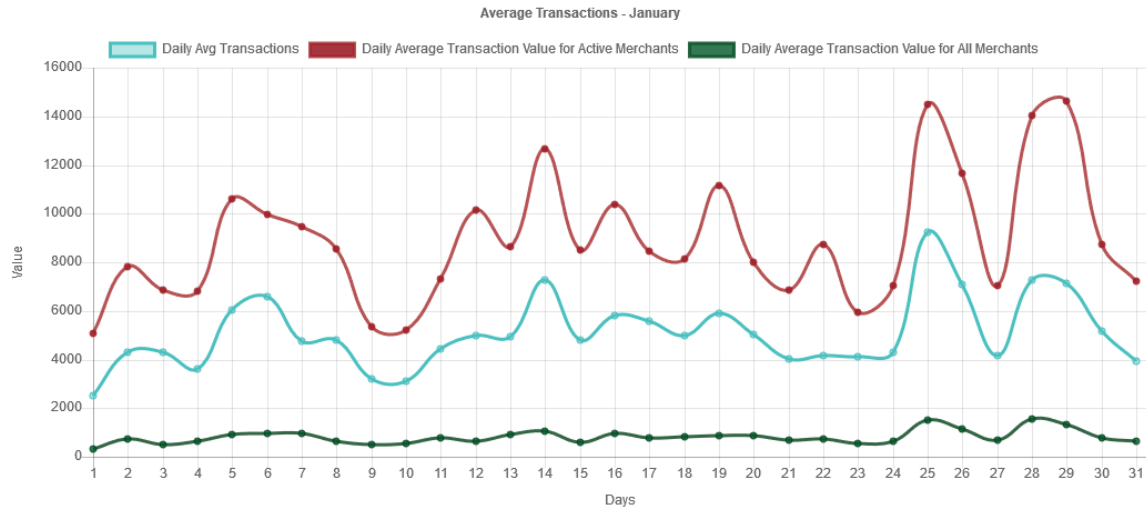


Figure D.4: Daily Transaction Average Graphs

DAILY TRANSACTION GRAPHS

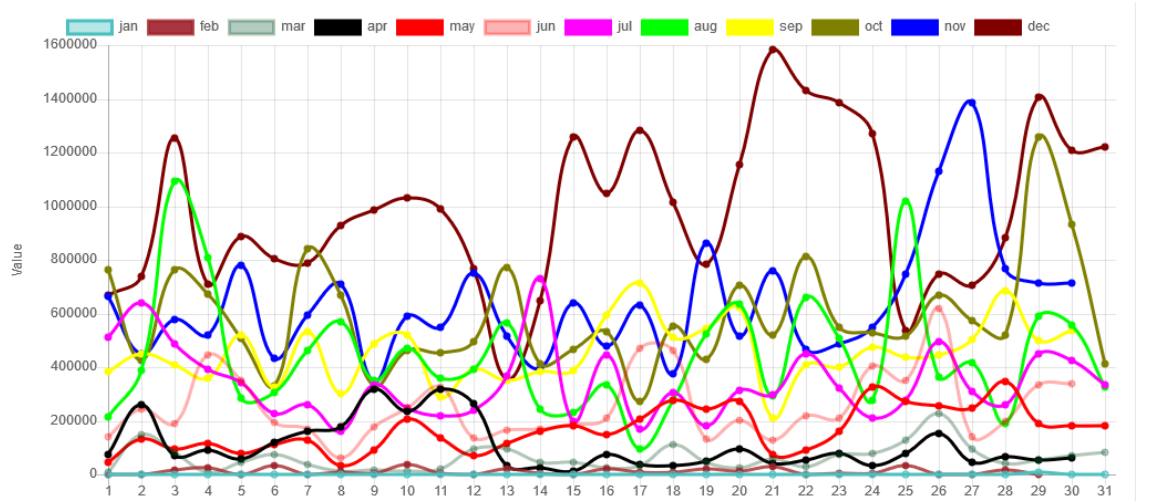


Figure D.5: Daily Transaction Graphs

APPENDIX E - TEST RESULTS

Major test cases at the evaluation stage along with test results are contained in this chapter.

TEST RESULTS FOR LOGIN

Test Case ID	Test Case	Last Result
1	Verify user can Login for PAYable-Reports successfully.	Pass
2	Verify user enter valid Email name and invalid password	Pass
3	Verify user enter invalid email and valid password.	Pass
4	Verify sign in attempts with leading spaces for Email	Pass
5	Verify user attempts to sign in with leading space for password	Pass
6	Verify user trying to sign in with trailing space for Email	Pass
7	Verify user trying to login with blank Credentials	Pass
8	Verify user trying to Login with blank password.	Pass
9	Verify user trying to login with blank email address	Pass
10	User trying to sign in to system with trailing spaces for password.	Pass
11	Verify functions of the system differ according to the sign in user	Pass

Table E.1: Test Results - Login

TEST RESULTS FOR USER MANAGEMENT

Test Case ID	Test Case	Last Result
12	Verify authorized user can access 'user management' tab	Pass
13	Verify 'Users' page loads with Existing merchants	Pass
14	Verify user can navigate 'Users, Roles and Permissions' tabs	Pass
15	Verify records paginations are applied properly.	Pass
16	Verify 'user management' page load with 'Users' tab	Pass
17	Verify Click on 'Create User'	Pass
18	Verify Fill all mandatory fields and Save details.	Pass
19	Verify mandatory fields should be indicated by asterisk (*) symbol.	Pass
20	Verify cannot save details without filling all mandatory fields	Pass
21	Verify select 'Roles' using drop down list.	Pass
22	Verify User can 'Active' users	Pass
23	Verify User can 'create users' without activation	Pass
24	Verify enter invalid Confirm password	Pass
25	Verify User can 'Deactivated' users	Pass
26	Verify User can edit existing users	Pass
27	Verify user cannot enter same 'Email' more than once	Pass
28	Verify enter valid Email.	Pass
29	Verify User can Delete existing users	Pass
30	Verify deactivate functionality for users on page should ask for confirmation	Pass
31	Verify confirmation message, before Deleting existing users	Pass

32	Verify user can view 'user' details	Pass
33	Verify Click on 'Create New Role'	Pass
34	Verify Fill all mandatory fields and Save details.	Pass
35	Verify mandatory fields should be indicated by asterisk (*) symbol.	Pass
36	Verify cannot save details without filling all mandatory fields	Pass
37	Verify select 'Permission' using check boxes	Pass
38	Verify User can edit existing Roles	Pass
39	Verify user cannot enter same 'Name' more than once	Pass
40	Verify User can Delete existing user Roles	Pass
41	Verify confirmation message, before Deleting existing user role	Pass
42	Verify user can view 'user role' details	Pass
43	Verify Click on 'Create New Permission'	Pass
44	Verify Fill all mandatory fields and Save details.	Pass
45	Verify mandatory fields should be indicated by asterisk (*) symbol.	Pass
46	Verify cannot save details without filling all mandatory fields	Pass
47	Verify user can delete assigned Permissions	Pass
48	Verify User can edit existing Permissions	Pass
49	Verify user cannot enter same 'Name' more than once	Pass
50	Verify User can Delete existing user Permissions	Pass
51	Verify confirmation message, before Deleting existing user Permissions	Pass
52	Verify user can view Permissions	Pass

Table E.2: Test Results – User Management

TEST RESULTS FOR CALL CENTER VIEW

Test Case ID	Test Case	Last Result
53	Verify authorized user can access 'call Center View' tab	Pass
54	Verify Call center view fields.	Pass
55	Verify user can be filter call center view data	Pass

Table E.3: Test Results – Call Center View

TEST RESULTS FOR MERCHANTS

Test Case ID	Test Case	Last Result
56	Verify authorized user can access 'Merchant' tab	Pass
57	Verify records paginations are applied properly.	Pass
58	Verify Merchat tab fields.	Pass
59	Verify merchant tab data table.	Pass
60	Verify user can be filter merchant data	Pass
61	Verify user can be filter transaction data	Pass

Table E.4: Test Results – Merchants

TEST RESULTS FOR TARGRT SETTING

Test Case ID	Test Case	Last Result
62	Verify authorized user can access 'Target setting' tab	Pass
63	Verify user can select option from bank and year in 'Monthly target' sub tab	Pass
64	Verify user can't enter the value in last month or year target value 'Monthly target' sub tab	Pass

65	Verify target value 'Monthly target' sub tab	Pass
66	Verify Actual value 'Monthly target' sub tab	Pass
67	Verify user can add target value 'Monthly target' sub tab	Pass
68	Verify user can access select option in 'Monthly deployment' tab	Pass
69	Verify user can't enter the value in last month or year target value 'Monthly Deployment' sub tab	Pass
70	Verify target value 'Monthly Deployment' sub tab	Pass
71	Verify Actual value 'Monthly Deployment' sub tab	Pass
72	Verify user can add target value 'Monthly Deployment' sub tab	Pass

Table E.5: Test Results – Target Setting

TEST RESULTS FOR TOTAL COUNT

Test Case ID	Test Case	Last Result
73	Verify authorized user can access 'Total Count' tab	Pass
74	Verify user can select option from bank dropdown	Pass
75	Verify the 'Account Total' data base on bank name	Pass
76	Verify the 'Test Accounts/Other' data base on bank name	Pass
77	Verify the 'Actual Count' data base on bank name	Pass

Table E.6: Test Results – Total Count

TEST RESULTS FOR SUMMARY

Test Case ID	Test Case	Last Result
78	Verify Summary page loads properly	Pass

79	Verify user can select bank	Pass
80	Verify filtered data changed according to the bank	Pass
81	Verify Summary tab loads properly	Pass
82	Verify Daily Reports/Year loads properly	Pass
83	Verify user can be select option field in 'Daily Reports/Year tab'	Pass
84	Verify Daily Users details	Pass
85	Verify target details	Pass
86	Verify authorized user can access 'Monthly Report' sub tab	Pass
87	Verify user can select option from dropdown in 'Monthly Report' sub tab	Pass
88	Verify 'Monthly Report' for every months base on selected year	Pass
89	Verify authorized user can access 'Average Transactions Value Per Merchant' sub tab	Pass
90	Verify user can select option from dropdown in 'Average Transactions Value Per Merchant' sub tab	Pass
91	Verify 'Average Transaction Value' for every months base on selected year	Pass
92	Verify authorized user can access 'Yearly Graphs' tab	Pass
93	Verify user can access details in 'Yearly Graphs' tab	Pass
94	Verify user can select option from dropdown	Pass
95	Verify 'Transactions' data base on selected year	Pass
96	Verify 'Active Users' data base on selected year	Pass

Table E.7: Test Results – Summary

APPENDIX F - CODE LISTING

CODE FOR LOGIN MODULE

Authentication Defaults

This option controls the default authentication "guard" and password reset options for your application. You may change these defaults as required, but they're a perfect start for most applications.

```
<?php
return [
    'defaults' => [
        'guard' => 'web',
        'passwords' => 'users',
    ],
];
```

Authentication Guards

Next, you may define every authentication guard for your application. Of course, a great default configuration has been defined for you here which uses session storage and the eloquent user provider. All authentication drivers have a user provider. This defines how the users are actually retrieved out of your database or other storage mechanisms used by this application to persist your user's data.

Supported: "session", "token"

```
'guards' => [
    'web' => [
        'driver' => 'session',
        'provider' => 'users',
    ],
    'api' => [
        'driver' => 'token',
        'provider' => 'users',
    ],
],
```

User Providers

All authentication drivers have a user provider. This defines how the users are actually retrieved out of your database or other storage mechanisms used by this application to persist your user's data. If you have multiple user tables or models you may configure multiple sources which represent each model / table. These sources may then be assigned to any extra authentication guards you have defined.

Supported: "database", "eloquent"

```
'providers' => [  
    'users' => [  
        'driver' => 'eloquent',  
        'model' => App\User::class,  
    ],  
]
```

Resetting Passwords

You may specify multiple password reset configurations if you have more than one user table or model in the application and you want to have separate password reset settings based on the specific user types. The expire time is the number of minutes that the reset token should be considered valid. This security feature keeps tokens short-lived so they have less time to be guessed. You may change this as needed.

```
'passwords' => [  
    'users' => [  
        'provider' => 'users',  
        'table' => 'password_resets',  
        'expire' => 60,  
    ],  
],  
];
```

CODE FOR SUMMARY MODULE

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;
use DB;

class Summary extends Model
{
    public function get_daily_txn_summary($data){

        $query = DB::table('tbl_daily_summery');
        // $query-
        >select(['*',DB::raw("tx_count+void_count AS
totalCount"),DB::raw("tx_amount AS totalAmount")]);
        if(is_numeric($data['bank'])){
            $query->where('bankid',$data['bank']);
        }
        if(isset($data['month'])){
            $query->where('month',$data['month']);
        }
        $query->where('year',$data['year']);

        return $query->get();
    }

    public function get_monthly_txn_summary($data){

        $query = DB::table('tbl_monthly_summery');
        $query->select(DB::raw('month, SUM(tx_amount) AS
closed_amount,SUM(void_amount) AS void_amount,SUM(tx_count) AS
closed_count,SUM(void_count) AS void_count'));
        if(is_numeric($data['bank'])){
            $query->where('bankid',$data['bank']);
        }
        $query->where('year',$data['year']);
        $query->where('is_demo',2);//get only real accounts data

        $query->groupBy('month');
        $query->orderBy('month', 'ASC');

        return $query->get();
    }
}
```

```

public function get_monthly_activated_merchant_count($data){

    $query = DB::table('tbl_rep_merchant_details');
    $query->select(DB::raw('MONTH(Activated_date) AS
month,count(*) as activated_count'));
    $query->whereYear('Activated_date', $data['year']);
    $query->where('businessName','!=','PAYable DEMO');

    if(is_numeric($data['bank'])){
        $query->where('bank_id',$data['bank']);
    }
    $query->groupBy('month');
    $query->orderBy('month', 'ASC');

    return $query->get();
}

// DAILY ACTIVE MERCHANTS (Merchants who done the transactions
for particular day)
public function get_daily_active_merchants($data){

    $query = DB::table('tbl_rep_closedtx AS t');
    $query->select(DB::raw('DATE(t.timestamp) AS date,
COUNT(DISTINCT t.userid) AS users_txn_count'));
    if(isset($data['month'])){
        $query->whereMonth('t.timestamp', $data['month']);
    }
    $query->whereYear('t.timestamp', $data['year']);
    $query->join('tbl_rep_merchant_details AS m', 'm.userid',
'=', 't.userid');
    $query->where('m.businessName','!=','PAYable DEMO');
    $query->where('t.transactionStatus', 2);

    if(is_numeric($data['bank'])){
        $query->where('t.bank_id',$data['bank']);
    }
    $query->groupBy('date');
    $query->orderBy('date', 'ASC');
    //print_r($query->toSql());

    return $query->get();
}

// NEW MERCHANTS (number of merchants who are done FIRST
transaction for the day)
public function get_daily_new_merchants($data){

    $query = DB::table('tbl_rep_merchant_details');
    $query->select(DB::raw('count(*) as activated_count,
DATE(Activated_date) AS date'));
    $query->whereMonth('Activated_date', $data['month']);

```



```

        $query->whereYear('Activated_date', $data['year']);
        $query->where('businessName','!=','PAYable DEMO');
        if(is_numeric($data['bank'])) {
            $query->where('bank_id',$data['bank']);
        }
        $query->groupBy('date');
        $query->orderBy('date','ASC');
        //print_r($query->toSql());
        return $query->get();
    }

// Total active merchants until particular day
public function total_merchants_for_day($data,$date){

    $query = DB::table('tbl_rep_merchant_details');
    $query->whereDate('Activated_date','<=',$date);
    $query->where('businessName','!=','PAYable DEMO');

    if(is_numeric($data['bank'])) {
        $query->where('bank_id',$data['bank']);
    }
    $query->where('status',3);
    return $query->count();
}

// DORMENT MERCHANTS START- Merchants who haven't done
trasactions for the specific period
public function get_dorment_merchants_for_period($data){

    extract($data);

    $query = DB::table('tbl_dorment_merchant');
    $query->select('count');
    $query->where('year', $year);
    $query->where('month', $month);
    if(is_numeric($bank)) {
        $query->where('bank_id',$bank);
    }
    $result =$query->first();

    if($result){
        return $result->count;
    }else{
        return 0;
    }
}
}

```

CODE FOR TRANSACTIONS

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Transaction extends Model
{
    protected $table = 'tbl_rep_closedtx';
}
```

CODE FOR MERCHANTS

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Merchant extends Model
{
    protected $table = 'tbl_rep_merchant_view';
}
```

CODE FOR MONTHLY SUMMARY

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class MonthSummary extends Model
{
    protected $table = 'tbl_monthly_summery';
}
```

APPENDIX G - CLIENT CERTIFICATE



No: 09/14, Daminagahawatta, Morenda, Kesbewa.
Office: 072 2982982 Mobile: 077 7726534

08th November 2017

Project Examination Board,
University of Colombo School of Computing,
221/2A, Dharmapala Mawatha,
Colombo – 07.

Dear Sir/ Madam,

LETTER OF CERTIFICATION

I would like to inform you that Miss. Nadeesha Kumari (0923818) has successfully designed and developed a Key Performance Indicator System for Smart Technology. The project was undertaken by her as a partial fulfillment of a requirement for the Bachelor of Information Technology Degree Program.

I glad to say that this system has facilitated to increase the productivity of Deployments and trace merchants. We strongly believe that this system will help us to improve our business significantly in further.

I would like to thank Miss. Nadeesha for her time and effort that she has extended towards the completion of this valuable system. She has successfully completed our business requirements. I wish to keep in touch with Miss. Nadeesha for further updates and her guidance on the functioning of our company.

This certification is issued in the request of Miss. Nadeesha Kumari.

Thank You
Yours Faithfully,

SMART TECHNOLOGY

9/14, Daminagahawatta, Morenda, Kesbewa
Phone: 072 2982982, 077 7726534


G. Chamal Roshan Suraweera

CEO – Founder
Smart Technology

Figure G.1: Client Certificate

INDEX

Class Diagram, 17
Code, 19, 21, 41, 66, 68, 71
customer, 35
Database, 19, 41, 50, 51
Degree, 37
Download, 41
Hardware, 18, 40
Interface, 43, 44, 45, 46, 47, 48, 49
Management, 57
server, 19, 23, 37
Software, 18, 24, 40, 41
system, iv, 15, 16, 17, 18, 19, 23, 24, 25,
26, 27, 29, 35, 36, 37, 40, 42, 47, 50, 52
Technologies, 19
Test Cases, 29, 30, 32, 33, 34
Testing, 24, 25, 26, 27, 35
Tools, 19
use case, 15, 16
Use Case Narrative, 16
validation, 19, 23, 27