An Undergraduate Internship Report on Plausible SaaS Application

By

Mohammed Shahadat Hossain

Student ID: **1731555**

Summer, 2021

Supervisor:

Asif Bin Khaled

Supervisor's Designation

Department of Computer Science & Engineering

Independent University, Bangladesh

September 11, 2021

Dissertation submitted in partial fulfillment for
The Degree of Bachelor of Science
in
Computer Science & Engineering

Independent University, Bangladesh

Attestation

I, Mohammed Shahadat Hossain, ID: 1731555, hereby declare my work to be in compliance with all rules and regulations for the internship project and Non Disclosure Agreement(NDA) with Ebexsoft IT Solutions.

Shahadat	02 Sept 2021
Signature	Date

Mohammed Shahadat Hossain

Name

Acknowledgement

First and foremost, I must express my deepest sense of gratitude to Almighty Allah, it is because of His mercy and blessing that I have come this far. It has been a great privilege to work for Ebexsoft.

I have learned a lot of things I previously had no idea about. I find myself extremely lucky that such experienced members have shared their time and knowledge with me and have guided me through this process. I would like to convey my special thanks to my supervisor, also CEO of Ebexsoft, Md. Mamunur Rashid for his continuous guidance.

I would like to express my gratitude to my internal supervisor, Asif Bin Khaled, Lecturer, Department of Computer Science and Engineering, Independent University, Bangladesh (IUB), for his instructions, guidance and support in preparation of this report.

Lastly, I proudly acknowledge the great sacrifices, support, advice, inspirations and encouragements from my family members and friends.

Mohammed Shahadat Hossain

Letter of Transmittal

2nd September 2021
Asif Bin Khaled
Lecturer,
Department of Computer Science and Engineering,
Independent University, Bangladesh

Subject: Letter of Transmittal for Internship Report, Summer 2021

Respected Faculty,
With due honor, this is to inform you that, I, Mohammed Shahadat Hossain, ID: 1731555 from Internship Course of Summer 2021 Semester, Section 10, would like to submit my Internship report. This report is based on my internship program and the project I have worked on. My internship period started from February 2021 and it is expected to end on the August 2021 at Ebexsoft.

This report is based on my experience and the work I did at Ebexsoft during my internship program. The primary goal for my internship was to gain experience from working in the software engineering industry and familiarize myself with all the different technology related fields of the company, documentation, software development and to get acquainted with software development processes and practices.

Over the period of my internship at Ebexsoft, I had to learn and adapt to the evolving technologies being used in the Software as a Service (Saas) applications. I hope the following report can achieve your approval and is adequate.

Sincerely yours,
Mohammed Shahadat Hossain, 1731555

Evaluation Committee

.....

Signature

.....

Name

.....

Supervisor

.....

Signature

.....

Name

.....

Internal Examiner

.....

Signature

.....

Name

.....

External Examiner

.....

Signature

.....

Name

.....

Convener

Abstract

The software industry at this moment is dominated by Software as a Service (Saas) applications. It means the software is hosted by a provider and delivered to the customer over the internet as a service to themselves. In recent years, It has taken the world by storm with popular services like Slack, Discord, Google Classroom etc[1].

The SaaS scene in Bangladesh is relatively new and has just started off but gaining traction and popularity very fast. SaaS applications are automating businesses, making it efficient and environmentally sustainable. But most SaaS applications are targeted towards middle to large enterprise or business owners which makes it hard to afford for startups or small business owners [2].

Plausible aims to deliver an all in one inventory management solution for small business owners with an affordable pricing. Because of the pandemic we are seeing a highly increased number of new businesses with poor management systems in place. We believe Plausible can remedy the situation and make their business efficient enough to turn a profit and make decisions from an analytical viewpoint.

To make things affordable, Plausible will be a monthly subscription based service, developed on the MERN Stack. The software has 3 core features. They are analytic, point of sales and management. It has 12 total components with subsections in those components for the core features. The primary components are:Dashboard, Product, Sale, Expense, Quotation, Transfer, Return, Accounting, HRM, Users, Reports Settings. The software will be online based, accessible and manageable by the any computer with internet using the right login credentials.

Contents

Attestation i

Acknowledgement ii

Letter of Transmittal iii

Evaluation Committee iv

Abstract v

1 Introduction 1

1.1 Background of the Work 1

1.2 Objectives 1

1.3 Scopes 1

2 Literature Review 2

2.1 Relationship with Undergraduate Studies 2

2.2 Related works 3

3 Project Management 4

3.1 Work Breakdown Structure 4

3.2 Process/Activity wise Time Distribution 5

3.3 Gantt Chart 5

4 Methodology 6

4.1 Software Development Life Cycle: 6

4.2 Rapid Application Development 6

5 Body of the Project 9

5.1 Work Description 9

5.2 System Analysis 9

5.2.1 Six Element Analysis 10

5.2.2 Feasibility Analysis 10

5.2.3 Problem Solution Analysis 10
5.3 System Design 11
5.3.1 Rich Picture 11
5.3.2 Software Architecture 11
5.3.3 Database Diagrams 13
5.3.4 Functional and Non-Functional Requirements 16
5.4 Product Features 16
5.4.1 Product UI 18

6 Results & Analysis **20**

7 Project as Engineering Problem Analysis **26**
7.1 Sustainability of the Project/Work 26
7.2 Social and Environmental Effects and Analysis 27
7.3 Addressing Ethics and Ethical Issues 27

8 Lesson Learned **28**
8.1 Problems Faced During this Period 28
8.2 Solution of those Problems 28

9 Future Work & Conclusion **30**
9.1 Future Works 30
9.2 Conclusion 30

Bibliography **31**

List of Figures

3.1 WBS 4
3.2 Critical Path Tracker 5
3.3 Gantt Chart 5

4.1 RAD Methodology 7

5.1 Plausible Rich Picture 11
5.2 MVC Architecture 12
5.3 Database Diagram Full Image 13
5.4 Database Diagram Cropped 14
5.5 Database Diagram Cropped 15
5.6 Dashboard UI 18
5.7 Create Sale 18
5.8 Sale List 19
5.9 Customer List 19

Chapter 1

Introduction

1.1 Background of the Work

We are in a pandemic called COVID-19. Because of the pandemic people are starting businesses out of boredom or desperation from unemployment with a small fund of savings. Some are also just looking for a second source of income to provide stability and security to their life. Because of this, there is an influx of new e-commerce platforms and small businesses with poor management systems in place.
I believe this project can increase their productivity by a noticeable margin and help them work towards a clear goal to profit by observing the analytic and make profitable decisions. Most Solutions are expensive and not affordable by small businesses, therefore we at Ebexsoft decided to go with a monthly subscription service to make it more affordable to the masses.

1.2 Objectives

The aim of this project is to deliver an affordable, analytical inventory management solution with point of sales software for small business owners so that they don't suffer anymore from poor management issues.

1.3 Scopes

Plausible is a Software As A Service (SAAS) application. The software has 3 core features. They are analytic, point of sales and management. It has 12 total components with subsections in those components for the core features. The primary components are: Dashboard, Product, Sale, Expense, Quotation, Transfer, Return, Accounting, HRM, Users, Reports Settings. The software will be online based, accessible and manageable by the any computer with internet using the right login credentials.

Chapter 2

Literature Review

2.1 Relationship with Undergraduate Studies

Almost all programming courses that are offered have provided knowledge that made this project sail smoothly. While some courses were implicitly helpful others provide knowledge that was directly incorporated in this project

Even before my enrollment, I was knowledgeable on basic web development technologies like HTML, CSS. But I did not have a good understanding of fundamental programming concepts which cannot be self learned. My first programming course, CSE101 helped me build up that knowledge and made my foundation strong. It introduced variables, functions, objects and methods while touching on simple algorithms. Moving forward, Data Structure(CSE203) introduced me to the concepts of Stack, Queue, Recursion, BST, Heap Priority Queue, Array, Linked List. This was a notch above **CSE101** and prior to the course, I did not know about any of these. Then, **CSE213**, introduced me to OOP concepts. This is one of the course that impacted me not because of the content but because of the faculty that teaches it. I thoroughly enjoyed his class and he has given me a good understanding of the OOP concepts. This is my first course where I built my first somewhat useful application. It taught me the importance of making a "somewhat useful application" and leaving room from improvement later. After which, **CSE303** introduced me to Relational and Non-relational Databases, which I have used extensively in my project plausible, the back-end of it is MySQL, used through phpMyAdmin(GUI).**CSE303** is the course which turned me into a software engineer from just a programmer. I build my first full-stack web java FXML application for it. It introduced me to a lot of new concepts such as System Development Life Cycle, Rich Picture, Requirement Analysis, Entity Relationship Diagram, Business Process Model and Notation Diagram and many more. This helped fortify my projects security. In,**CSE309** , I learned the front end technology using PHP, which I have used extensively in my project. My project uses Laravel, which is a framework of PHP. It works by having PHP as a

core to get HTML in the front end and fetching data using rest commands from Mysql Backend. This course covered the important topics such as Javascript, Bootstrap, PHP and Mysql which is the underlying foundation for my Project.

2.2 Related works

I believe Inventory Management with POS is a common software that almost every software company provides but not affordable by small business owners(SME).

What makes our software unique is the fact that it is going to be a subscription based service, basically SaaS. Most companies in our country are profit oriented and is not interested in the small market of entrepreneurs and small business owners. They choose a one time profit margin of corporate clients over helping the masses. We believe that we can kick-start a wave in subscription based software services with our application. The goal is to provide an affordable solution to the SME so that they can automate their businesses just like their corporate counterparts/rivals, giving them a competitive and technological edge.

Chapter 3

Project Management

3.1 Work Breakdown Structure

Below is the WBS of the Plausible project.

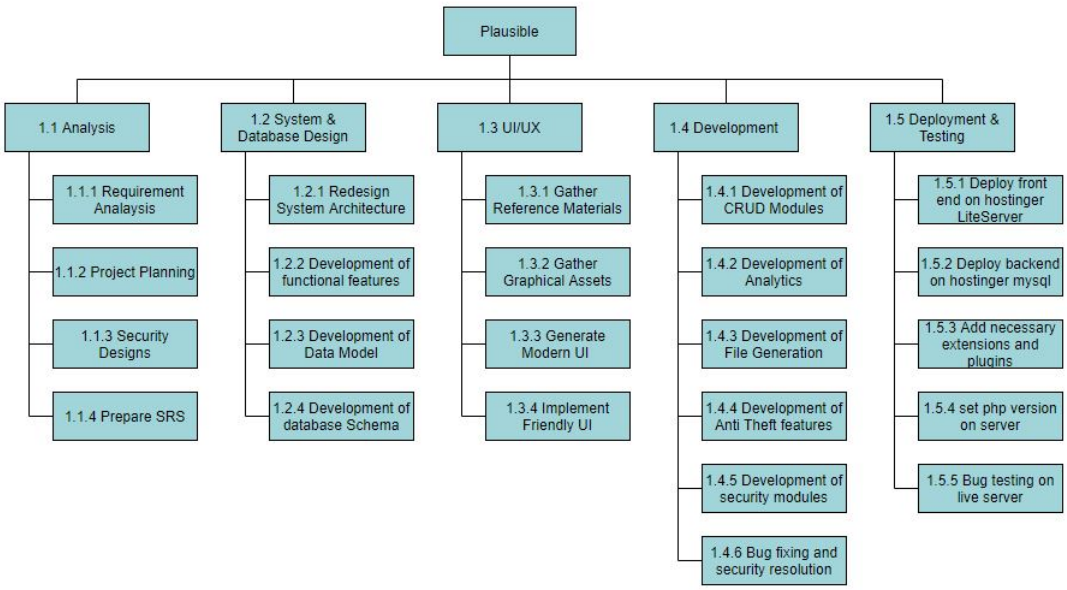


Figure 3.1: WBS

3.2 Process/Activity wise Time Distribution

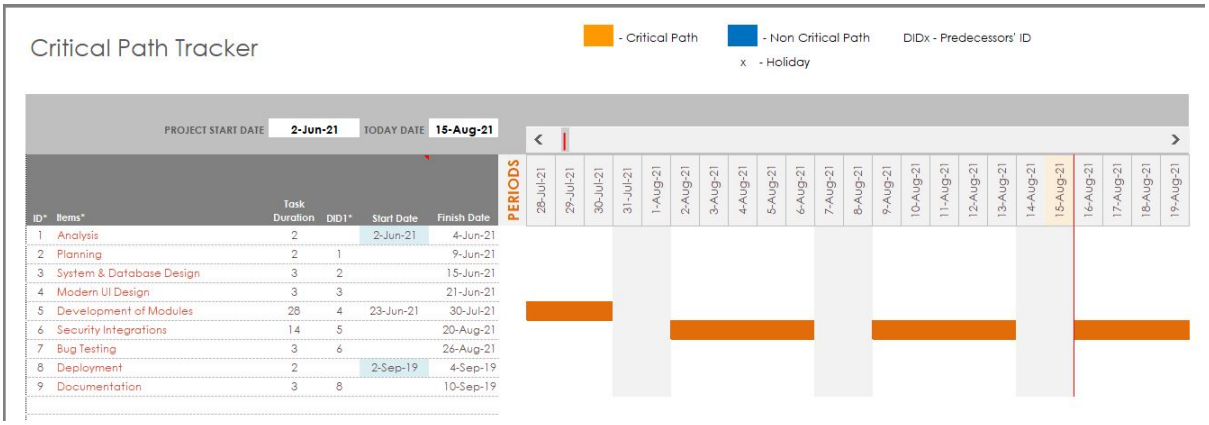


Figure 3.2: Critical Path Tracker

3.3 Gantt Chart

The Chart here functions in a way where the highlighted number, is the number of weeks. changing it allows to scroll through the Gantt chart.

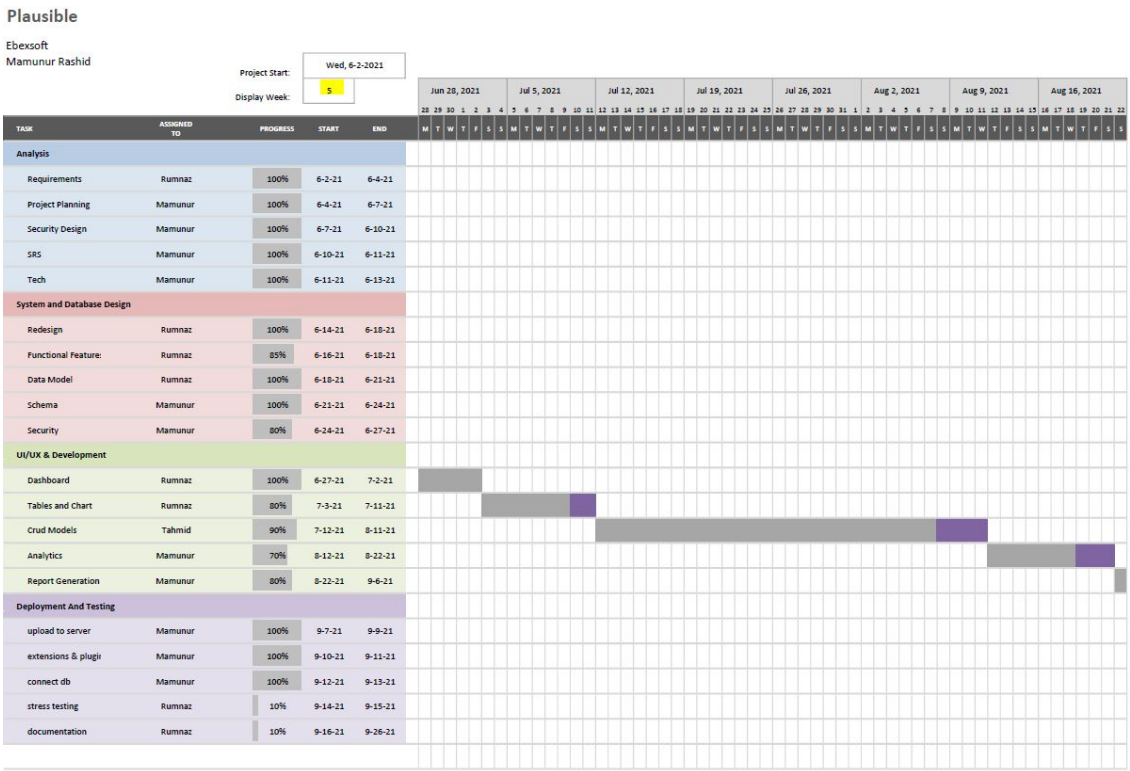


Figure 3.3: Gantt Chart

Chapter 4

Methodology

4.1 Software Development Life Cycle:

Software development process is the process of dividing software development work into distinct phases to improve design, product management and project management. It is also known as a system development life cycle (SDLC). We can define SDLC as a framework that describes the activities performed at each stage of a System Development Project. So, it has some basic stages to be followed during the development phase. There many different SDLC to choose from like

- Rapid Application Development
- Waterfall Model
- Prototyping
- Agile
- Spiral Model
- V-Model
- Incremental
- Evolutionary Model

4.2 Rapid Application Development

The reason I choose RAD is because my system is modeled towards an older existing system that had some security design flaws. As the requirements are clear and the system model is finalized, Our team decided on RAD to spend the least amount of time

upgrading to a meaningful solution.

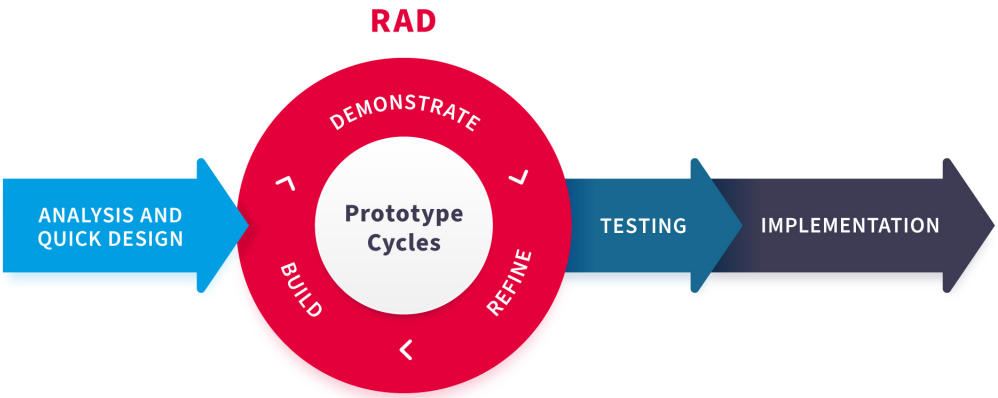


Figure 4.1: RAD Methodology

Steps in RAD

When considering the rapid application development methodology there are some key steps that need to be understood. These steps are as follows[3]:

- Understand the requirements – starting a project without a clear understanding of what the outcome should look like is a recipe for failure. In this step, you are trying to define the goals of the project as well as the budget and delivery time-frame. After this step is complete and all parties are satisfied then you can move onto the next one.
- Create a Prototype – a minimum viable product or MVP is the initial prototype, you should build as you begin developing. The chances of a successful prototype are significantly higher when the development teams, project leads and clients all work closely together throughout the build phase.
- Solicit feedback – during this phase, developers are looking for users' feedback. This could be on the usability of the product or even its interface, look and feel. As developers gather information, adjustments can be made to better tailor the product to user needs.
- Testing – during the testing phase, the solution needs to be checked closely against all client requirements to ensure that it is functioning as required. In addition, testing should also review any third-party integration's within the software, as changes made in one area can impact others. Once your finished product is ready to go live, you need to train users on its use and ensure that all of the correct data is loaded into it.

Advantages of RAD[3]

Due to its flexibility and adaptability to new inputs, a RAD approach carries far less risk than a basic plan-based method. With an early prototype, it is fairly easy to identify any key challenges associated with the project. As such, RAD weeds out any potential problems early on in the life-cycle, making it cheaper and easier to address during development. As a direct result, RAD projects typically take a shorter time to complete.
RAD approaches make it easier to deal with any budgetary drawbacks. Due to its flexibility and incremental nature, a RAD method allows developers to identify and tackle monetary and technical issues faster and react accordingly. Compared to a Waterfall approach, the risk of any large-scale failures is drastically lower.

Disadvantages RAD[3]

There are some key drawbacks when it comes to RAD approaches, as the flexibility and user functionality come with some trade-offs. The emphasis on user experience and feedback could in effect reemphasize non-functional requirements (or NFRs) in the development process. To put it simply, focusing on improving what the software does (functional) might neglect the system's architecture (non-functional) and overall structure. While NFRs are not visible to an end user, they are important for a software's longevity.

Implementation of RAD in Plausible

We believe that RAD was perfect for our project as we are always looking to add new features to it. The requirements of the project was very clear as it is treated as a product for the company rather than software for a client demand. After the initial requirement analysis phase, we created a prototype which met the requirements. Since it was a prototype, the visual outlook was not as good, but it served to demonstrate the functional capabilities. After which the software was given to a few small businesses to use for 1 week to gather feedback and most notable response we got from the users is that, the business owner needs to have an option for managing older customer dues before they started using the software. It was implemented later on, As RAD leaves room from improvement. With RAD the goal is not to develop a perfect product, but to develop a manageable functioning software which can be upgraded later on [4].

Chapter 5

Body of the Project

5.1 Work Description

Plausible is Software As A Service(SAAS) application designed to be an anti-theft inventory management solution with point of sales (POS). Although some solutions provide the necessary analytic and report generations, The main issue with most inventory management solutions is they only provide the bare minimum security services because the modules are not interconnected with each other. In Plausible, we have linked every module to each other so that no item or cash can be misplaced/stolen by the management.
Plausible is based off another application of ours, which had the above mentioned issues and the core of that software was Codeigniter 2, which is now outdated as the applications initial release was in 2014. We decided to rebuild it with a modern UI and a better framework. Current version of Codeigniter4 follows the same file, folder structure as Laravel, so we thought it is best to dump our core altogether and migrate to Laravel. So, we built it upon the Laravel Framework, with PHP version support for ¿=7.3 with a MySQL database. For front-end UI design we used a premium template (NobleUI) to minimize development time. Some minor Database changes were made from the previous application to comply with the fix for above mentioned security issues, but the most notable difference between the two applications is the report section of Plausible.

5.2 System Analysis

System Analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

5.2.1 Six Element Analysis

Process	System Roles				
	Human	**Computing Hardware**	**Software**	**Database**	**Comm. & Network**
Manage Stock	Manager	Desktop/Laptop	Chrome/Firefox	MySQL	Apache, WAN
Manage Sales	Biller	Desktop/Laptop	Chrome/Firefox	MySQL	Apache, WAN
Manage Return	Biller	Desktop/Laptop	Chrome/Firefox	MySQL	Apache, WAN
Manage Employees	Owner	Desktop/Laptop	Chrome/Firefox	MySQL	Apache, WAN
View Analytic Reports	Owner	Desktop/Laptop	Chrome/Firefox	MySQL	Apache, WAN

Table 5.1: Six Element Analysis

5.2.2 Feasibility Analysis

Feasibility Study is a study to evaluate feasibility of a proposed project or system.

Feasibility	Planning	Implementation	Evaluation
Technical	Use a modern framework	Plausible is built using Laravel, Vue.Js & MySQL. These are the technologies that are very popular in the modern software industry and the three developers assigned for this project are well familiar with all the technologies	Project is Technically Feasible.
Operational	Make User Friendly	Plausible is a web application made with complex logic and technology but for any end user it is quite self-explanatory. The UI of the project was designed with usability in mind. A lot of difficult choices were made to make it user-friendly even though they made the development complicated	Determined as Operationally Feasible
Economic	Cut down operations costs	The software is designed to depreciate any kind of manual paper activity and fully automate the business. The cost of developing this software in terms of other services is the domain and hosting. Compared to long term the money will be insignificant to the service it will be providing	Project is Economically Feasible

Table 5.2: Feasibility Analysis

5.2.3 Problem Solution Analysis

The existing system for management in small businesses is heavily backdated and unreliable, leading to missing products and mismanagement. This is 2021 and there has been advances in technology which allows businesses to automate their processes. The problems our software will solve is listed below:

- Product out of stock and expiry date
- Sale, Purchase, Sale Return, Purchase Return history
- Customer, Supplier profiles
- Accounts history by date and duration

- Customer, Supplier Balance status (Due, Clear etc)
- Net Profit Loss at a glance

5.3 System Design

5.3.1 Rich Picture

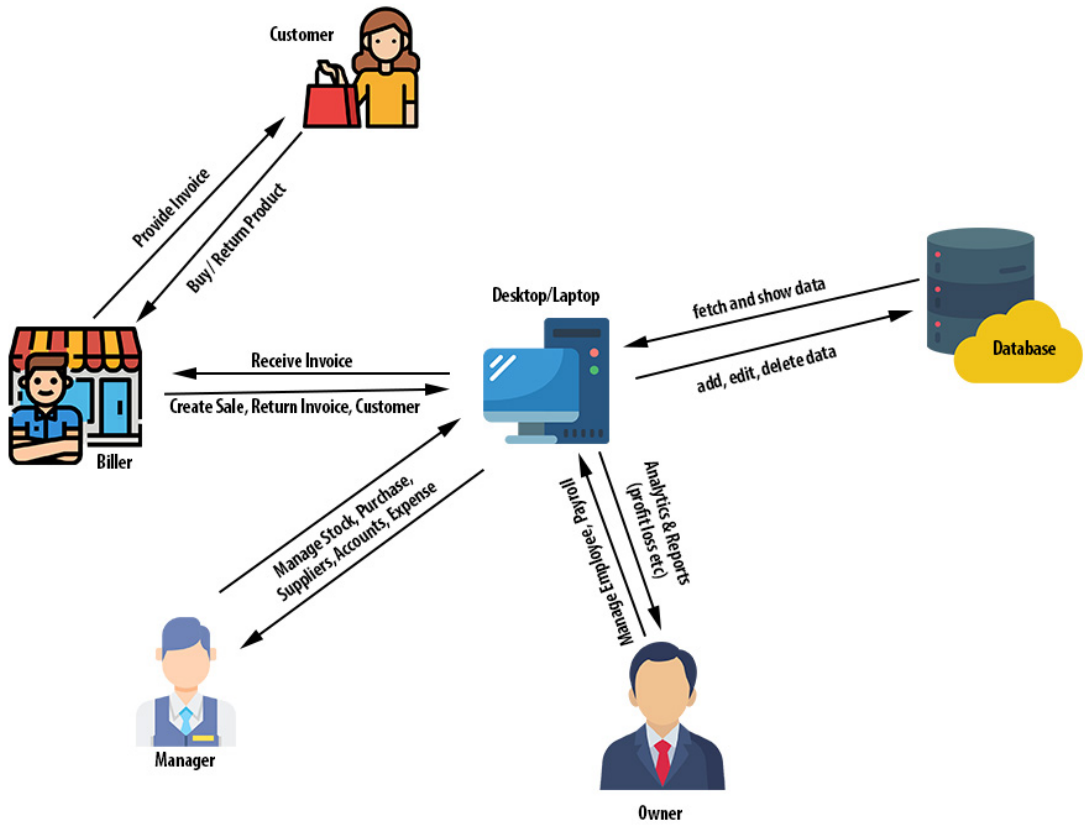


Figure 5.1: Plausible Rich Picture

5.3.2 Software Architecture

Laravel follows the MVC architecture pattern. MVC is not linear, therefore there is no tier system here[5]. MVC architecture is triangular: the view sends updates to the controller, the controller updates the model, and the view gets updated directly from the model.

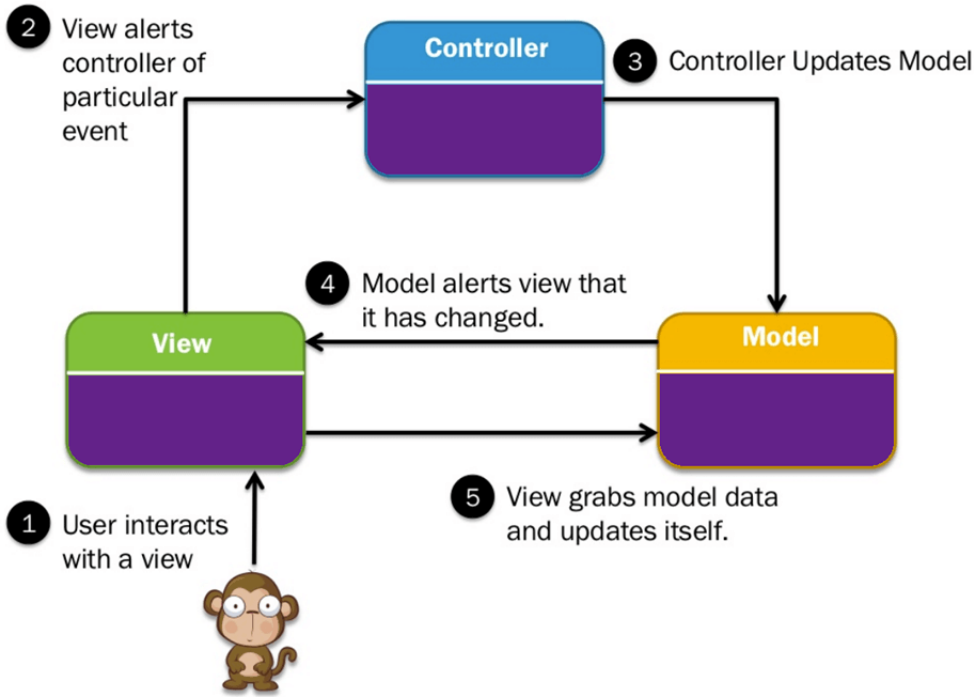


Figure 5.2: MVC Architecture

5.3.3 Database Diagrams

Figure 5.3: Database Diagram Full Image

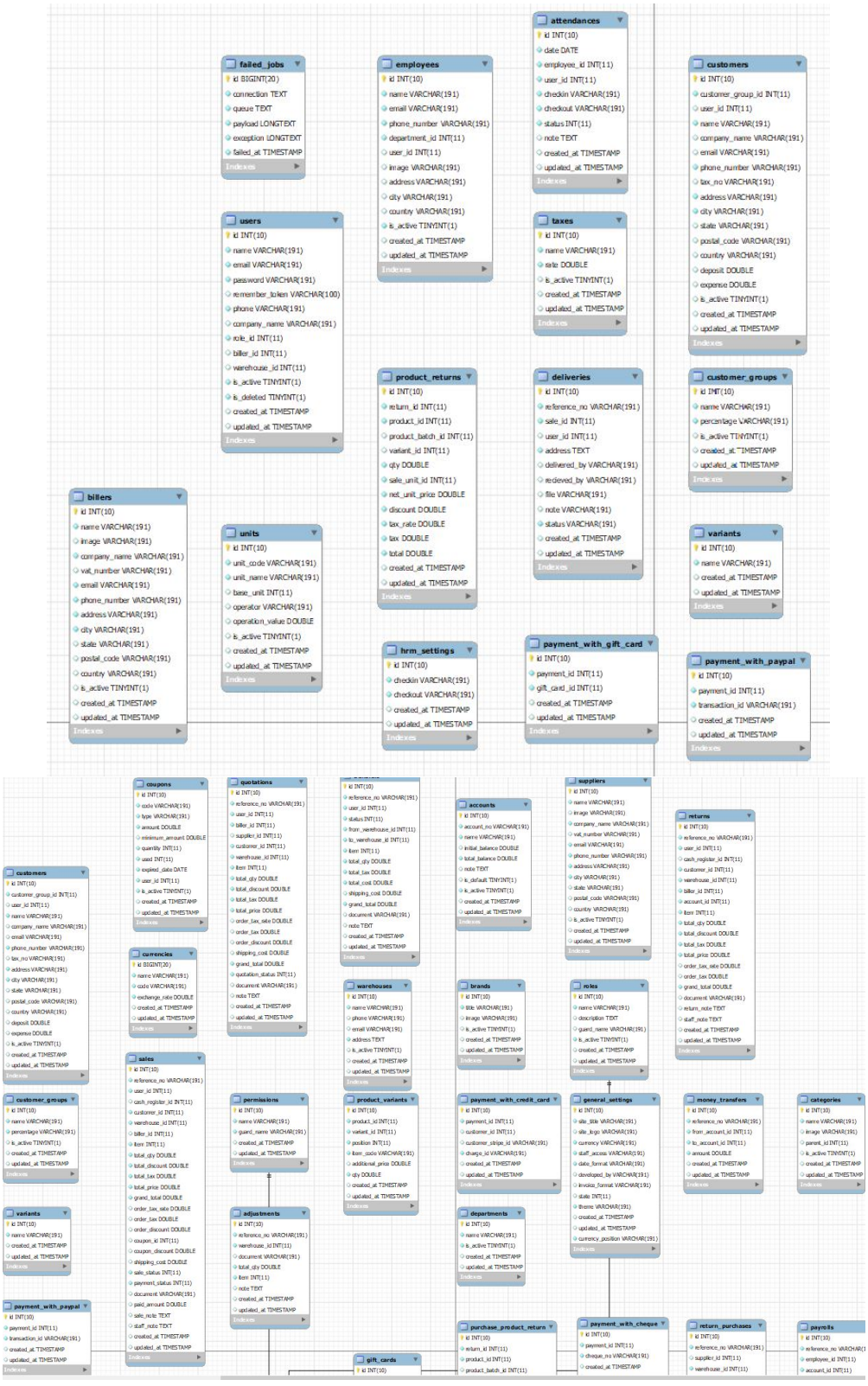


Figure 5.4: Database Diagram Cropped

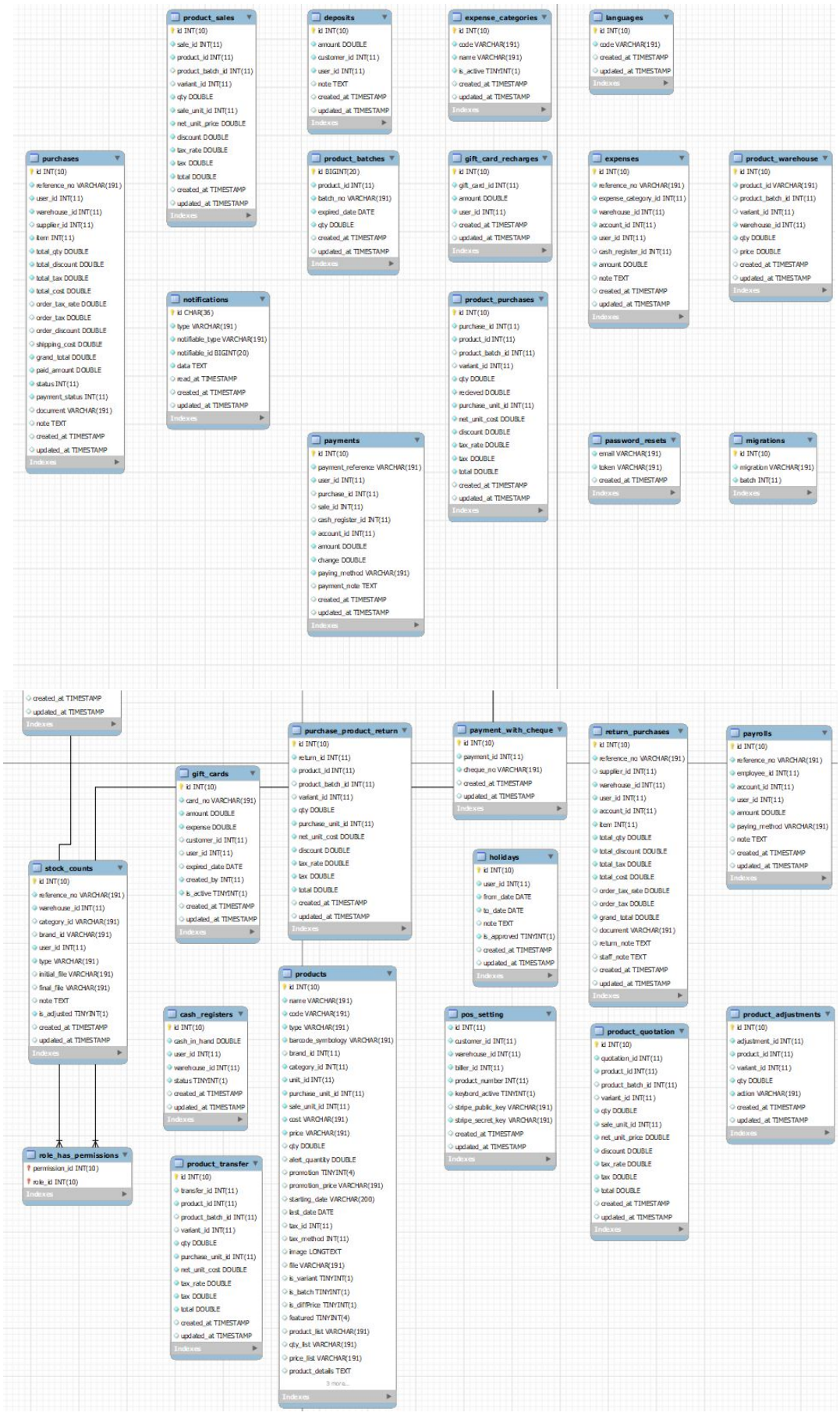


Figure 5.5: Database Diagram Cropped

5.3.4 Functional and Non-Functional Requirements

Functional Requirements: A functional requirement is a function or feature that must be included in an information system in order to satisfy the business need and be acceptable to the users. A functional requirement defines what an application and its components are and what these components are supposed to accomplish. The following functional requirements were gathered with our decided requirements gathering methods. The inputs, processes and output are discussed below[6]:

Non-Functional Requirements: Another type of requirement is non-functional requirements. A nonfunctional requirement is a description of the features, characteristics, and attributes of the system as well as any constraints that may limit the boundaries of the proposed solution. Non-functional requirements are briefly described below:

- **Performance:** represents the performance of the system which is required to exhibit and to meet the needs of users. Performance describes the acceptable throughput rate and acceptable response time. This application should provide a smooth experience for the Billers, Managers and also should have no lag in displaying the real-time as long as the devices are connected to a stable internet.
- **Security and Control:** Security and administrations are always a concern for any system but it is more sensitive in this project since this system will be dealing with monetary data. All information on the server side and client side is secured. Only the application administrators and developers have access to core code of the application to be able to directly manipulate any sort of information. Control requirements represent the environment in which the system must operate, as well as the type and degree of security that must be provided. Access to the system or information must be controlled with the privacy requirements.
- **Efficiency:** represents the system's ability to produce outputs with minimal waste. We have tried to eliminate duplicate steps in the processes and to use the resources in an efficient way. Keeping our code non repetitive by using reusable code and components is how we achieved efficiency

5.4 Product Features

1. Dashboard
   (a) Overview yearly Sales Chart
   (b) 5 Latest Sales
   (c) Top 5 best selling products monthly yearly

2. Products
    (a) Standard Product, Digital Product
    (b) Product Category
    (c) Add, edit, delete view products
3. Sales
    (a) Add, edit, delete view sales
    (b) Payments (Multiple add, edit, delete)
    (c) Return Sales
    (d) Add, edit, delete view delivery
4. Purchases
    (a) Add, edit, delete view purchases
    (b) Payments (Multiple add, edit, delete)
5. Transfers
    (a) Transfer products to warehouses
    (b) Add, edit, delete view transfers with status
6. People
    (a) Add, edit, delete view users, assign roles
    (b) Add, edit, delete view customers, billers, suppliers
7. Reports
    (a) Overview warehouse stock charts
    (b) Product quantity expiry alerts
    (c) Daily, monthly custom sales, payments, profit/loss purchase report
    (d) Customers, supplier, due, best-seller reports
8. Settings
    (a) Edit user profile, change password
    (b) Create roles for users
    (c) Add, edit, delete view customer groups
    (d) Add, edit, delete view warehouses

(e) Add, edit, delete view Tax rates

(f) Add, edit, delete view brands

(g) Add, edit, delete view units

5.4.1 Product UI

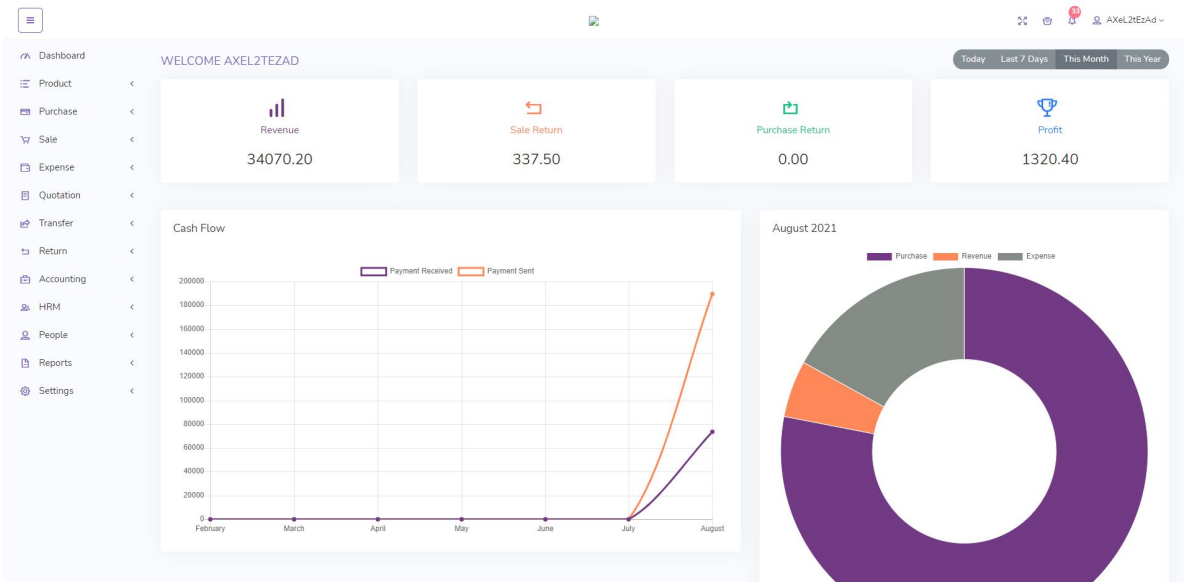


Figure 5.6: Dashboard UI

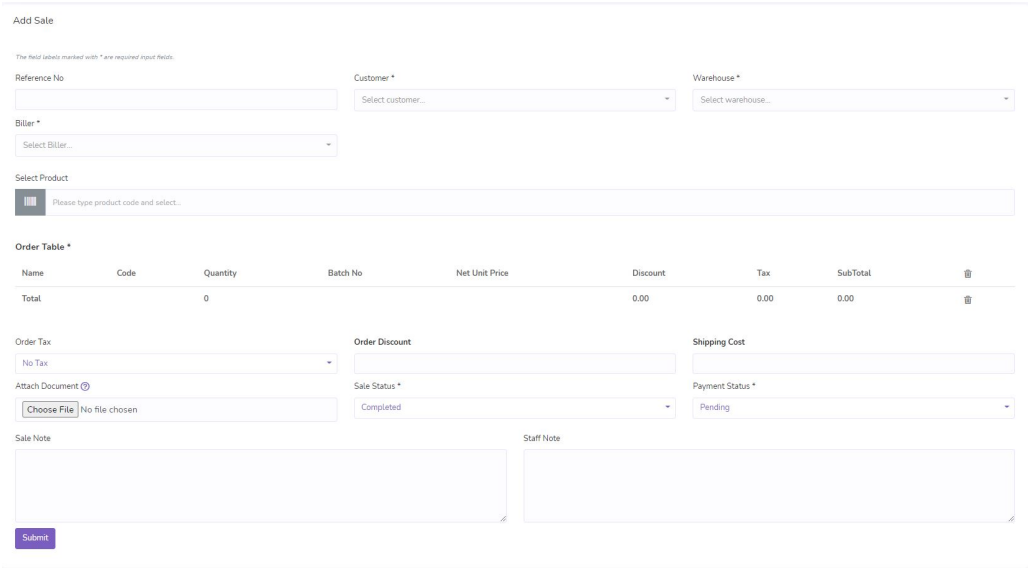


Figure 5.7: Create Sale

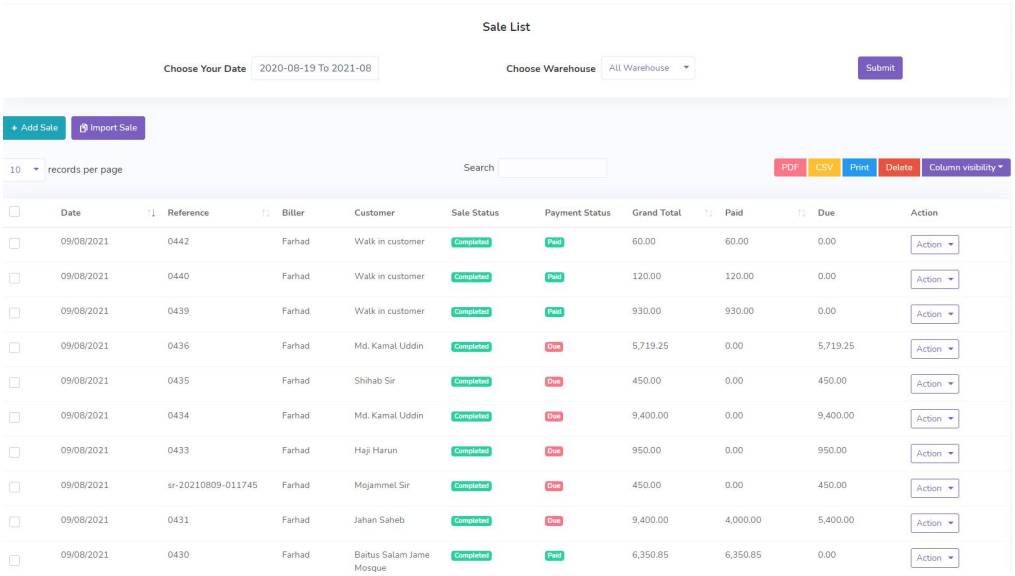


Figure 5.8: Sale List

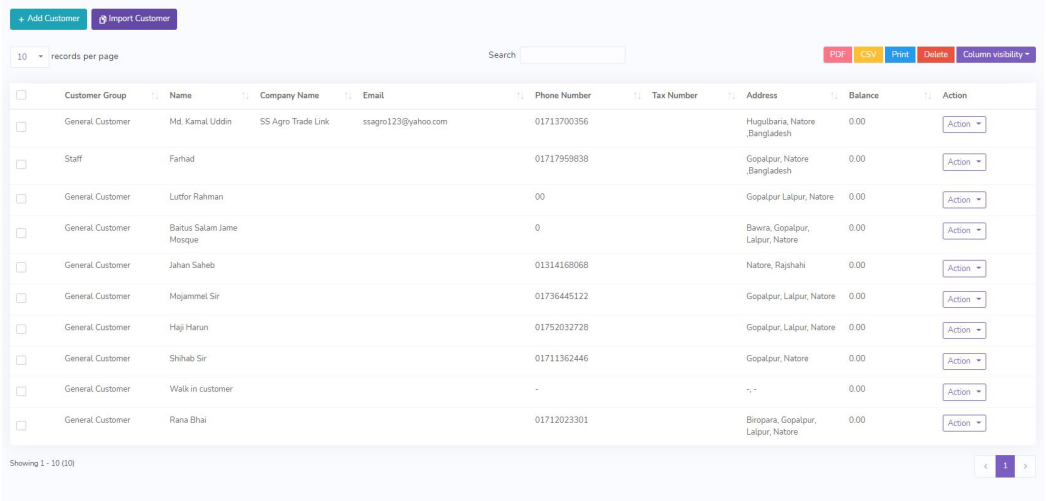
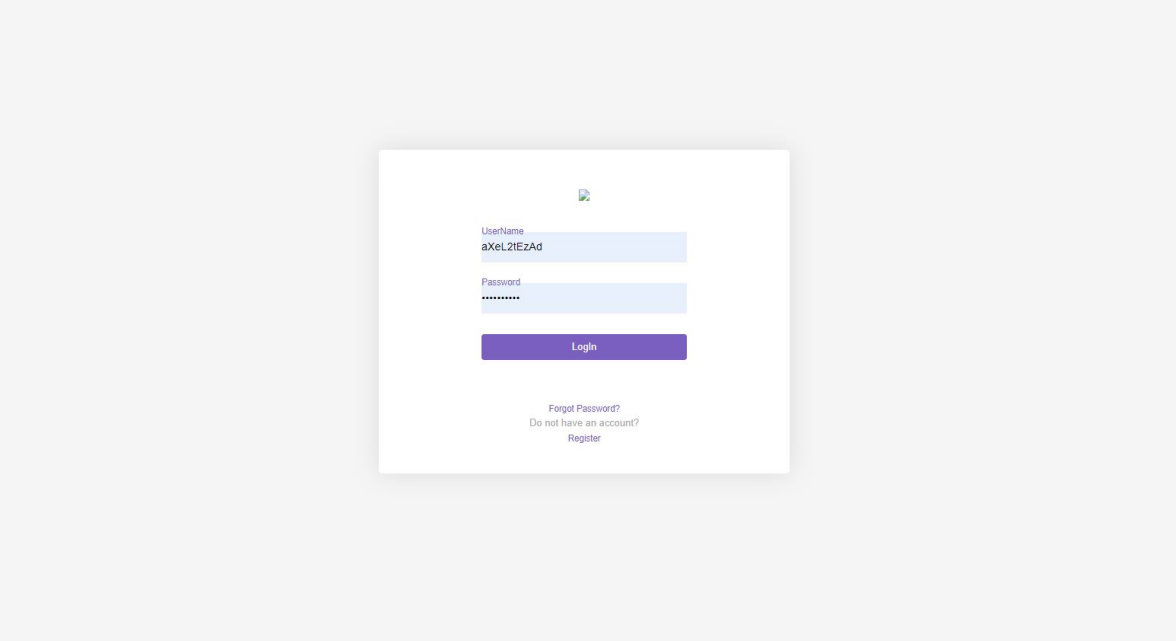


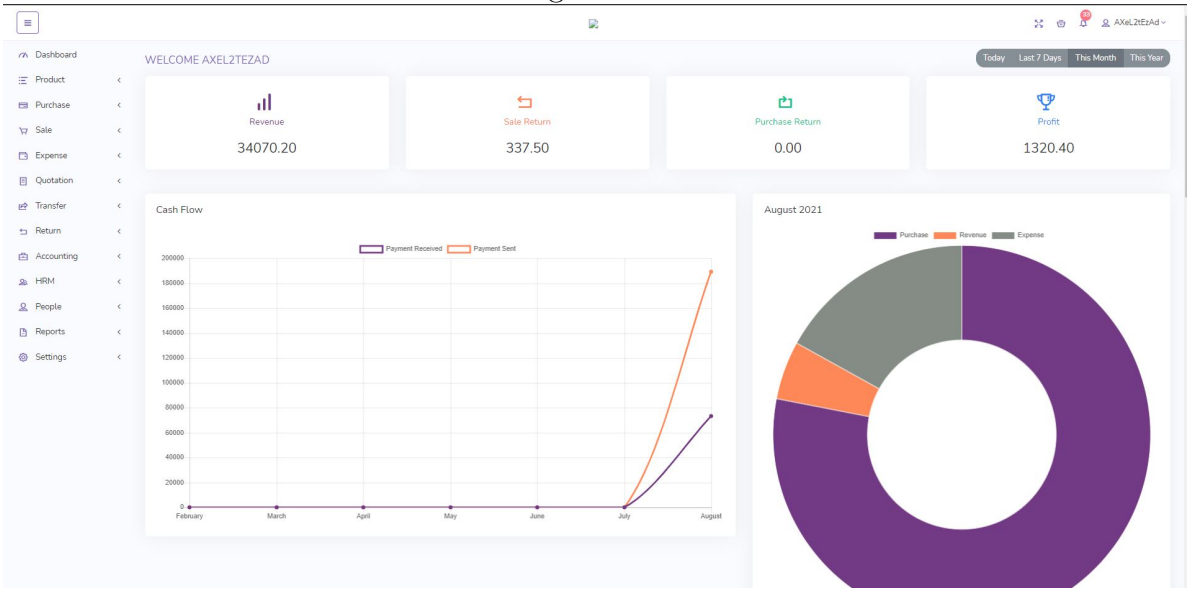
Figure 5.9: Customer List

Chapter 6

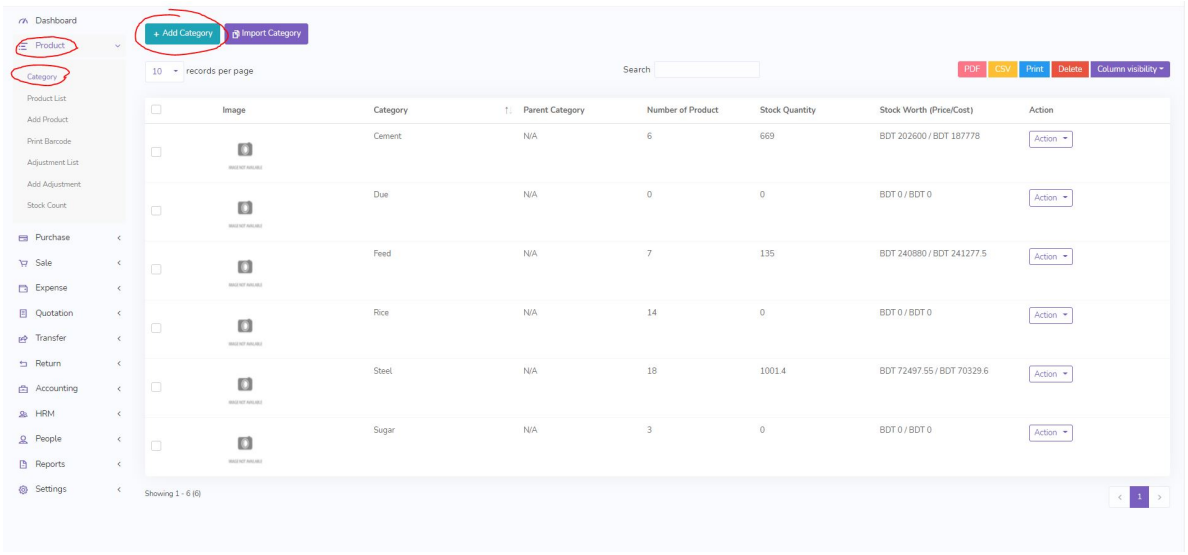
Results & Analysis



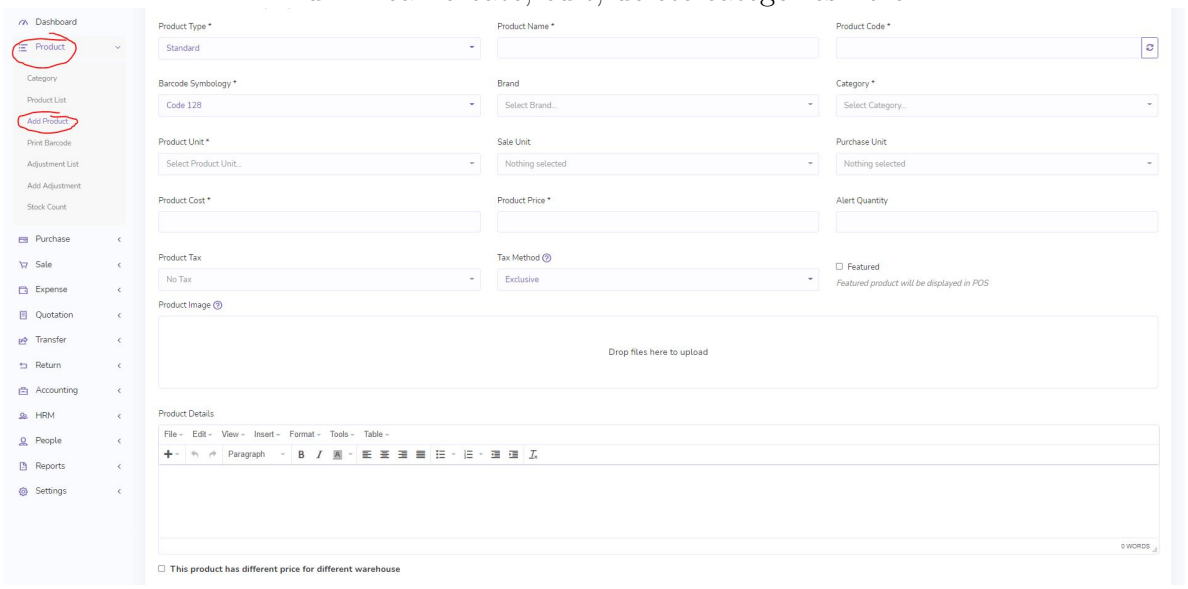
Login Panel



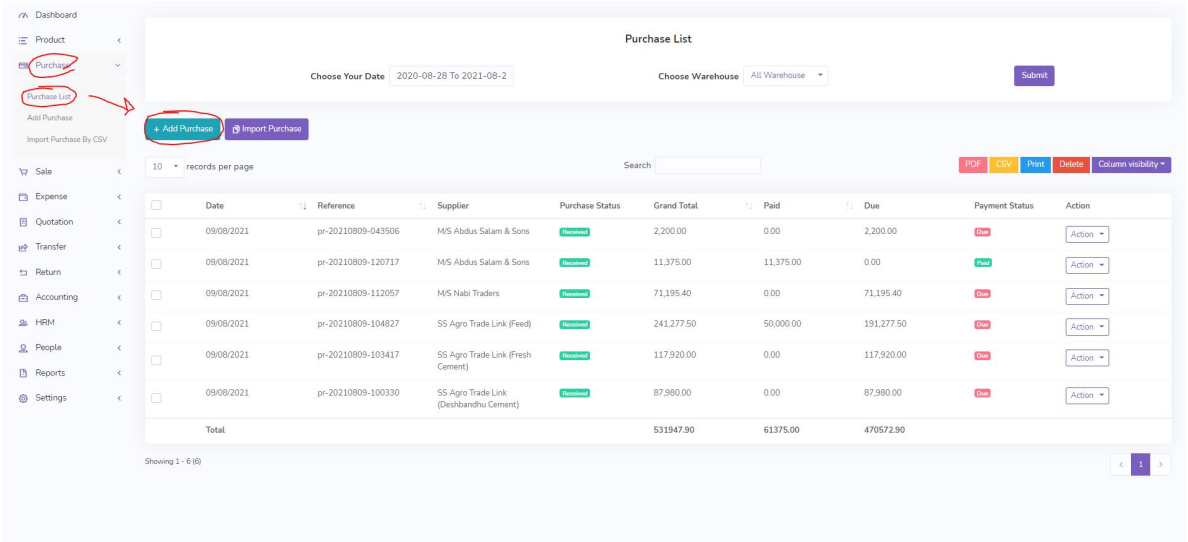
After Successful Auth, Admin is taken to dashboard



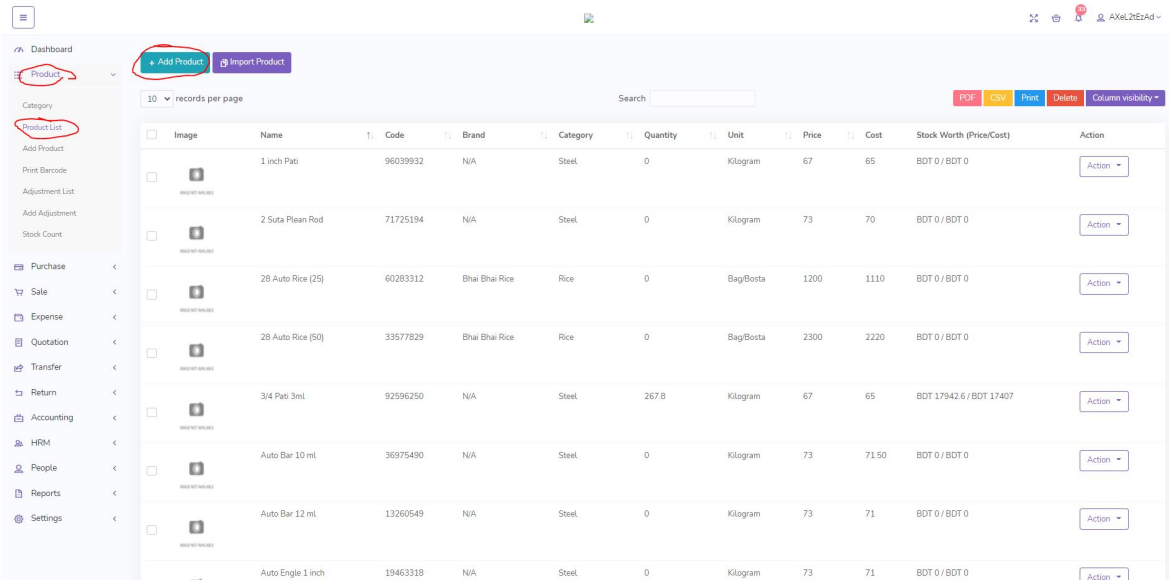
Admin can create, edit, delete categories here



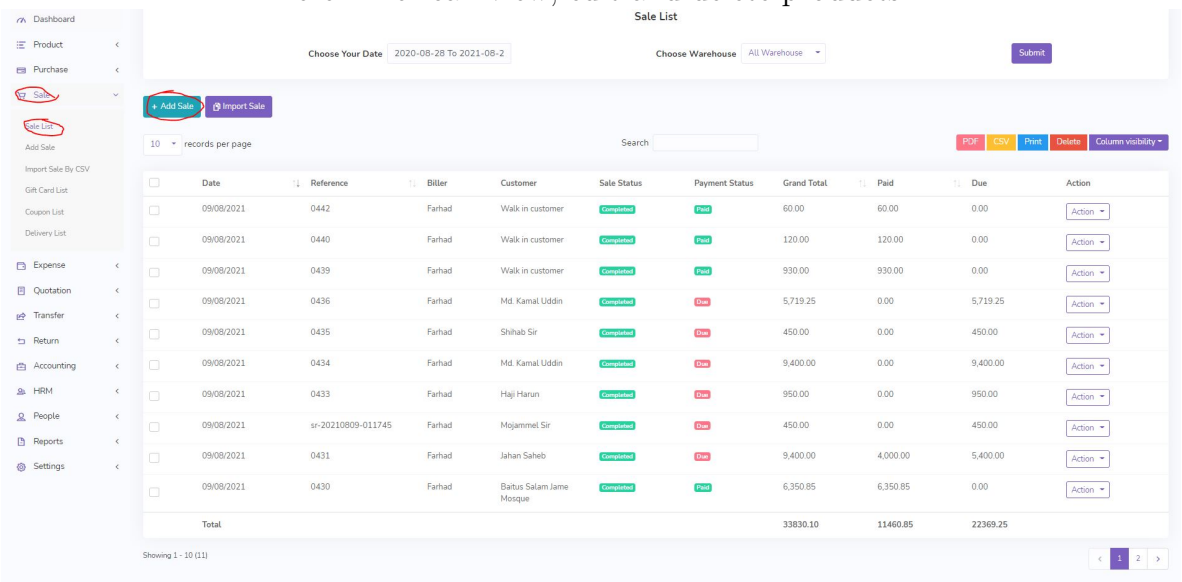
Admin can add product but this is only a placeholder, product stock is added upon being purchased from supplier.



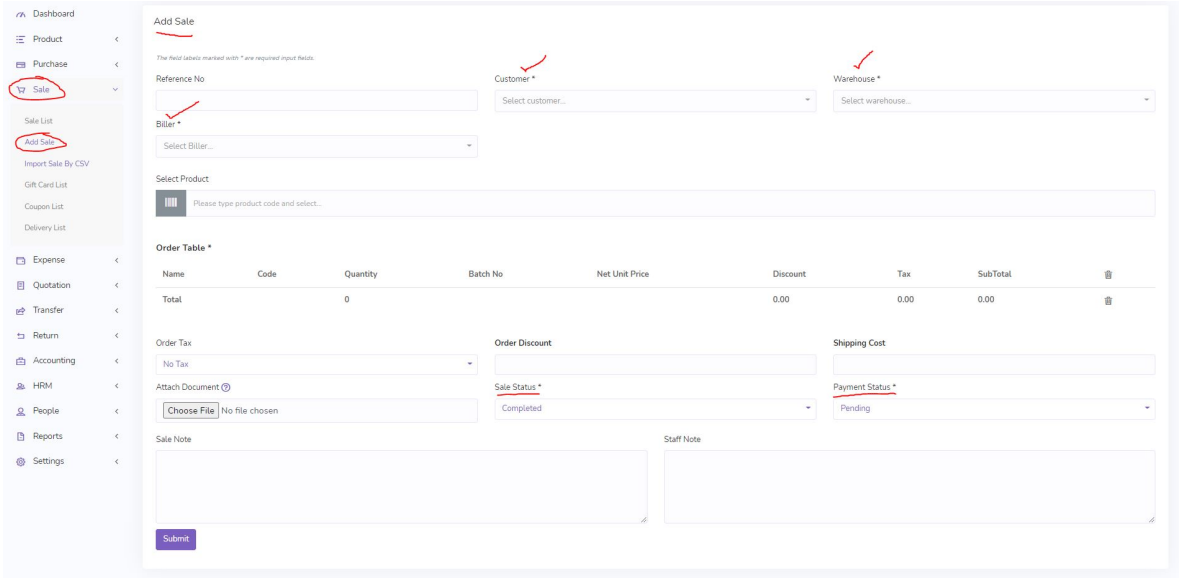
Purchase list. Admin can add, edit and delete purchases



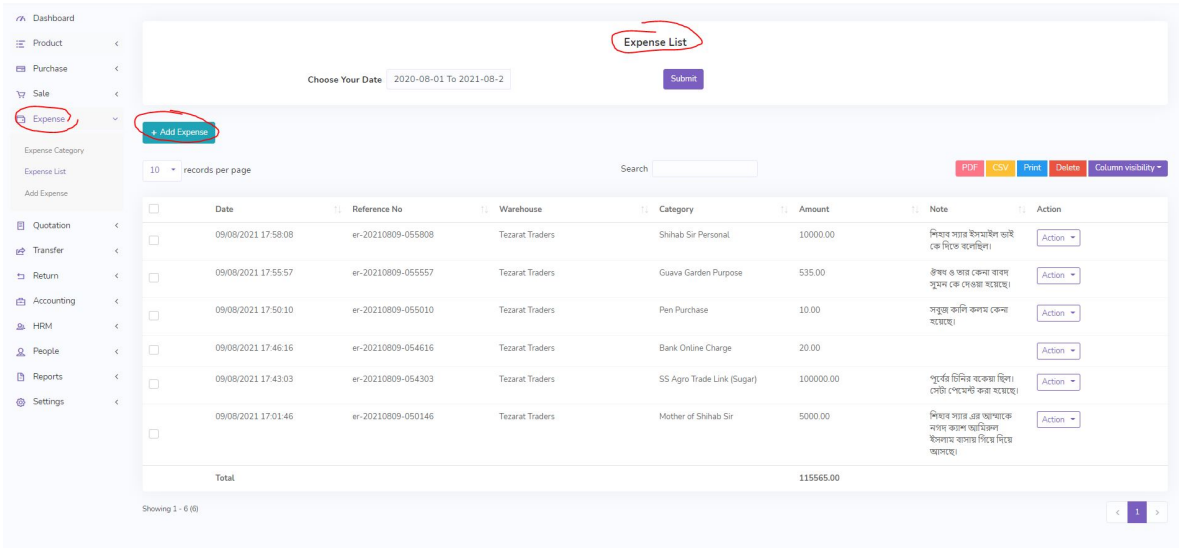
Here Biller can view, edit and delete products.



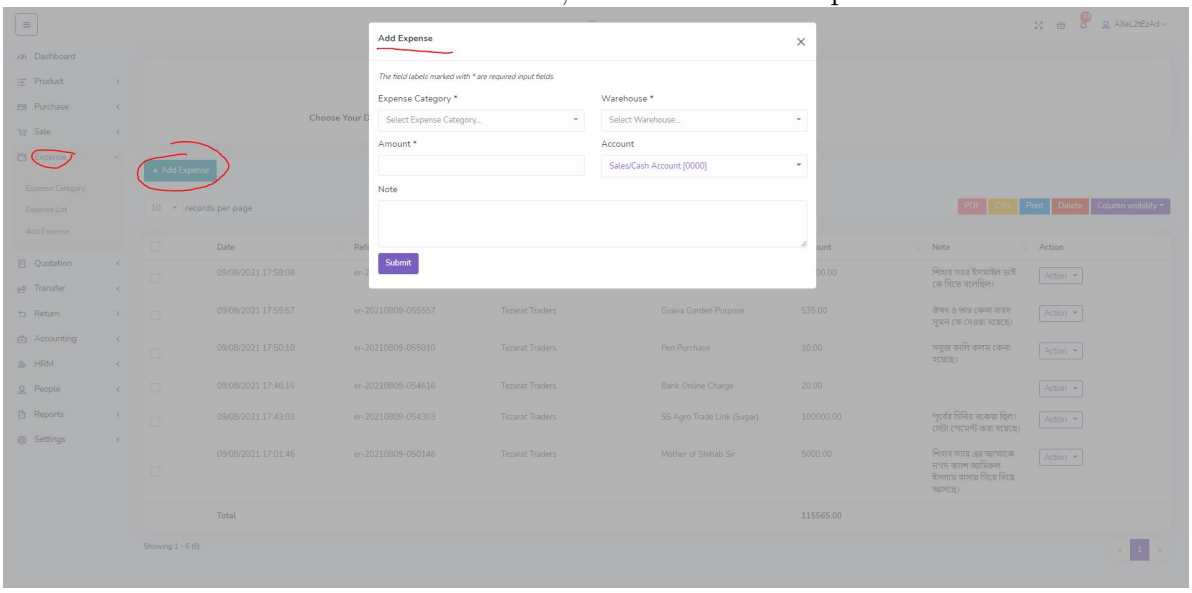
Biller can view, edit, delete, modify purchase status here.



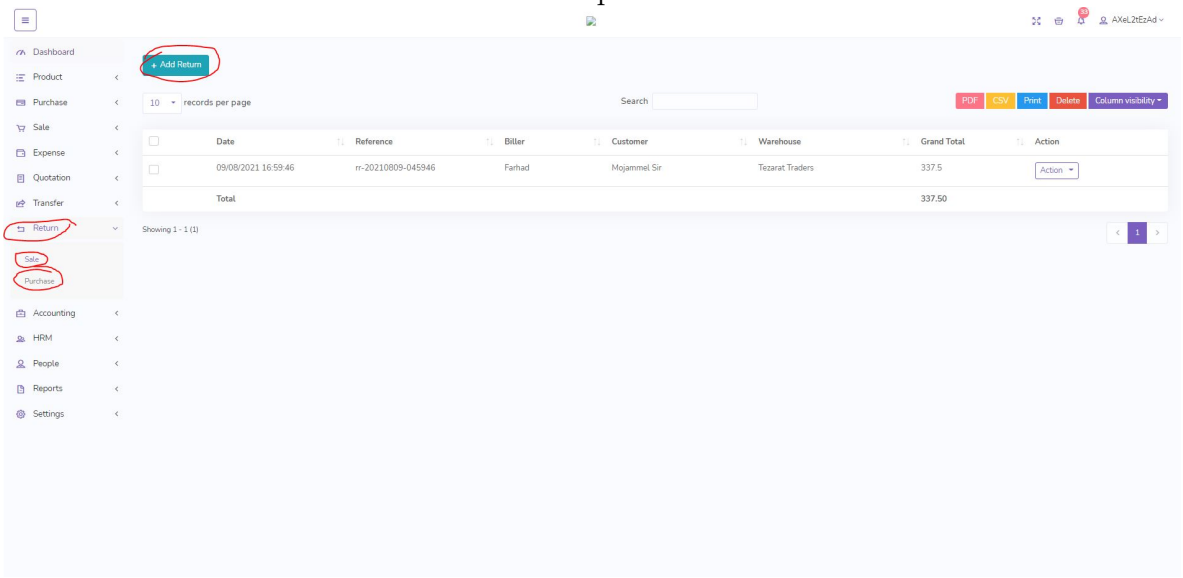
Biller can add sales (POS)



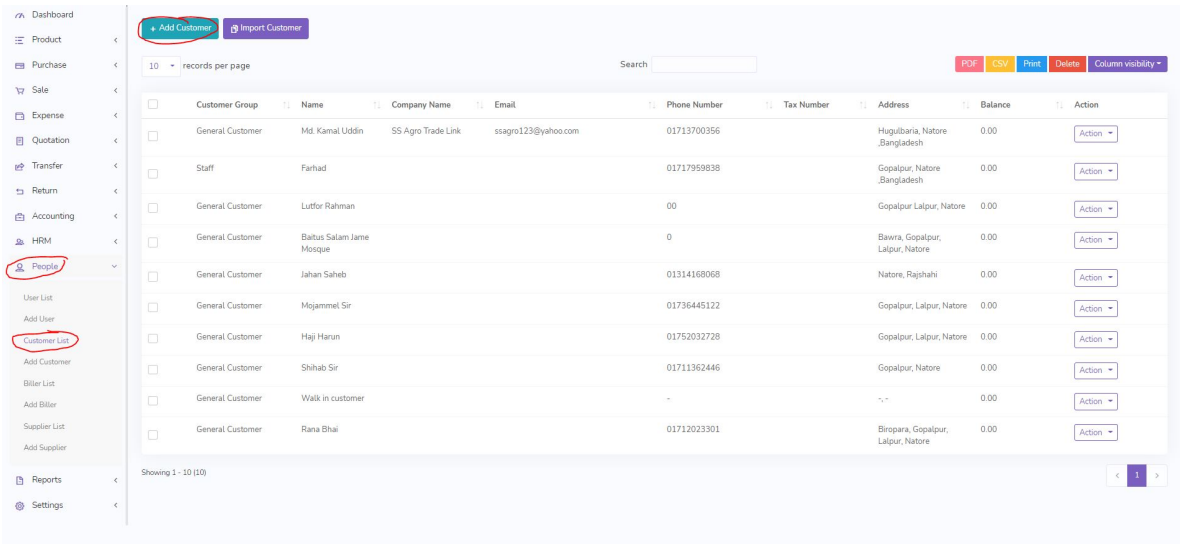
Here Admin can view, edit and delete Expenses.



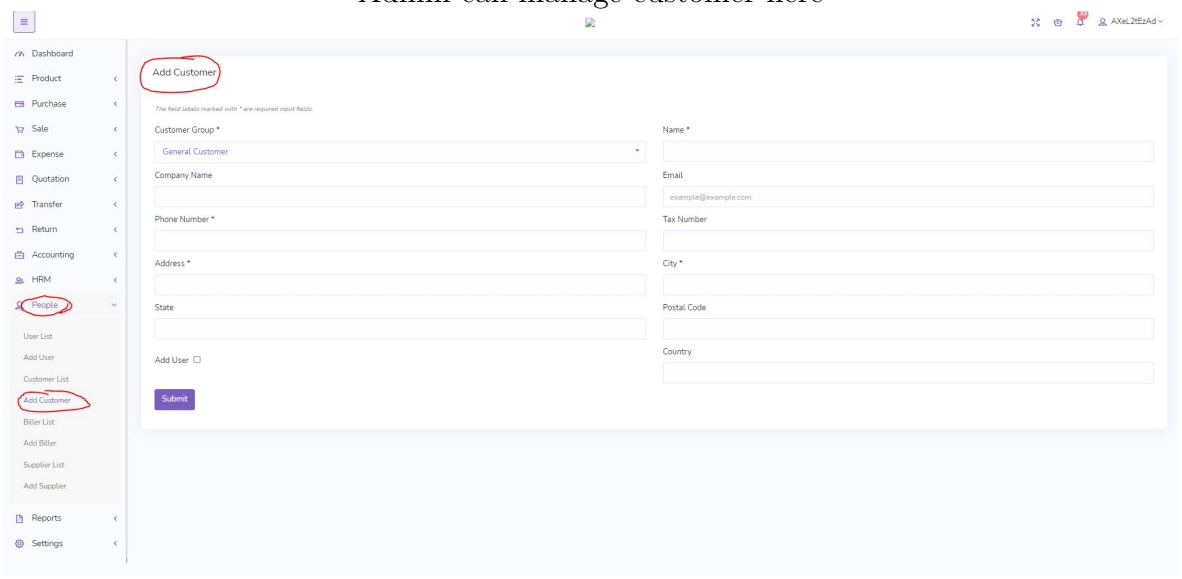
Add Expenses.



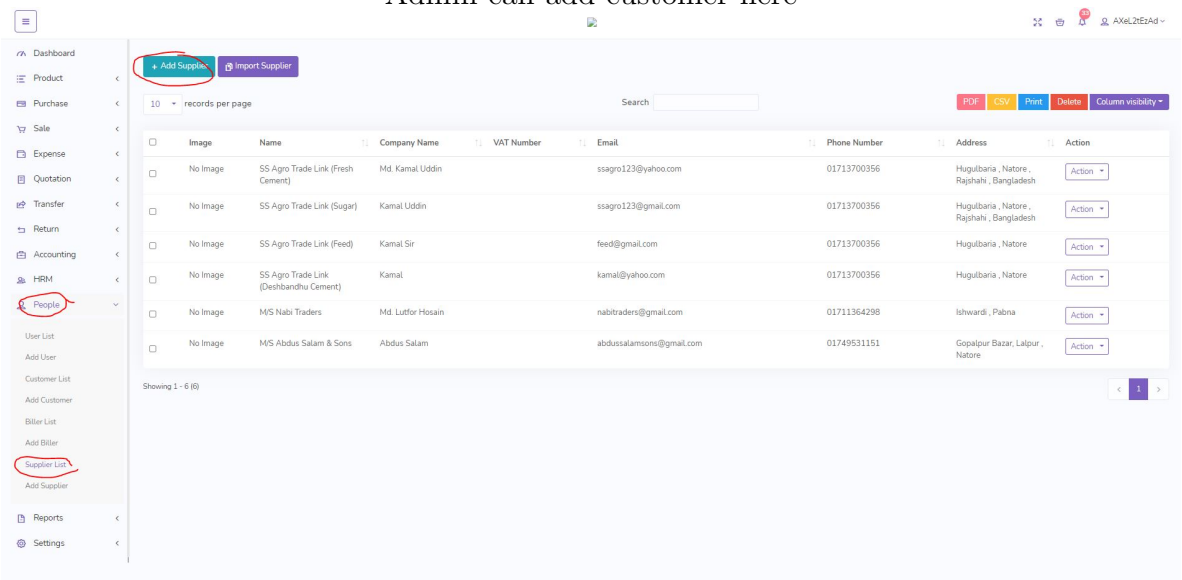
Biller can process sale return and purchase return here



Admin can manage customer here



Admin can add customer here



Admin can manage supplier here

Chapter 7

Project as Engineering Problem Analysis

7.1 Sustainability of the Project/Work

Sustainability of the product refers to its ability to be maintained and updated. In the modern world, every application being released needs to be maintained and continuously updated for its user base.

A product can be sustainable in three main categories:

- Community Sustainability : After official release of Plausible Software, it is believed that it will have a strong user base since the target audience for the application is specific. We can expect the users to refer to other Shops regarding the application and thus growing the user base. With a growing user base it will also grow a community and hence it can be said that it is Sustainable in terms of Community
- Financial Sustainability: This refers to how the application's running cost will be maintained after it has been released and whether it will generate enough revenue as acceptable profit. An application's running cost includes - server cost, database storage cost etc. The initial release of Plausible will have a maintenance charge which will generate profit.
- Organizational Sustainability: It relates to how the organization will continue to operate after the release of the application. After the release of an application, usually the organization maintains the application via its current team, Plausible has few more features planned for the future to be worked on and released. In conclusion, it can be said that the project is Organizationally Sustainable.

7.2 Social and Environmental Effects and Analysis

- Social Effect: Plausible aims to get more Small businesses interested in an efficient and data oriented management process which can generate more revenue for them, and in term contributing to the economic growth.
- Environmental effect: Plausible aims to automate management process by digitizing everything and therefore getting rid of paper use. No paper use means less trees being cut and more CO2 absorbed from the atmosphere leading to a better, safer environment.

7.3 Addressing Ethics and Ethical Issues

There are unspoken rules and ethics guidelines that need to be followed when working on creating and releasing an application. The developers of the Plausible believe that the application does not breach any code of conduct of application release and development since they all have been taken into serious concern. Some of them are:

- Clear Promotion: Plausible only intends to promote the company that created it and itself. Other than what has been mentioned, the it has no intention of promoting anything or anybody else.
- No Discrimination or Favoritism: Plausible does not discriminate of any kind based on race, sexuality, gender, religious beliefs, color, language, political or other opinion, national or social origin, property, birth, or other status.
- Data Storage Security: Only the lead developer and the owner has access to the server and the database. Since they are hosted in the cloud and can only be accessed via lead developer's and the owner's login credentials; the data stored can be deemed as safe and secure.

Chapter 8

Lesson Learned

8.1 Problems Faced During this Period

During my internship program, I have faced lots of challenges while working on this Project. The main ones are:

- Adapting to Technologies: Since my freelancing days, I have mostly used JavaScript based frameworks and worked on front end side only. I was not familiar with backend side of things and Laravel or MVC applications in general.
- Changing Frameworks: Since we switched from Codeigniter(older pos) to Laravel(SaaS Plausible), the file-folder structure, code writing convention was completely different even though both were PHP frameworks.
- PHP Version Compatibility's: Unlike JavaScript, PHP itself is not backwards compatible with their older versions, causing frameworks to have certain compatible PHP versions and there are quite a few releases in the past 5 years, all of them containing breaking changes.
- Identifying and Fixing Bugs: often, there were very visible bugs and even though the fix was applied it would still not fix.

8.2 Solution of those Problems

As I learned everything from scratch with the help of my mentor+supervisor, I found that MVC applications followed a very simple and easy pattern of having the front-end and back-end in the same package which was very convenient to develop rapidly and provide a solution much quicker. As I have experience from freelancing, I am able to read code of object oriented notations which helped me immensely on overcoming this issue. I only learned the laravel framework and applied the algorithms accordingly, with

some help from my supervisor. For maximum compatibility, an older version of laravel was used to that it would run on older and newer Servers. The bugs were mostly caused from PHP version compatibility with the plugins used. Different plugins had different requirements so for some we had to use older version of the plugins.

Chapter 9

Future Work & Conclusion

9.1 Future Works

The rapid development and scalibility of Software as a Service (Saas) Applications is taking over the world by a storm. They are sometimes called Web-based/on-demand/hosted software. Whatever the name, The application runs on the providers servers. The provider manages access to the application, including security, availability, and performance[1].

I believe Plausible 2.0 will come with further security measures and more affordable server costs. Plausible is a refined version of a 4 year older software from our services, But as of now, It is a reliable SaaS application for small business owners providing them with all they will ever need.

9.2 Conclusion

Although as a developing country, Bangladesh is seeing an increase of demand for SaaS applications as it is a booming hub for small businesses. More and more owners are interesting in automating their businesses which increases efficiency by a wide margin, cuts down labour and management costs, reduces manual work and generates more revenue making it a sustainable choice. The cost of the software outweighs the service it provides. That is the goal of SaaS and it will dominate the software industry moving forward.

Bibliography

[1] M. Godse and S. Mulik, "An approach for selecting software-as-a-service (saas) product," in *2009 IEEE International Conference on Cloud Computing*, pp. 155–158, IEEE, 2009.

[2] M. Cusumano, "Cloud computing and saas as new computing platforms," *Communications of the ACM*, vol. 53, no. 4, pp. 27–29, 2010.

[3] N. S. Engineering, "The Ultimate Guide to Rapid Application Development," 2020.

[4] N. M. N. Daud, N. A. A. A. Bakar, and H. M. Rusli, "Implementing rapid application development (rad) methodology in developing practical training application system," in *2010 International Symposium on Information Technology*, vol. 3, pp. 1664–1667, IEEE, 2010.

[5] A. Leff and J. T. Rayfield, "Web-application development using the model/view/controller design pattern," in *Proceedings fifth ieee international enterprise distributed object computing conference*, pp. 118–127, IEEE, 2001.

[6] Y. G. Fang-chun and L. G. Xiang-ming, "An overview on software non-functional properties research," *Journal of beijing university of posts and telecommunications*, vol. 27, no. 3, p. 1, 2004.