Premier University

# Department of Computer Science & Engineering



**Final Year Project Report**

**On**

**Track My Subscription**

# Submitted by

Mohammed Arafat Ali

ID: 1603110201187

Abdul Karim

ID: 1603110201188

Suchitra Das

ID: 1603110201209

***In partial fulfillment of the degree of***

Bachelor of Science in Computer Science & Engineering

# Under the Guidance of

Mr. Anik Sen

Assistant Professor

Department of Computer Science & Engineering

Premier University, Chittagong

April 2022

**Declaration**

In partial fulfillment of the requirements for the degree of Bachelor of Science, the project report is submitted to Department of Computer Science and Engineering, Premier University. We hereby declare that the work provided in this paper was completed under the supervision of Mr. Anik Sen, Assistant Professor, Department of Computer Science & Engineering, Premier University and that materials from other researchers' work are mentioned as references. This project report has never been submitted before, in whole or in part, for any degree.

Mohammed Arafat Ali

ID: 1603110201187

Abdul Karim

ID : 1603110201188

Suchitra Das

ID : 1603110201209

## 

## CERTIFICATE OF APPROVAL

This project title “Track My Subscription” submitted by Mohammad Arafat Ali (ID: 1603110201187), Abdul Karim (ID: 1603110201188) and Suchitra Das (ID: 1603110201209)

has been accepted as satisfactory in fulfilment of the requirement for the degree of Bachelor of Science in Computer Science & Engineering (CSE) as B.Sc. Engineering to be awarded by Premier University, Chittagong.

**………………………….**

## Prof. Dr. Taufique Sayeed

Chairman

Department of Computer Science and Engineering

Premier University Chittagong

**…………………………**

## Mr. Anik Sen

Assistant Professor

Department of Computer Science and Engineering

Premier University Chittagong

**ACKNOWLEDGEMENT**

First and foremost, when presenting this project, we express our heartfelt gratitude to Almighty for his grace and blessings, which enabled us to successfully finish this project work.

We'd like to take this occasion to thank our esteemed supervisor, Mr. Anik Sen Sir, Assistant Professor, Department of Computer Science & Engineering, Premier University, Chattogram. This project executed successfully under his supervision. His continual assistance and desire to help grasp this project and its manifestation in great depth of knowledge aided us in completing the assignment. Throughout the advancement of our work, his unwavering support, cooperation, inspiration, amazing direction, constructive criticism, and fantastic suggestions helped us make this work considerably better. We are cordially grateful to him and we really feel blessed to have him as our supervisor.

We are also grateful to Prof. Dr. Taufique Sayeed, Chairman of Premier University's Department of Computer Science and Engineering, for his unwavering encouragement and support. Finally, we'd want to express our gratitude to our parents, whose blessings and appreciations aided us in finally finishing this project. We'd also like to convey our gratitude and respect to our parents for their unwavering support, without which this work would not have been possible.

Last but not least, we'd like to express our gratitude to our parents. We appreciate our family's love, as well as their moral and financial support. We owe a debt of gratitude to our friends, classmates, seniors, juniors, and lab assistants, who have provided us with unwavering support and assistance in every way. We are extremely fortunate to have such a kind person as them by our side.

**ABSTRACT**

“Track My Subscriptions” is like having users own virtual assistant keeping watch on all users recurring payments of purchased subscriptions. Subscribers may use “Track My Subscription” to keep track of their subscription activity and receive notifications and reminders, as well as manage their payment methods, currency rates, and more. Automated reminders and notifications are sent by “Track My Subscription” so that users aren't billed for subscriptions users don't want. It helps users keep track of their monthly spending and see how much money they have remained for the rest of the month. Filtering by folder or payment type might also help consumers quickly find the information they are looking for. Therefore, we have built an interactive and effective website called “Track My Subscriptions”. ​

## 

**Table of Contents**

## Chapter 1 1

## Introduction 1

* 1. Introduction 1
  2. Objectives 1
  3. Motivation 1
  4. Summary 2

## Chapter 2 3

## Literature Review 3

* 1. Introduction 3
  2. Review of Existing Systems 3
  3. Comparison of Our System 4
  4. Necessity of Methodology 4
  5. Software Development Life Cycle (SDLC) 4
     1. Feasibility Study 5
     2. Requirement Gathering and Analysis 5
     3. System Specification 5
     4. System Design 5
     5. Program Design and Coding 5
     6. Testing 5
  6. Software Process Model 6
  7. Summary 6

## Chapter 3 7

## Software Requirement Specification 7

* 1. Introduction 7
  2. Requirement Engineering 7
     1. Requirement Analysis 7
     2. Requirement Specification 9
     3. Requirement Validation 10

|  |  |  |
| --- | --- | --- |
| 3.3 Specific Requirements |  | 11 |
| 3.3.1 User Interface |  | 11 |
| 3.3.2 Hardware Interface |  | 11 |
| 3.3.3 Software Interface |  | 11 |
| 3.4 Software Requirement Specification |  | 12 |
| 3.4.1 User Requirements |  | 12 |
| 3.4.2 System Requirements |  | 12 |
| 3.4.3 Software Requirements |  | 12 |
| 3.5 Summary |  | 12 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Chapter 4 13** | | |  |
|  | **Normalization and Data Dictionary 13** | | |  |
|  | 4.1 Introduction 13 | | |  |
|  | 4.2 Database 13 | | |  |
|  | 4.3 Database Management System (DBMS) 13 | | |  |
|  | 4.4 Database Normalization 13 | | |  |
|  | 4.5 Normalization Rule 13 | | |  |
|  | 4.5.1 First Normal Form (1NF) 14 | | |  |
|  | 4.5.2 Second Normal Form (2NF) 14 | | |  |
|  | 4.5.3 Third Normal Form (3NF) 14 | | |  |
|  | 4.5.4 Boyce-Codd Normal Form (BCNF) 15 | | |  |
|  | 4.5.5 Fourth Normal Form (4NF) 15 | | |  |
|  | 4.6 Data Dictionary 15 | | |  |
|  | 4.7 Summary 26 | | |  |
|  |  | | |  |  |
|  | **Chapter 5 27** | | |  |
|  | **Modeling 27** | | |  |
|  | 5.1 Introduction 27 | | |  |
|  | 5.2 Use Case Diagram 27 | | |  |
|  | 5.3 Activity Diagram 29 | | |  |
|  | 5.3.1 Activity Diagrams for User 29 | | |  |
|  | 5.3.2 Activity Diagrams of Admin 32 | | |  |
|  | 5.4 ER Diagram 36  5.5 Summary 36  **Chapter 6 37**  **Software Implementation 37**  6.1 Introduction 37  6.2 Software Implementation 37  6.3 Web Server 39  6.4 Text Editor 40  6.5 Software 40  6.6 Snapshots 41  6.6.1 Product Add 41  6.6.2 Add New Deal Type 41  6.6.3 Subscription Add 42  6.6.4 Add Subscription by User 43  6.6.5 Deal Platform 44  6.6.6 Deal Type 45 | | |  |
|  | 6.6.7 Product Category 45  6.6.8 Add New Product Founder 46  6.6.9 User Register 47  6.6.10 User Log In 47  6.6.11 User Dashboard 48  6.6.12 All Subscription In User Dashboard 49  6.6.13 Membership Level 49  6.6.14 User Subscription Add 50  6.6.15 User Product Request 51  6.6.16 User Dashboard After Purchase 51  6.6.17 Purchase Subscription Information 52  6.6.18 Subscription Alert 53  6.6.19 Subscription Bulk Add 54  6.7 Summary 55  **Chapter 7 56**  **Discussion & Conclusion 56**  7.1 Introduction 56  7.2 Discussion 56  7.3 Limitation of The System 57  7.4 Future Work 57  7.5 Summary 57  **Reference 58** | | |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  | | |  |  |
|  |  |  |
|  |  |  |

**Table of Figures**

Figure 1: Software Development Life Cycle 4

Figure 2: Requirement Analysis Process Model 8

Figure 3: Requirement Specification Modeling 9

Figure 4: Non-functional Requirement 10

Figure 5: Use Case Diagram 28

Figure 7: Activity Diagram of Profile 29

Figure 8: Activity Diagram of Subscriptions 30

Figure 9: Activity Diagram of Accomplishment 30

Figure 10: Activity Diagram of Search 31

Figure 11: Activity Diagram of Request 31

Figure 12: Activity Diagram of All Users 32

Figure 13: Activity Diagram of Membership Request for Admin 33

Figure 14: Activity Diagram of Admin Product View 33

Figure 15: Activity Diagram of All Categories 34

Figure 16: Activity Diagram of Membership Request for User 35

Figure 17: Activity Diagram of Subscription for Admin And User 35

Figure 18 : ERD 36

# Introduction

**Chapter 1 Introduction**

The project “Track My Subscription” is a subscription tracking tool offering notifications, reminders, and payment type management. It gives automated reminders and cautions to avoid unwanted subscriptions. This project is to add the subscriptions user has purchased. It also provides a search feature where users can search subscriptions and also see a list of deals. In this project admin can add membership for users and requesting admin for adding deals. This system is providing facilities to change ownership of subscriptions. It'll send users a reminder exactly when user want it. So users can cancel the payment before it hits their credit card. Users can even send a reminder to someone else.

**Objectives**

* To add the subscriptions user has purchased.
* To provide a search feature so that users can search subscriptions.
* To see a list of deals on this platform.
* To see a list of founders having their own deals on this platform.
* To provide a feature where admin can add membership for users.
* To request admin for adding deals which purchased by users.
* To change ownership of subscriptions.

# Motivation

There are many tracking subscription application for keeping watch on all recurring payments of purchased subscriptions. But the main motive is to build this application “Track My Subscription” which will give the facilities of purchasing subscriptions and changing ownership of purchasing subscriptions.

# Summary

“Track My Subscription” is a subscription monitoring tool featuring notifications, reminders, and payment type control. It delivers automated reminders and warnings to avoid unnecessary subscriptions. It allows users to track spending month-to-month and estimate remaining budget for the current month. Users may create payment folders and view them all in a calendar. Users can also filter their view by folder or payment type to quickly find the information they need.

# Chapter 2 LITERATURE REVIEW

* 1. **Introduction**

A literature review is a narrative, analytical summary depiction of current material pertaining to a given subject or field of study. A literature review is also not a compilation of paraphrased excerpts from other sources. The fundamental goal of writing a literature review is to convey to the reader the type of knowledge and thoughts that have been established on a topic, as well as the strengths and flaws.

# Review of existing systems

There are some Track Subscriptions systems present. These are –

# Truebill

Truebill identifies your subscriptions to help you stop paying for things you no longer need. Your concierge is there when you need them to cancel unwanted subscriptions so you don't have to. [1]

# Trim

Trim is a financial management tool that enables users to view recurring charges and manage subscriptions. A free account will give you a personal finance dashboard, spending alerts and subscription management help. For additional fees, you may access bill negotiation, debt payoff and other features. One premium feature, the bill negotiation service, charges 33% of your savings. [2]

# Appsumo

A simple Notion template for keeping track of all of your subscriptions in one spot. In today's world where almost everything is accessible on a subscription basis. Whether it is a movie, music, apps, etc. Subscription is charged on a monthly or yearly basis. A person subscribes to a minimum of 10 services as per the latest survey. We are finding it more difficult to keep track of all our subscriptions as a customer because we subscribed on different dates and ends on different dates. [3]

# Comparison of our system

The main objective of this project is to add the subscriptions user has purchased. So that all users can purchase subscriptions in the system. Users can search subscriptions. They can also see the list of deals. In our application it has feature of changing ownership. But existing systems don't have this feature.

# Necessity of Methodology

We should follow a method when developing software or a system. A system or software is a complicated human-made item. The entire complicated process is created in a step-by-step technique known as a methodology for that artifact. As a result, it is critical to create a system that is consistent.

To discuss about it, we need to have a basic knowledge of why it is necessary. We are entirely aware of our goal through studying points. It describes how we will progress successfully, and it will present a methodology that will carry out the processes to construct our proposed system. We need a technique to develop our system, and a Software Life Cycle is a good fit at this time.

# Software Development Life Cycle (SDLC)

A software life cycle model (also known as a process model) is a visual depiction of the software development life cycle. A life cycle model is a diagram that depicts all of the steps involved in moving a software product through the stages of its life cycle.



## Figure 2.1. Software development life cycle.

It also encapsulates the framework in which these techniques will be implemented. SDLC is a process that gives us a broad overview and guidelines for creating high-quality software. The "System Design Life Cycle" is another name for this procedure. To make the system dependable for the user, we shall follow the SDLC process.

* + 1. **Feasibility Study**

It assists in identifying whether software can meet all criteria as intended.

# Requirement Gathering and analysis

The requirement collection and analysis phase of the SDLC is the most significant stage in the development of a system. A meeting was conducted to determine the needs. During this phase, a series of generic questions are asked and answered.

# System specification

A System Needs Specification (SRD) is a structured collection of data that encapsulates a system's requirements. Our first task in this phase is to create an entity relationship diagram, which depicts the relationship between entities and their attributes.

# System Design

There are two methods for obtaining system design: overall design and detailed design. The overall design determines the system's structural shape. A functional design strategy is called a "structure chart." Detailed design is a procedure that the entire program goes through. Our proposed system could benefit from a progressive refinement.

# Program design and coding

We followed the SDLC approach to design and code our solution. This approach necessitates only 10% of the time spent on the job. We used PHP, Laravel, HTML, JavaScript, and CSS as web-based programming languages.

# Testing

One of the most important stages in the development of a system is the testing phase. We finished our coding for design purposes in our system. Then we double-checked all of our dynamic fields. As a result, we looked for flaws. There are two ways to implement our proposed system:

* + - * White Box Testing
      * Black Box Testing

# Software Process Model

A graphical representation of an object is provided by a software process model. It depicts a software system's activity. We followed the Agile model while designing the program.

Following are the Agile manifesto principles -

* **Individuals and interactions:** We work both independently and together on our project. Individuals and interactions are valued in the Agile methodology.
* **Working software:** To understand public demand, we constructed our system step by step and tested it with our supervisor. Because the greatest way to develop is with a working demo.
* **Customer collaboration:** Our project is entirely centered on the needs of our clients. We can't get the public demand and necessity without consumer engagement. As a result, constant client connection is essential.
* **Responding to change:** We can make them rapidly and continue to develop them using the agile model.

**Advantages of Agile Model:**

* Customer satisfaction is achieved by the constant distribution of software.
* Customer, testers, and developers communicate on a regular basis.
* During the development phase, requirements can be updated.

# Summary

The existing systems and the Software Development Life Cycle are discussed in this chapter (SDLC).

**Chapter 3**

**SOFTWARE REQUIREMENT SPECIFICATION**

* 1. **Introduction**

A software requirement specification (SRS) is a process that consists of a document that describes how the system should function. Normally shut off near the end of the requirements engineering cycle. To create the system, the functional and non-functional requirements are checked here.

# Requirement Engineering

The user's needs are known to as requirements. To convert these requirements into programming, we must first identify the most important demands and desires of clients. To do so, we must analyze the framework issue area that the customer requires during the precondition building stage. Requirement engineering is the method for achieving this goal.

We examine a set of facts related to the system's aims and objectives during this procedure. It's critical to recognize that a significant portion of requirements engineering is concerned with the stakeholders or parties involved in the process.

.Requirement Engineering process consists of the following main activities:

* + - Requirement Analysis
    - Requirement Specification
    - Requirement Validation

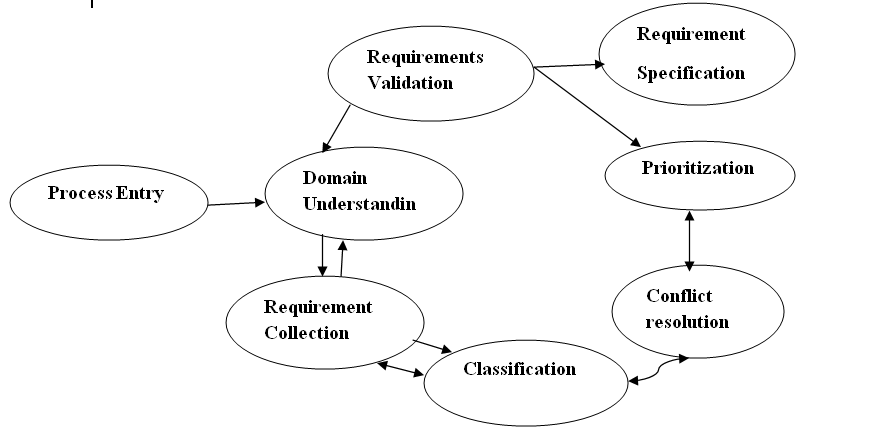
# Requirement Analysis

Requirement analysis is an important process that gathers all of the necessary information to create a requirement model. To create uniform and unambiguous requirements, we examine, improve, and scrutinize the obtained requirements.

The following methods were used to collect data for analysis:

* + - * Asking peoples
      * Observing their operations
      * Gather information with some questionnaires

We can represent requirement analysis process as the below as shown in Figure 3.1:



## Figure 3.1. Requirement analysis process model.

* **Domain Understanding**

This level is all about learning about the sector in which software systems will be used.

# Requirement Collection

The user's needs and the problem domain concept are covered in this section. Customers are the general public in this initiative.

# Conflict Resolution

When a system has a large number of subscribers, the requirements will clash. We are acutely aware of conflicts in our system. Our system is mature enough to handle any conflicts that may arise.

# Prioritization

Interaction with the user is required for prioritization in order to determine the most important requirements, among others.

# Requirement Validation

This phase determines whether they are full and accordance with the subscriber's specific requirements.

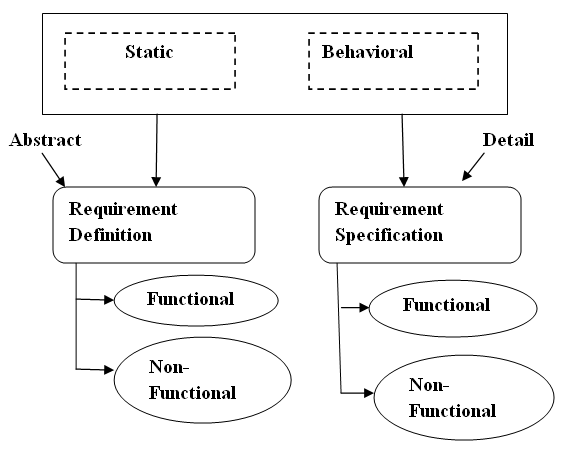
# Classification

This method divides an unorganized collection of needs into many clusters. We have four categories of user information in our system, which is stored in coherent clusters.

# Requirement Specification

The requirement specification is a technique that comprises requirements for constituents, constructions, existence, performance, and so on.

**Diagram of Requirement Specification as shown in Fig 3.2:**

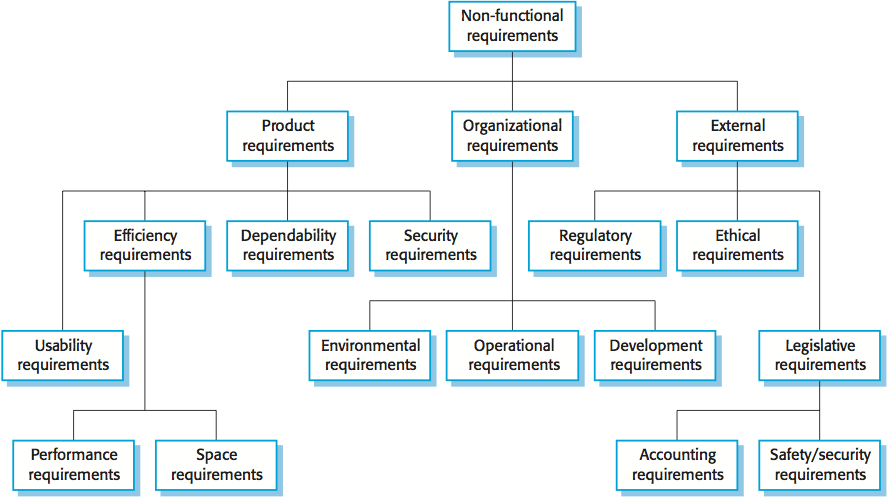


## Figure 3.2. Requirement specification modeling.

**Non Functional Requirement**

A non-functional requirement (NFR) is a set of criteria that can be used to assess a system's performance rather than specific behavior.

Non-functional requirements are represented below:



## Figure 3.3. Non-functional requirement.

* + 1. **Requirement Validation**

Gives the customer an overview of the requirements that have been developed through the system. This is an investigative stage that also identifies the issue. These issues could result in significant rework expenses.

Different types of checks are performed on the requirements:

1. **Validity Check:** Stakeholder-proposed functionality should be in line with what the system needs to do. Our system, for example, has a variety of clients with a variety of requirements.
2. **Consistency Check:** The document's requirements should be separated, which implies they should not clash.
3. **Completeness Check:** This requirement document should include specifications that define all functions and limitations.
4. **Realism Check:** The need must be verified in light of current technology in order to ensure their realization.

# Specific Requirements

**3.3.1 User Interface**

Because of the system's step-by-step processing, a user-friendly interface was required.

# 3.3.2 Hardware Interface

## Table 3.1. Hardware requirement table.

|  |  |
| --- | --- |
| Operating System | Windows 10, Linux, Unix etc. |
| Computer/Processor | Intel CORE i5 |
| Memory | At least 512 MB RAM |
| Web Server | PHP compatible web server |
| Required memory on the server | 2 GB |

**3.3.3 Software Interface**

**Table 3.2. Platform and tools table.**

|  |  |
| --- | --- |
| Database | MySQL |
| Web browser | Any PHP supported web browser such as Mozilla Firefox, Google Chrome, etc. |
| Editor | Visual Studio Code |
| Programming Language & Technologies | PHP, Wordpress, HTML, CSS, JavaScript, Vue.js, Tailwind, Headless UI. |

**3.4 Software Requirement Specification**

**3.4.1 User Requirements**

An extension might be anything from a high-level abstract declaration of services to a system limitation.

The requirements are divided into various categories. The following are some examples of required categorizations:

* + - * + Diagrams showing the service provided by the system, as well as its operational limits.
* Basically, it was written with customers in mind.

# 3.4.2 System Requirements

* + - * + Structural Documents are documents that lay out the entire description of a system's services.
* Written in the form of a contract between the contractor and the client.

# 3.4.3 Software Requirements

* + - * + A entire set of software data that can be used to create a design or implementation.
* Designed with developers in mind.

# 3.5 Summary

Software requirement specifications are discussed in this chapter. What software and hardware interfaces are needed to develop this platform, as we mentioned in this chapter.

# Chapter 4 Normalization and Data Dictionary

* 1. **Introduction**

We'll discuss about databases, database management systems (DBMS), database normalization, and data dictionaries in this chapter.

# Database

A database is a collection of data that has been organized in a systematic way. They allow for electronic data storage and manipulation. The main purpose of a database is to store data. A database management system is usually in charge of a database (DBMS). The data, the DBMS, and the applications that go with them are referred to as a database system, which is commonly shortened to just database.

# Database Management System (DBMS)

A database management system (DBMS) is software that allows you to create, define, and manipulate databases. It enables the user to conveniently store, process, and analyze data. A database management system (DBMS) provides us with an interface or tool that allows us to execute various tasks such as building a database, putting data in it, updating data, adding tables in the database, and so on. Some DBMS examples include MySQL, Microsoft Access, SQL Server, FileMaker, Oracle, RDBMS, Clipper, and FoxPro.

# Database Normalization

The process of arranging a database, usually a relational database, is known as database normalization. The goal of this phase is to optimize the storage and retrieval of data in a database. In most cases, normalization means separating existing tables into several ones.

# Normalization Rule

At various phases, normalization rules are used to edit or update bibliographic metadata.

Normalization rules are divided into the following normal forms:

1. First Normal Form (1NF)
2. Second Normal Form (2NF)
3. Third Normal Form (3NF)
4. Boyce-Codd Normal Form (BCNF)
5. Fourth Normal Form (4NF)

# First Normal Form (1NF)

The first normal form (1NF) refers to a single table within a relational database system and establishes the fundamental rules for database normalization. Normalization consists of three main steps, each of which builds on the previous. The first normal form is the most basic.

The first normal form states that:

* The table's columns must all be unique
* For each collection of related data, separate tables must be created
* A unique column or concatenated columns called the primary key must be used to identify each table
* There can't be any duplicate rows
* There can't be any duplicate columns
* There are no null values in any row/column intersections
* There are no multi valued fields at row/column intersections

# Second Normal Form (2NF)

The second normal form (2NF) is a database normalization normal form. If a relation meets both of the following conditions, it is said to be in the "second normal form."

* + - 1. It should be in its original form ( First normal Form).
      2. It has no non-prime attributes that are functionally dependent on any suitable subset of any relation's candidate key.

# Third Normal Form (3NF)

The third normal form, often known as 3NF, is one of several database normalization ideas that also includes the first normal form (1NF) and second normal form (2NF), in which no primary key attribute is transitively dependent on the primary key.

# Boyce-Codd Normal Form (BCNF)

The Boyce-Codd Normal Form, also known as 3.5 Normal Form, is an extension of the third normal form. The following two conditions must be met for a table to satisfy the Boyce-Codd Normal Form:

* + - 1. It needs to be written in the Third Normal Form
      2. A should be a super key for any dependency A → B

# Fourth Normal Form (4NF)

Other than a candidate key, the fourth normal form (4NF) is a level of database normalization in which there are no non-trivial multi valued dependencies. It is based on the Boyce-Codd Normal Form and the first three normal forms (1NF, 2NF, and 3NF) (BCNF). They create a different table in the case of multiple relationships. This protocol is maintained by a database known as Fourth Normal Form.

# Data Dictionary

A data dictionary is a well-structured document in which all data pieces are applicable to the system and have accurate descriptions, allowing users and system analysts to grasp inputs, outputs, and even intermediate calculations with ease.

In wp\_deal\_platforms table ID is the primary key and name, slug are the index. Other fields are platform\_url, platform\_logo, created\_at, updated\_at. This table is used to store data of platforms of Deals on our platform.

**Table 4.1. wp\_deal\_platforms.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| Name | varchar(255) | - | No | None |
| slug | varchar(255) | - | Yes | Null |
| platform\_url | varchar(255) | - | Yes | Null |
| platform\_logo | varchar(255) | - | Yes | NULL |
| created\_at | timestamp | - | No | Current\_timestamp() |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_deal\_types table id is the primary key and name, slug both are Index. Other fields are description, created\_at, updated\_at. This table is used to store data of types of Deals on our platform.

**Table 4.2. wp\_deal\_types.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| Name | varchar(255) | - | No | None |
| Slug | varchar(255) | - | Yes | Null |
| description | varchar(255) | - | Yes | None |
| created\_at | timestamp | - | Yes | NULL |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_pricing\_plans table ID is the primary key and product\_id, is foeign key and code, plan\_name, deal\_price, regular\_price are index. Other fields are name,link, created\_at, updated\_at. This table is used to store data of pricing plans of a deal.

**Table 4.3. wp\_pricing\_plans.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| product\_id | bigint(20) | Foreign Key | No | None |
| code | varchar(255) | - | Yes | NULL |
| features | varchar(255) | - | No | None |
| plan\_name | varchar(255) | - | No | None |
| name | varchar(255) | - | Yes | NULL |
| deal\_price | varchar(255) | - | Yes | NULL |
| screenshot | varchar(255) | - | Yes | NULL |
| link | varchar(255) | - | Yes | NULL |
| regular\_price | varchar(255) | - | Yes | NULL |
| created\_at | timestamp | - | No | Current\_timestamp() |
| updated\_at | timestamp | - | No | NULL |

In wp\_products, table ID is the primary key and product\_founder\_id , product\_category\_id are foreign keys and name, slug are the index. Other fields are description, stacking, max\_stack, ltd\_base\_price ,deal\_url, website\_url, login\_url, logo, plan\_name, deal\_type, relase\_date, end\_date, redeem\_deadline, fb\_group, twitter, youtube, blog, verdict, product\_status, featured\_image, deal\_status, currency, avg\_rating, rating\_user\_count, deal\_document, deal\_platform\_id, created\_at, updated\_at. This table is the table where we stored our deals as product.

**Table 4.4. wp\_product.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint (20) | Primary Key | No | None |
| product\_founder\_id | int (11) | Foreign Key | No | None |
| Product\_caregory\_id | varchar (255) | Foreign Key | No | None |
| name | varchar (255) | - | No | NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| slug | varchar (255) | - | Yes | NULL |
| description | text | - | Yes | NULL |
| stacking | tinyint (1) | - | Yes | NULL |
| max\_stack | varchar (255) | - | Yes | NULL |
| Itd\_base\_price | varchar (255) | - | Yes | NULL |
| deal\_url | varchar (255) | - | Yes | NULL |
| website\_url | varchar (255) | - | Yes | NULL |
| login\_url | varchar (255) | - | Yes | NULL |
| logo | varchar (255) | - | Yes | NULL |
| plan\_name | varchar (255) | - | Yes | NULL |
| deal\_type | varchar (255) | - | Yes | NULL |
| release\_date | date | - | Yes | NULL |
| end\_date | date | - | Yes | NULL |
| redeem\_deadline | int (11) | - | Yes | NULL |
| fb\_group | varchar (255) | - | Yes | NULL |
| twitter | varchar (255) | - | Yes | NULL |
| youtube | varchar (255) | - | Yes | NULL |
| blog | longtext | - | Yes | NULL |
| verdict | text | - | Yes | NULL |
| product\_status | varchar (255) | - | Yes | NULL |
| feature\_image | varchar (255) | - | Yes | NULL |
| deal\_status | varchar (255) | - | Yes | NULL |
| currency | varchar (255) | - | Yes | NULL |
| avg\_rating | float | - | Yes | NULL |
| rating\_user\_count | bigint(20) | - | Yes | NULL |
| deal\_document | text | - | Yes | NULL |
| deal\_platform\_id | bigint(20) | Foreign Key | Yes | NULL |
| created\_at | timestamp | - | Yes | NULL |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_product\_categories table ID is the primary key and name and slug both are index.

Other fields are description, created\_at and updated\_at. Categories of deals are stored in this table.

**Table 4.5. wp\_product\_categories**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| name | varchar (255) | - | No | None |
| slug | varchar (255) | - | No | None |
| description | varchar (255) | - | Yes | NULL |
| created\_at | timestamp | - | Yes | current\_timestamp() |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_product\_founders table ID is the primary key. Other fields are avatar, twitter, facebook, website, email, youtube, linkedin, country, bio, created\_at and updated\_at. Founders info are stored in this table.

**Table 4.6.wp\_product\_founders.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| name | varchar (255) | - | Yes | None |
| avatar | varchar (255) | - | Yes | NULL |
| twitter | varchar (255) | - | Yes | NULL |
| facebook | varchar (255) | - | Yes | NULL |
| website | varchar (255) | - | Yes | NULL |
| email | varchar (255) | - | Yes | NULL |
| youtube | varchar (255) | - | Yes | NULL |
| linkedin | varchar (255) | - | Yes | NULL |
| country | varchar (255) | - | Yes | NULL |
| bio | varchar (255) | - | Yes | NULL |
| created\_at | timestamp | - | No | current\_timestamp() |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_product\_rating table ID is the primary key and product\_id, user\_id are foreign keys. Other fields are review,rating, created\_at and udpated\_at. This table is used to store rating of a deal.

**Table 4.7. wp\_product\_raitng.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| product\_id | bigint(20) | Foreign Key | No | None |
| user\_id | bigint(20) | Foreign Key | No | None |
| rating | Float(5.2) | - | Yes | NULL |
| review | varchar (255) | - | Yes | NULL |
| created\_at | timestamp | - | No | current\_timestamp() |
| updated\_at | timestamp | - | No | NULL |

In wp\_product\_with\_deal\_platforms table, ID is the primary key and product\_id, deal\_platform\_id are the foreign key. Other fields are created\_at and updated\_at. This table is used to store multiple deals info on single platform.

**Table 4.8. wp\_product\_with\_deal\_platform.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| product\_id | bigint(20) | Foreign Key | No | None |
| deal\_platform\_id | bigint(20) | Foreign Key | No | None |
| created\_at | timestamp | - | Yes | NULL |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_subs table, ID is the primary key and user\_id, product\_id, pricing\_plan\_id, subs\_tag\_id, deal\_platform\_id, deal\_type\_id,subs\_folder\_id, are foreign key . Other fields are migration, name, stacked, price, status, bought\_date, refund\_date, redeem\_date, payment\_processor and plan. This table is used to store the data of subscriptions, user has purchased from various deals.

**Table 4.9. wp\_subs.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| user\_id | bigint(20) | Foreign Key | No | None |
| product\_id | bigint(20) | Foreign Key | Yes | NULL |
| Pricing\_plan\_id | bigint(20) | Foreign Key | Yes | NULL |
| subs\_tag\_id | bigint(20) | Foreign Key | Yes | NULL |
| deal\_platform\_id | bigint(20) | Foreign Key | Yes | NULL |
| deal\_type\_id | bigint(20) | Foreign Key | Yes | NULL |
| subs\_folder\_id | bigint(20) | Foreign Key | Yes | NULL |
| name | varchar (255) | - | No | NULL |
| stacked | varchar (255) | - | Yes | NULL |
| price | varchar (255) | - | Yes | NULL |
| status | varchar (255) | - | Yes | NULL |
| bought\_date | date | - | Yes | NULL |
| refund\_date | date | - | Yes | NULL |
| redeem\_date | date | - | Yes | NULL |
| signup\_email | varchar (255) | - | Yes | NULL |
| payment\_processor | varchar (255) | - | Yes | NULL |
| login\_id | varchar (255) | - | Yes | NULL |
| currency | varchar (255) | - | Yes | NULL |
| invoice | varchar (255) | - | Yes | NULL |
| cycle | varchar (255) | - | Yes | NULL |
| folder | varchar (255) | - | Yes | NULL |
| retail\_price | varchar (255) | - | Yes | NULL |
| listing\_price | varchar (255) | - | Yes | NULL |
| sold\_for | varchar (255) | - | Yes | NULL |
| for\_sale | tinyint(1) | - | No | NULL |
| Notes | varchar (255) | - | Yes | NULL |
| privacy | varchar (255) | - | Yes | NULL |
| plan | varchar (255) | - | Yes | NULL |
| created\_at | timestamp | - | No | NULL |
| updated\_at | timestamp | - | Yes | NULL |

In wp\_subs\_folder table, ID is the primary key and user\_id is the foreign key. Other fields are name, slug, created\_at and updated\_at. This table is used to store folder data of subs.

**Table 4.10. wp\_subs\_folder.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| user\_id | bigint(20) | Foreign Key | No | None |
| name | varchar (255) | - | No | None |
| slug | varchar (255) | - | No | None |
| created\_at | timestamp | - | No | NULL |
| updated\_at | timestamp | - | No | NULL |

In wp\_subs\_reminder table, ID is the primary key and subs\_id is the foreign key. Other fields are one\_contact, one\_note, one\_date, two\_contact, two\_note, two\_date, three\_contact, three\_notes, three\_date created\_at, updated\_at. Here in this table we are storing reminder info of a subscription.

**Table 4.11. wp\_subs\_reminder.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| subs\_id | bigint(20) | Foreign Key | No | None |
| one\_contact | varchar (255) | - | Yes | NULL |
| one\_note | varchar (255) | - | Yes | NULL |
| one\_date | date | - | Yes | NULL |
| two\_contact | varchar (255) | - | Yes | NULL |
| two\_note | varchar (255) | - | Yes | NULL |
| two\_date | date | - | Yes | NULL |
| three\_contact | varchar (255) | - | Yes | NULL |
| three\_note | varchar (255) | - | Yes | NULL |
| three\_date | date | - | Yes | NULL |
| created\_at | timestamp | - | Yes | NULL |
| updated\_at | timestamp | - |  | NULL |

In wp\_subs\_tags table, ID is the primary key and user\_id is foreign key. Other fields are name, slug, created\_at and updated\_at. Subscription tag info is storing in this table.

**Table 4.12. wp\_subs\_tags.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Type** | **Index** | **Null** | **Default** |
| ID | bigint(20) | Primary Key | No | None |
| user\_id | bigint(20) | Foreign Key | No | None |
| name | varchar (255) | - | No | None |
| slug | varchar (255) | - | Yes | NULL |
| created\_at | timestamp | - | Yes | NULL |
| updated\_at | timestamp | - | Yes | NULL |

**4.7 Summary**

We discussed about databases, database management systems (DBMS), normalization, and data dictionaries in this chapter.

# Introduction

**Chapter 5 Modeling**

We'll discuss about our project's use case diagram, activity diagram, and ER diagram in this chapter.

# Use Case Diagram

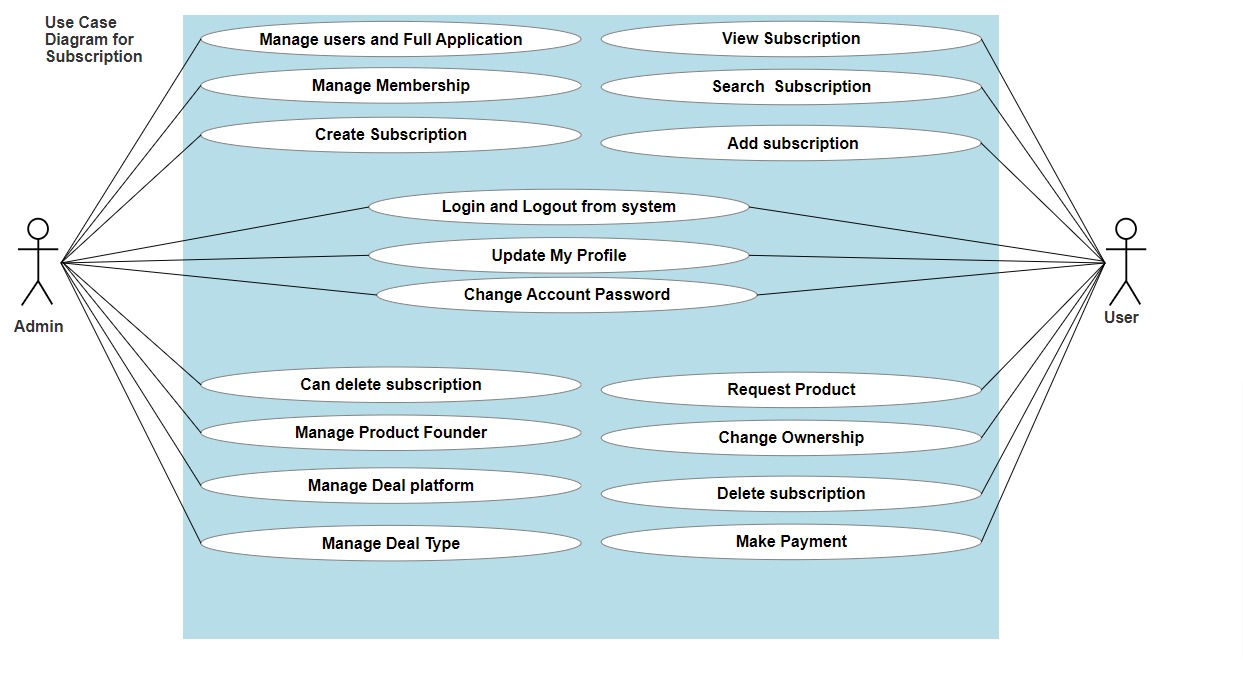
Use case diagrams are the scenario-based method, which identifies the actions of communication.

# Admin

* Admin can see the list of users of this platform. He can approve or disapprove user
* Admin can add and manipulate all the membership functionality. He can approve and disapprove membership update request
* Admin can see, add, update and delete products on this platform
* Admin can see, add, update and delete subscription on this platform
* Admin can see, add, update and delete subscription deal type on this platform
* Admin can view and update his profile
* Admin can see, add, update and delete founders on this platform
* Admin can see, add, update and delete rating
* Admin can see list of membership on this platform
* Admin can see users request of adding products
* Admin can track ownership change of subscription

**User**

* Users can register and login on this platform
* Users can view and update his profile
* Users can see, update and delete subscription from his profile
* User can buy subscription on this platform
* User can see list of subscription on this platform
* User can search subscription using this platform
* User can request to update the membership to premium after completing the subscription



**Figure 5.1. Use case diagram of user and admin.**

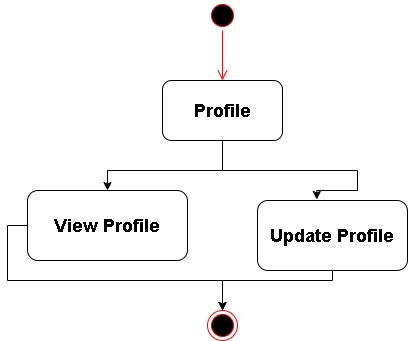
# 5.3 Activity Diagram

## 5.3.1 Activity Diagrams for User

On this platform, there are three kinds of users. They are user, Premium, & Admin. All of the users will get a separate profile section. In this profile section, they can see their subscription information, expire date and accomplishment information as well. They will also get search functionality.

**Profile**

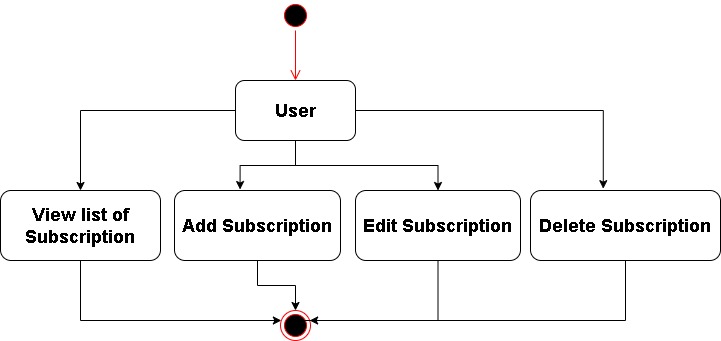
In this platform, every user will get an individual profile section. He can see his profile information and can update information as well like country, region, mobile number.



**Figure 5.2. Activity diagram of profile.**

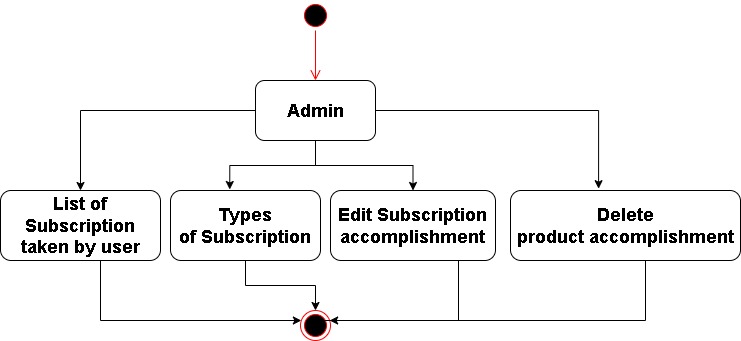
**Subscription**

Every user can see his subscription information in his profile. Users can add new subscription information. Users can update or delete their subscription information as well.



**Figure 5.3. Activity diagram of subscription.**

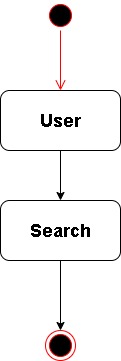
**Accomplishment**

Every user can see his accomplishment information in his profile. Users can add new subscription accomplishment information. User can update or delete his accomplishment information as well. 

**Figure 5.4. Activity diagram of accomplishment.**

**Search**

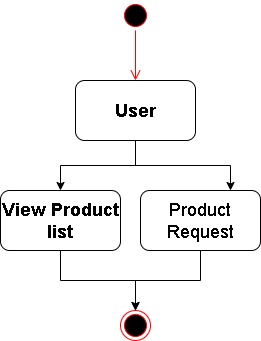
Every user can search for subscription. They can also filter the search result as well.



**Figure 5.5. Activity diagram of search.**

**User Product Request**

Every user can Request for subscription.

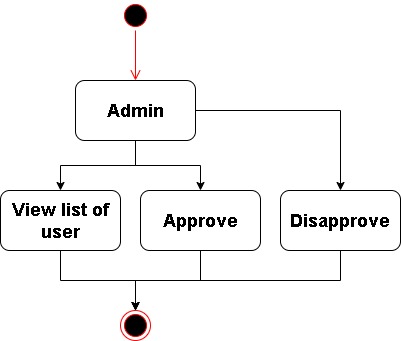


**Figure 5.6. Activity diagram of user product request.**

## 5.3.2 Activity Diagrams of Admin

**All Admin**

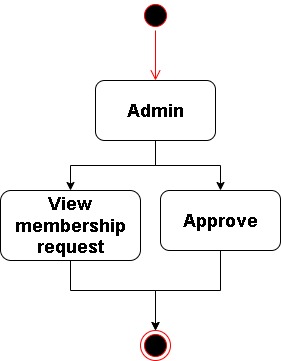
Admin can see the list of users in our platform. He can approve or disapprove users by checking their information.



**Figure 5.7. Activity diagram of all users.**

**Membership Request** **(For admin)**

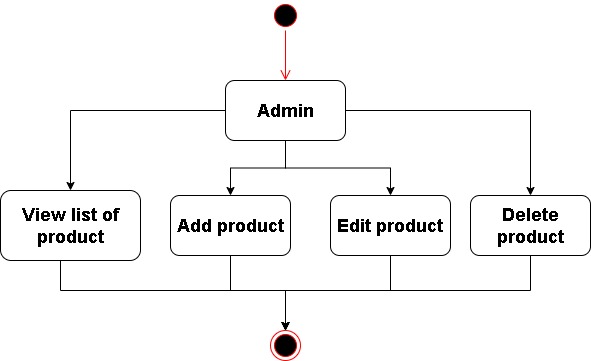
After taking subscription, users can update their membership to Premium by providing necessary information. Again, a premium can update his membership to platinum by providing necessary information. Admin can see all the requests of the users and after checking he can approve membership update requests.



**Figure 5.8. Activity diagram of membership request for Admin.**

**Admin Product View**

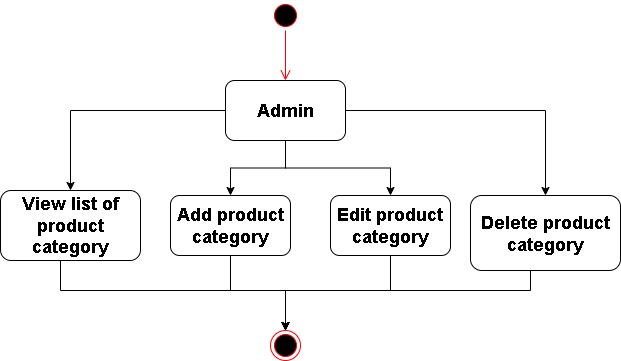
After Adding products, admin can see all the product list. Admin can edit and delete all the products.



**Figure 5.9. Activity diagram of Admin Product view.**

**Admin Product Category**

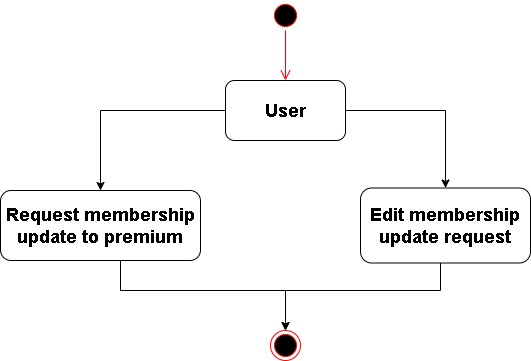
Admin can see the list of categories in our platform. He can edit or delete category.



**Figure 5.10. Activity diagram of all categories.**

**Membership Request (For User)**

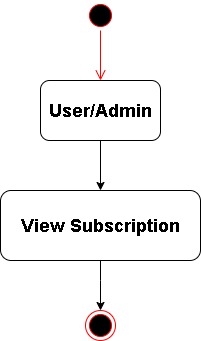
After completing subscription, user can request to update their membership to premium with necessary information. If they provide wrong information accidentally, they can update their request also.



**Figure 5.11. Activity diagram of membership request for user.**

**Admin / User**

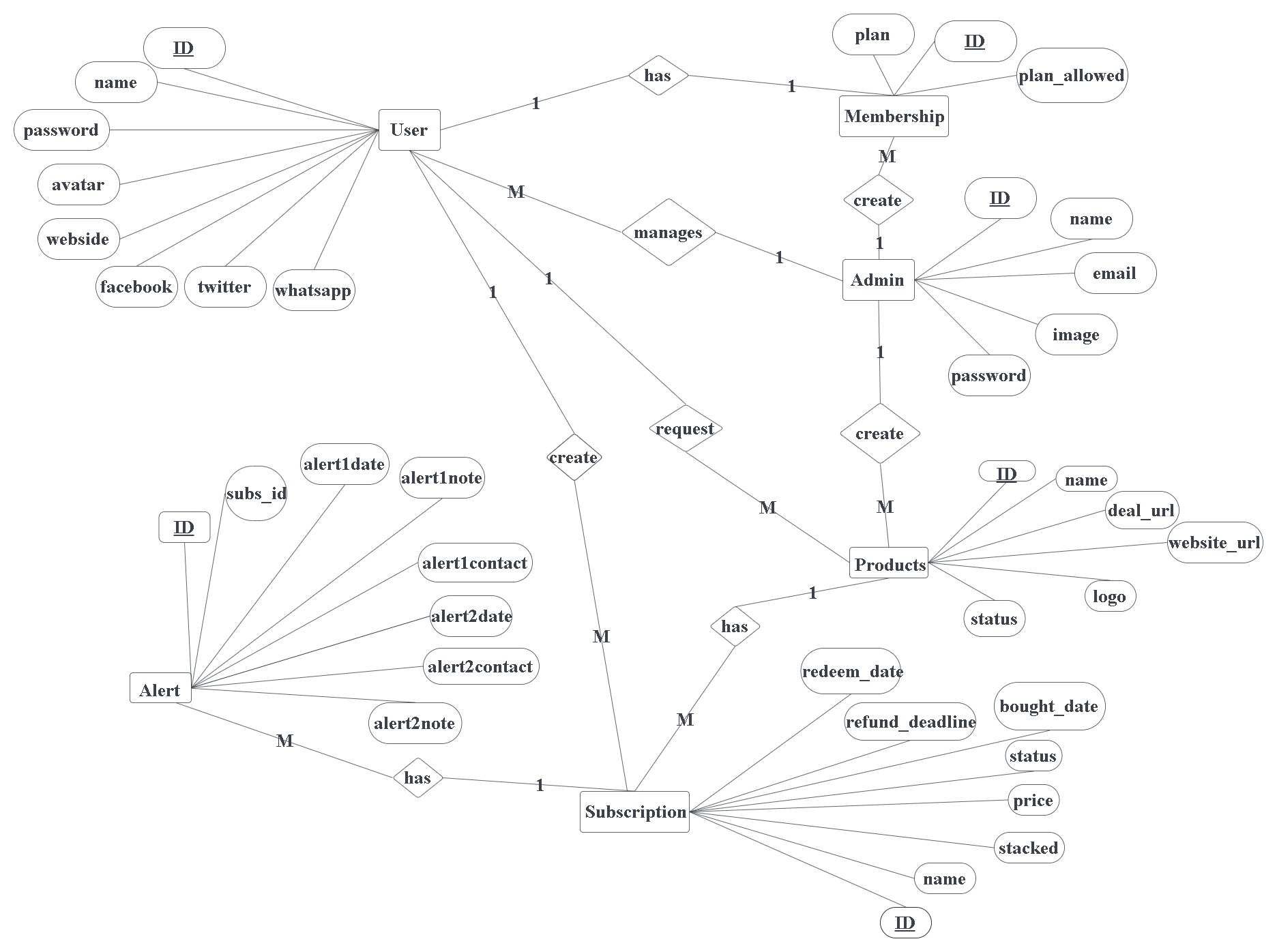
They can see all the subscriptions that are made by Admin on our platform.



**Figure 5.12: Activity diagram of subscription for admin and user**.

# 

# 5.4 ER Diagram



**Figure 5.13. ERD.**

# 5.5 Summary

We demonstrated how our product works by utilizing a use case diagram and an activity diagram.

# Introduction

**Chapter 6 SOFTWARE IMPLEMENTATION**

We searched for all known and available materials that could assist us as we made the decision to conduct the project. Implementing a website is difficult since it entails storing and retrieving a large amount of data. We employed a relational data-based system and an object-oriented database connection to make string and searching more reliable, easier, and faster than ever before. PHP, Wordpress, HTML, CSS, JavaScript, Vue.js, Tailwind, and Headless UI have all made significant contributions to the development of websites. We've employed all of them to ensure the integrity of our project.

# Software Implementation

We implement our software using given below:

* + 1. PHP
    2. Wordpress
    3. HTML
    4. CSS
    5. JavaScript
    6. Vue.js
    7. MySQL
    8. Tailwind

# PHP

PHP is a server side scripting language. Basically, it is used to develop static websites or dynamic websites or web applications. PHP scripts are illustrated on a server which has PHP installed. This language is very efficient for web development, and can easily be embedded in HTML code. [4]

**WordPress**

WordPress is free, open source publishing software that can be installed locally on a web server and viewed on a proprietary web site or hosted in the cloud and viewed on the WordPress web site. In 2001, Matt Mullenweg and Mike Little co-founded WordPress, launching it in May 2003 with the help of several core contributing developers. At that time, WordPress was merely a free blogging forum for users. Since then, WordPress has evolved to be a full content management system with widgets, plugins and customizable themes. WordPress is licensed under GPLv2, which lists terms and conditions for copying, modifying and distributing free software.[5]

# HTML

HTML stands for Hypertext Markup Language. It allows the user to create and structure sections, paragraphs, heading, links and block quotes for web pages and applications. HTML is not a programming language, meaning it doesn’t have the ability to create dynamic functionality. Instead, it makes it possible to organize and format documents, similarly to Microsoft Word. HTML documents are files that end with a .html extension. You can view them using any web browser (such as Google Chrome, Safari or Mozilla Firefox). The browser reads the HTML file and renders its content so that internet users can view it. [6]

# CSS

CSS (Cascading Style Sheets) allows you to create great looking web pages, but how does it work under the hood? This article explains what CSS is, with a simple syntax example, and also covers some key terms about the language. CSS can be used for very basic document text styling For example changing the color and size of headings and links. It can be used to create a layout. Besides turning a single column of text into a layout with a main content area and a sidebar for related information. It can even be used for effects such as animation. Have a look at the links in this paragraph for specific examples. [7]

# JavaScript

JavaScript is one of the most popular scripting languages. At present, millions of web pages improve their design, validate forms, detect browsers, create cookies and much more. It runs almost all the major browsers such as google chrome, internet explorer firefox, opera.

* It is a lightweight programming language
* JavaScript directly embedded to HTML pages

● Easy to use require no license [8]

# Vue.js

Vue (pronounced /vjuː/, like view) is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with modern tooling and supporting libraries. [9]

# MySQL

MySQL is a freely available open source RElational Database Management System (RDBMS) that uses Structured Query Language (SQL). SQL is the most popular language for adding, accessing and managing content in a database. It is most noted for its quick processing, proven reliability, ease and flexibility of use. [10]

# Tailwind

Tailwind CSS is basically a utility-first CSS framework for rapidly building custom user interfaces. It is a highly customizable, low-level CSS framework that gives you all of the building blocks you need to build bespoke designs without any annoying opinionated styles you have to fight to override. [11]

# Web Server

A web server is a program that uses HTTP (Hypertext Transfer Protocol) to serve the files that form Web pages to users, in response to their requests, which are forwarded by their computer’s HTTP clients. [12]

# Apache

Apache is the most widely used web server software which is developed and maintained by Apache Software Foundation. It can be customized to meet the needs of many different environments by using extensions and modules. Most of the WordPress hosting providers use Apache for their web server software. [13]

# Web Browser

A browser is a software application used to locate, retrieve and display content on the World Wide Web, including web pages, images, video and other files. We used Google Chrome to develop and run the system. Because it is more reliable for databases.[14]

# Text Editor Visual Studio Code

Visual Studio Code is a lightweight but powerful source code editor. It runs on desktop and is also available for Windows, macOS and Linus. That provides different facilities with build-in support for JavaScript, TypeScript and Node.js etc. [15]

* IntelliSense for programming language
* Efficient command palette
* Highly integrated version control
* Debugging
* Side by side edition on different files

# Software XAMPP

XAMPP is a free and open source, cross-platform where web server, solution stack package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming language. It is very easy to use because of its several functionality.

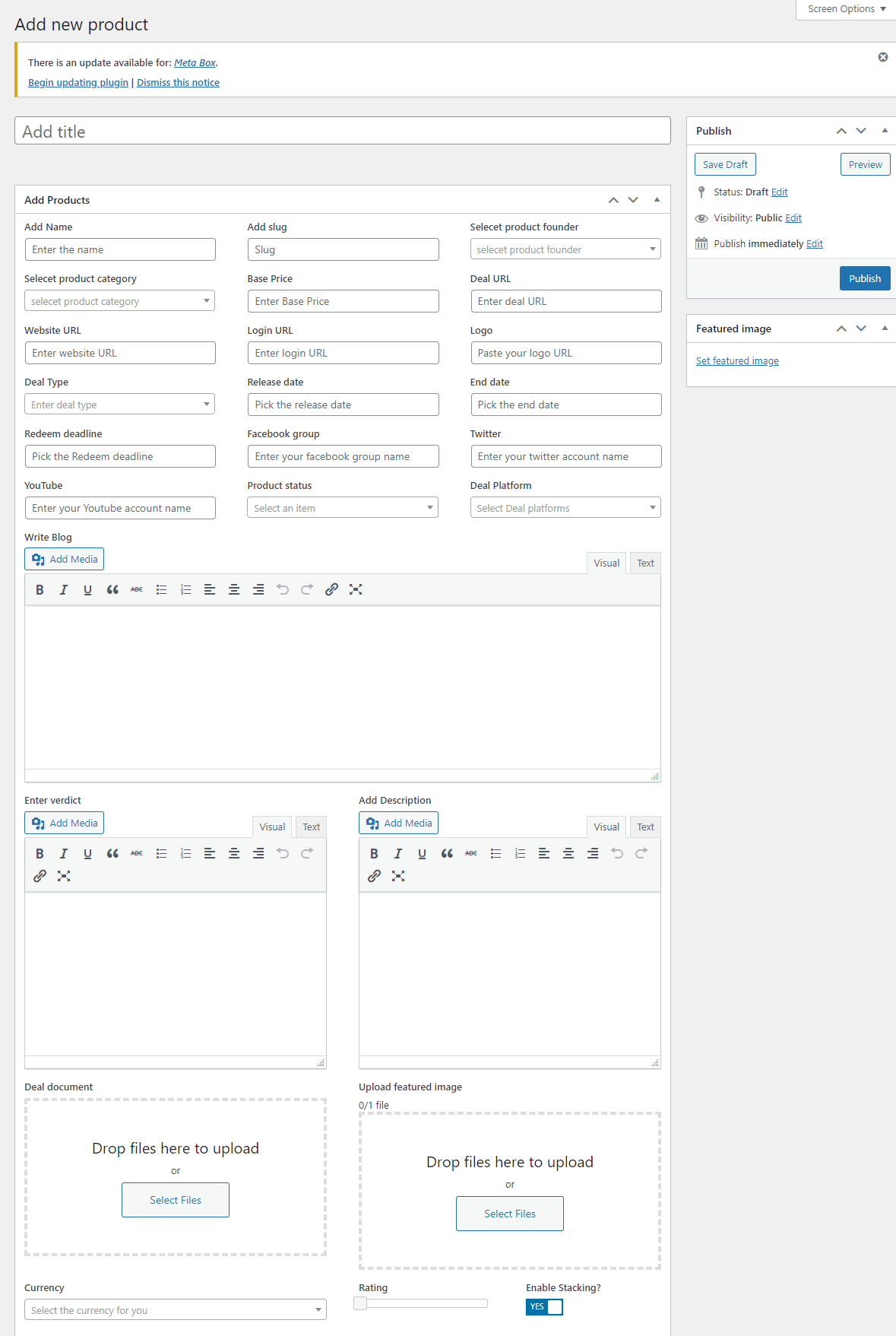
* Installation procedure is very easy
* XAMPP is available for Windows and Linus
* It has several modules
* Offers both a full and a standard version [16]

# 

# 6.6 Snapshots

## 6.6.1 Product Add

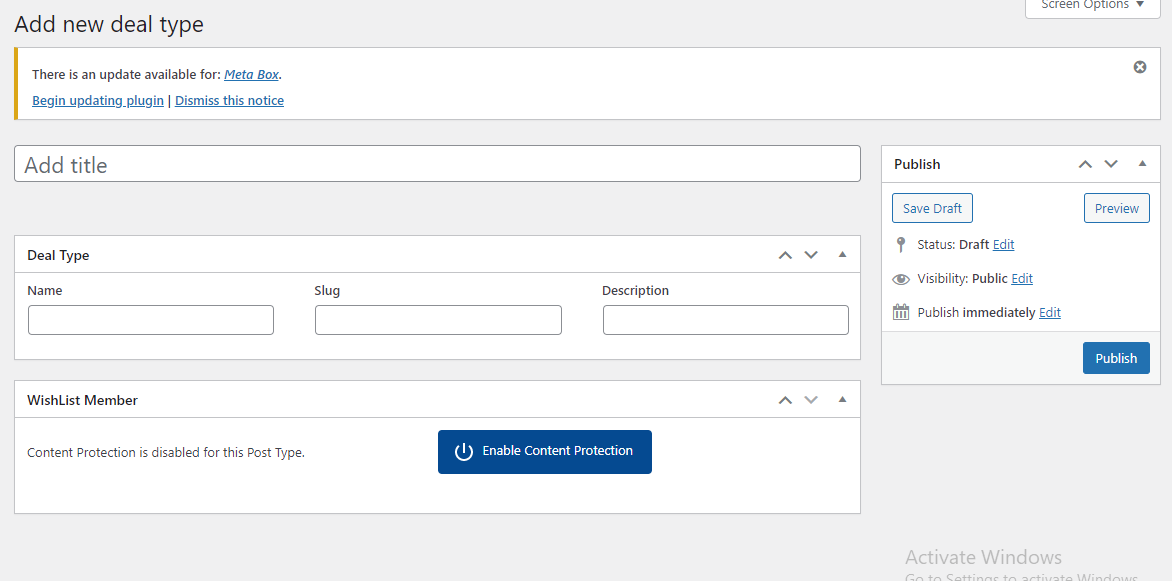
Admin can easily add product to our system by providing their name, product\_founder\_id, product\_category\_id, name, slug and their membership type. After successful form submission, the admin will check the product information.



**Figure 6.1. Admin Product Add.**

## 6.6.2 Add new Deal Type

Admin can easily create new deal type and also edit deal type on our system.

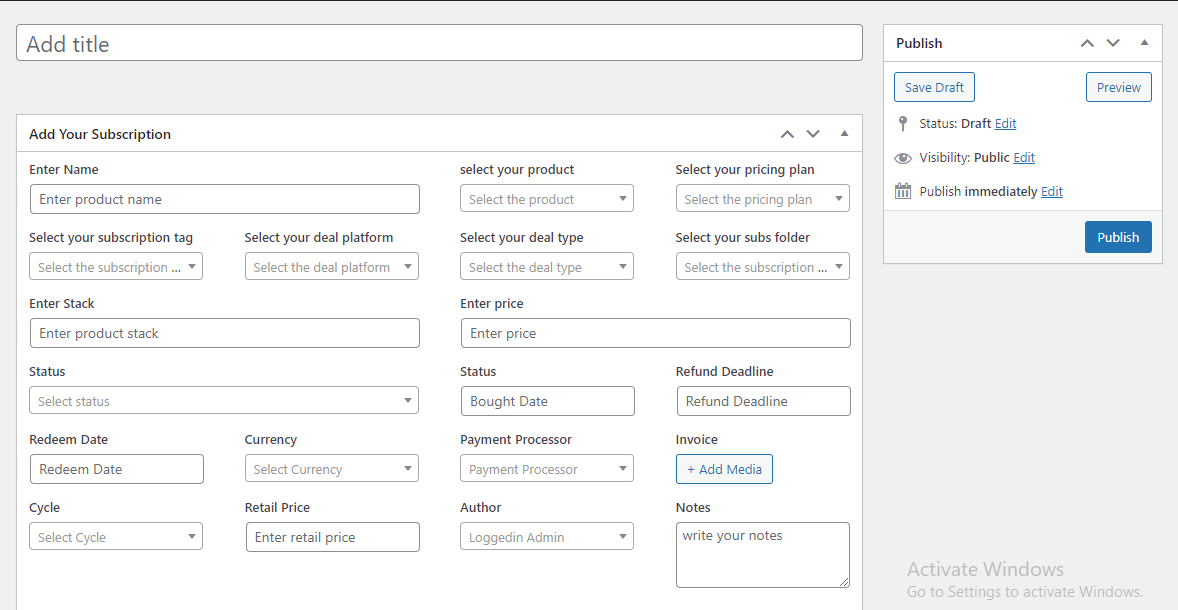


**Figure 6.2. Add new deal type.**

## 

## 6.6.3 Subscription Add

Admin can easily add subscription into our system. All other users are able to see all the subscription. Users are also able to buy or delete their subscription.

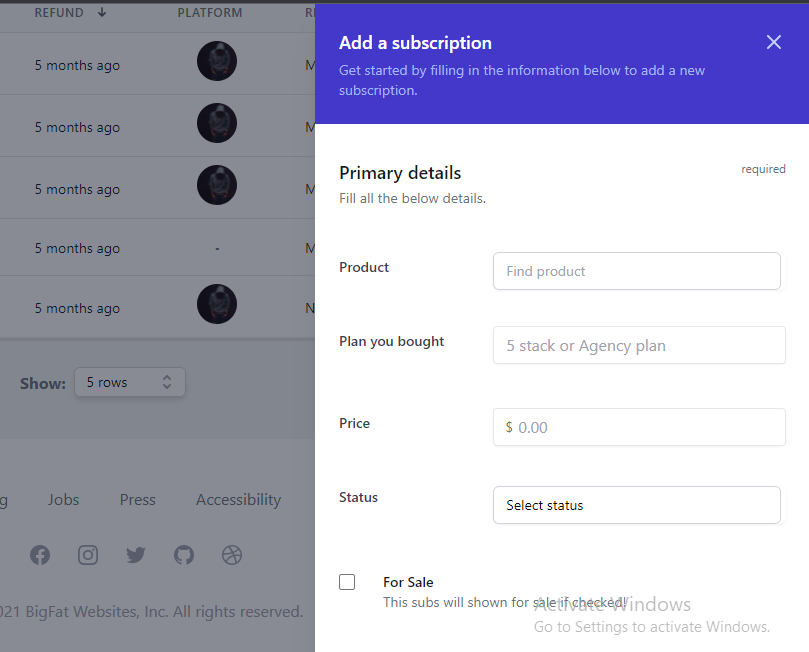


**Figure 6.3. Subscription Add.**

## 

## 6.6.4 Add Subscription by User

User can easily add subscription. Users are able to receive notifications.

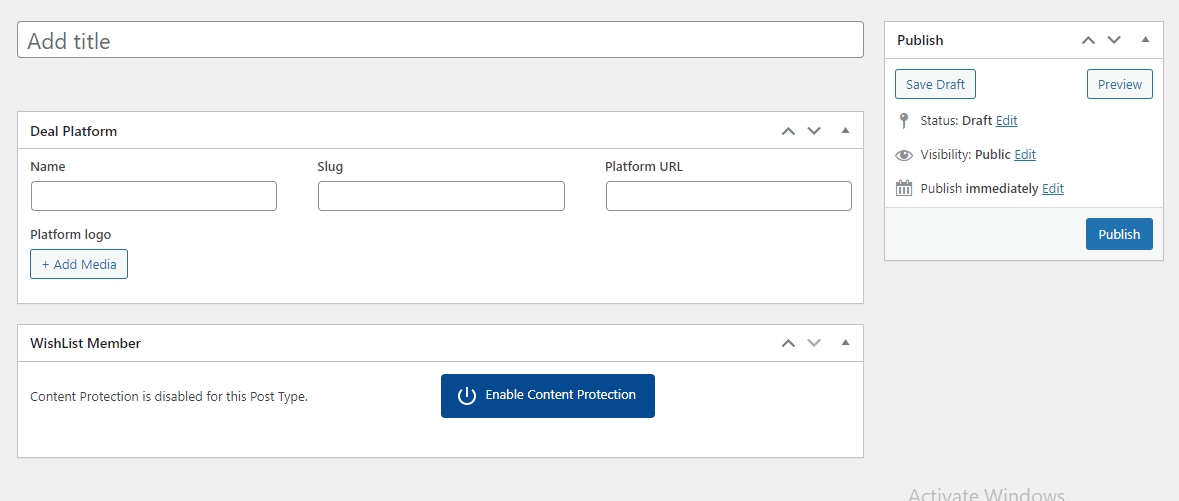


**Figure 6.4. Subscription Add By User.**

## 

## 6.6.5 Deal Platform

Admin can easily create deal platform and also edit deal platform.

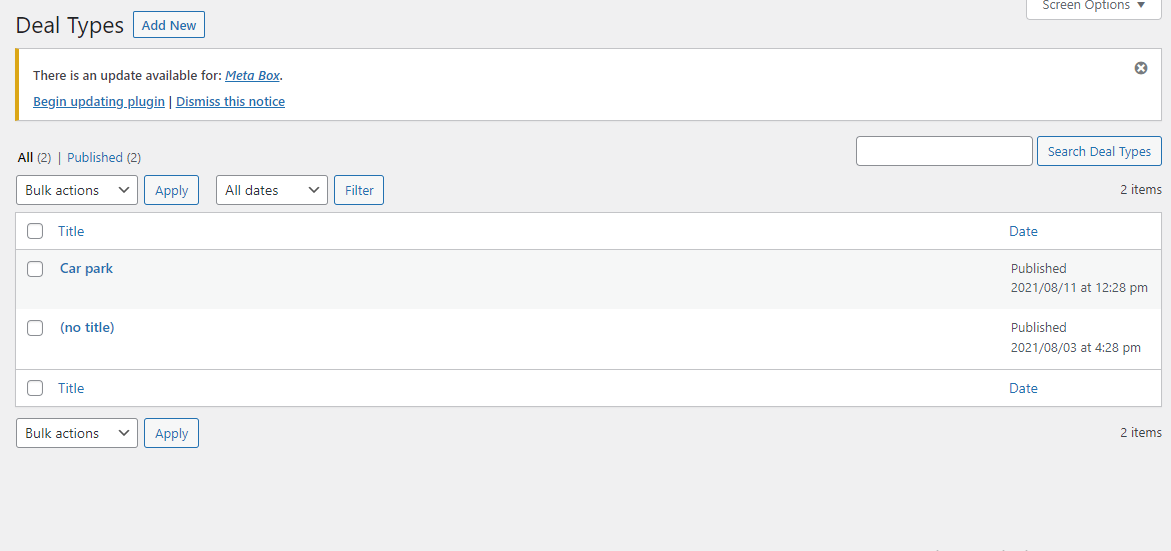


**Figure 6.5. Deal Platform.**

## 

## 6.6.6 Deal Type

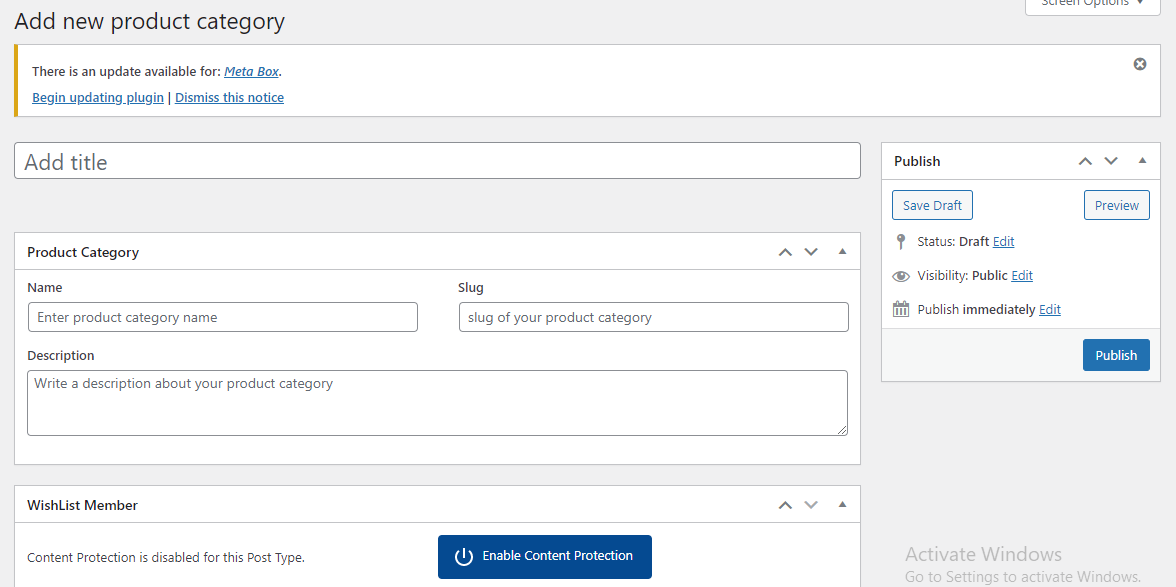
Admin can add deal type which subscription user want to buy.



**Figure 6.6. Deal Type.**

**6.6.7 Product Category**

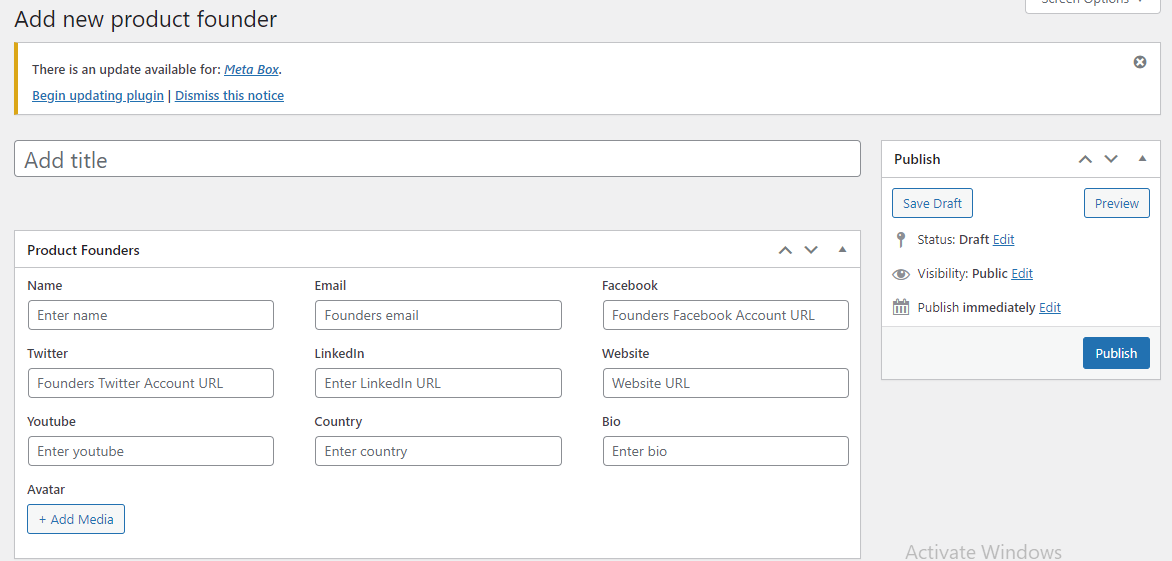
Admin can easily add category to our system by providing their name, slug and description. After successful form submission, the admin will check the product category.



**Figure 6.7. Product Category.**

**6.6.8 Add New Product Founder**

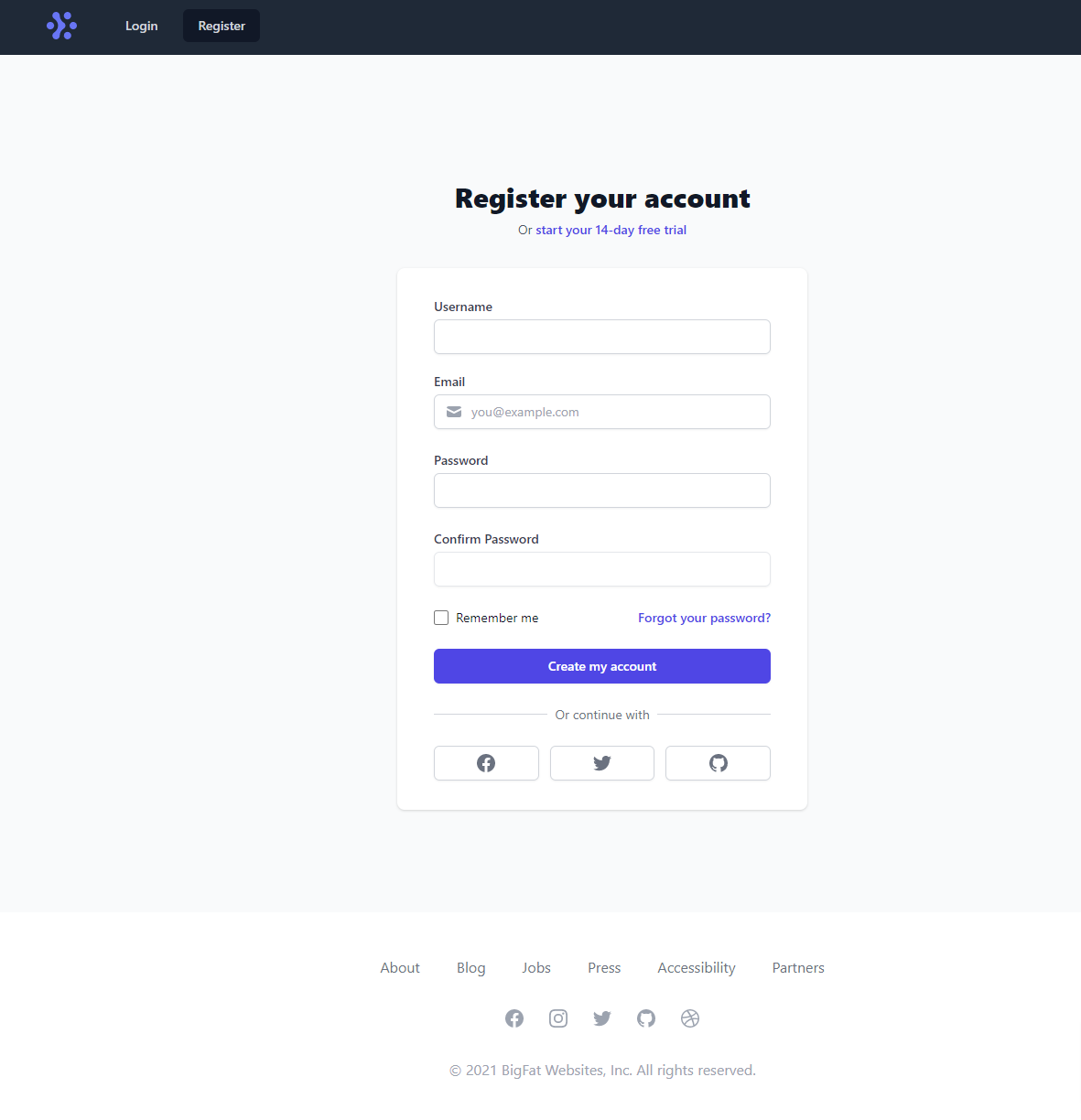
Admin can easily add new product founder to our system by providing their name, email, Facebook, Twitter, country and Avatar. After successful form submission, the admin will check the product founder.



**Figure 6.8. Product Founder.**

**6.6.9 User Register**

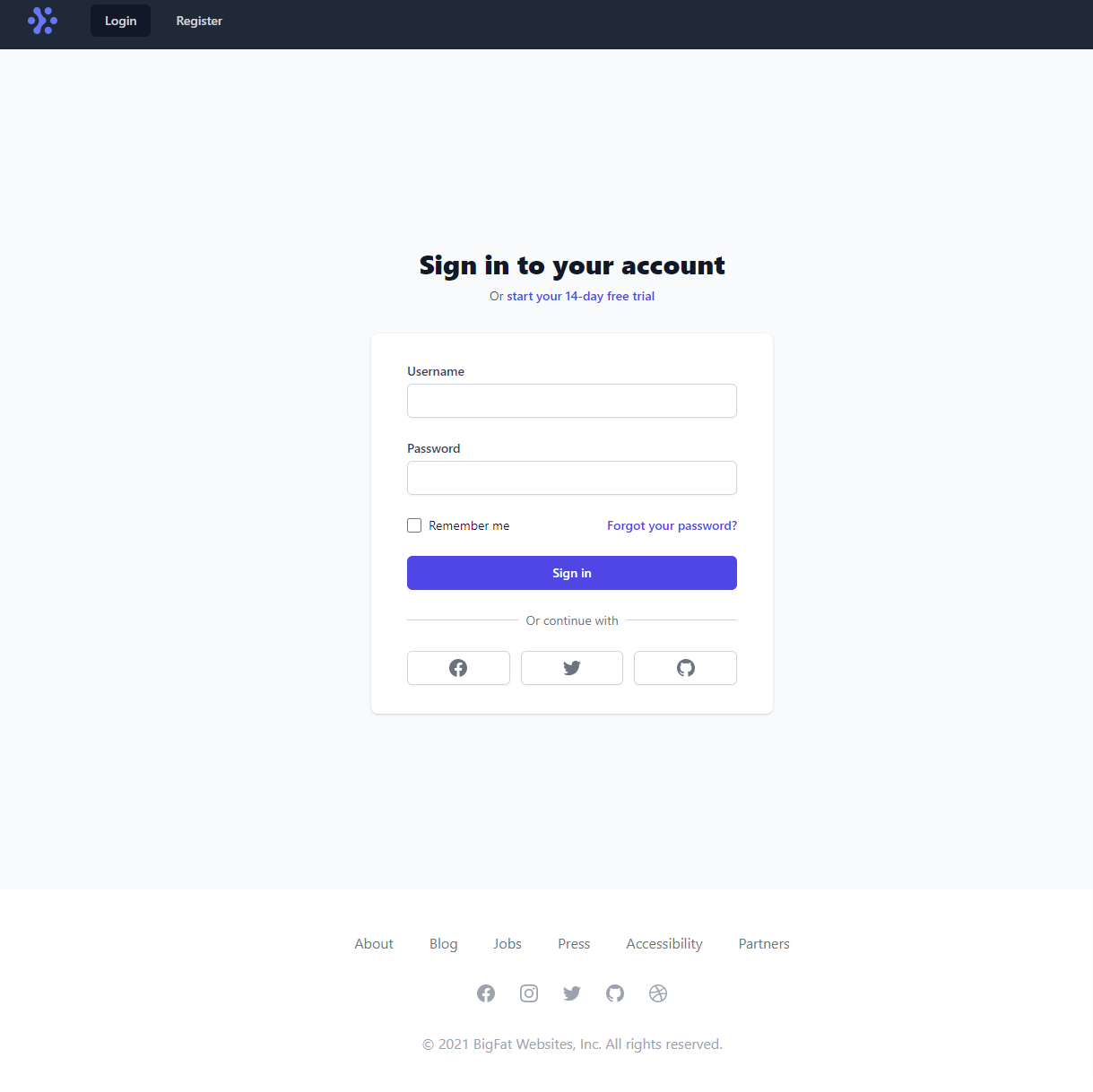
User can register on our system, user can easily signup.



**Figure 6.9. User register.**

**6.6.10 User Login**

User can login on our system, user can easily sign in.

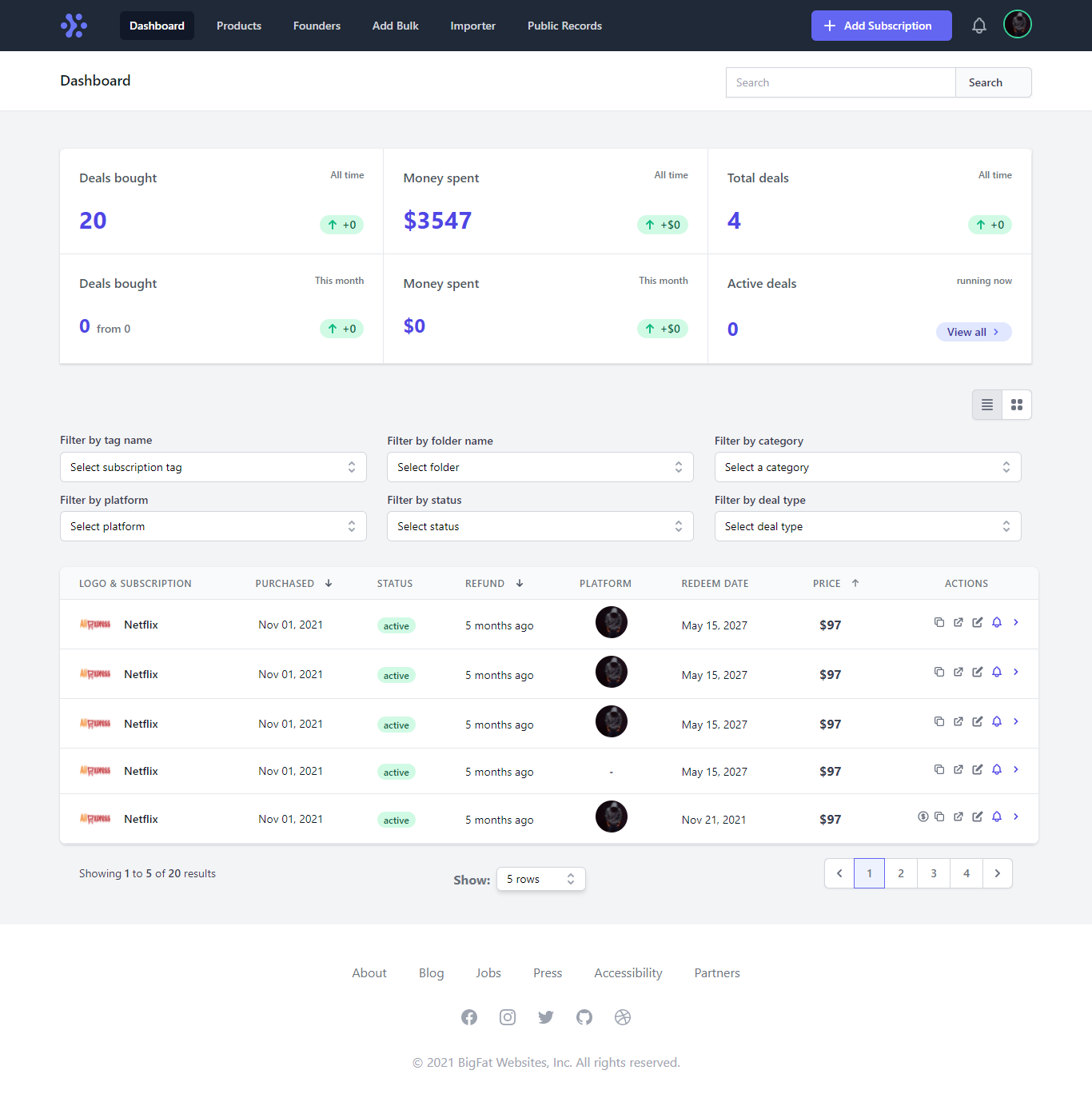


**Figure 6.10. User login.**

**6.6.11 User Dashboard**

In user dashboard, user can easily see –

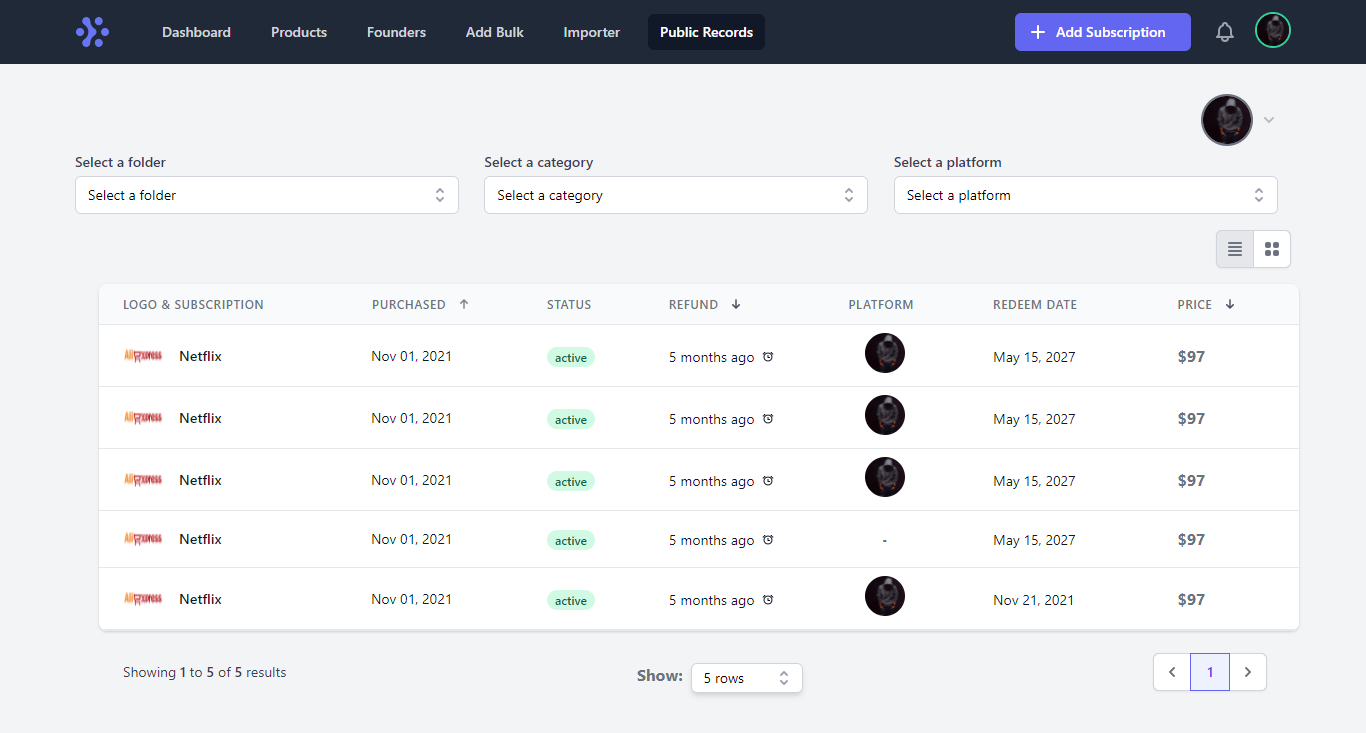
1. Total number of users in our system.
2. Total number of subscriptions.
3. Total number of subscription types.
4. Total number of requests to update membership.
5. Total number of subscriptions sold.
6. Total number of active deals.
7. Total numbers of deals bought.
8. Total amount of money spent.
9. Total numbers of active deals.



**Figure 6.11. User dashboard.**

**6.6.12 All Subscription in User Dashboard**

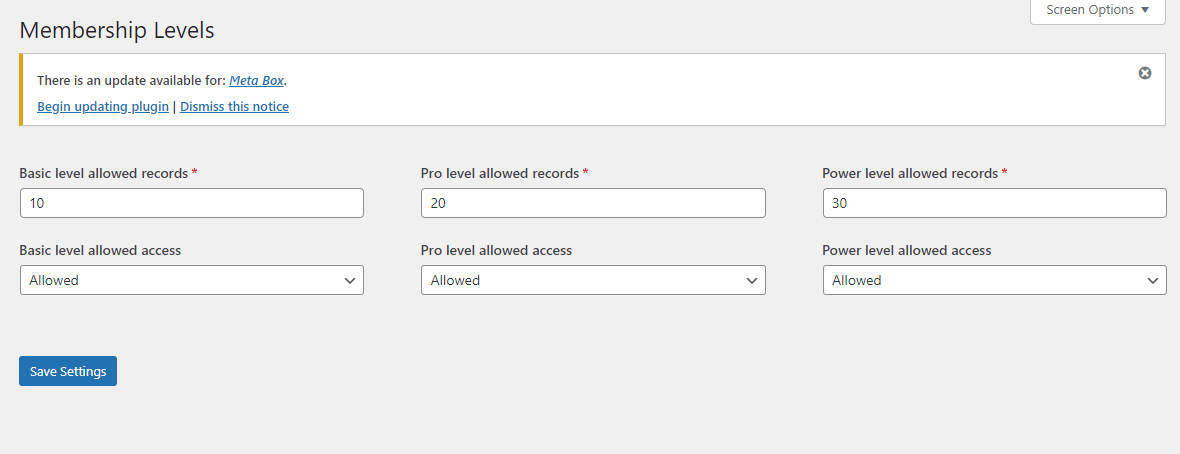
By clicking into the All-Users tab, User will see a list of subscription with their information. User can filter as well by clicking into the top 3 tab to see folder, category and platform for see specific subscription.



**Figure 6.12. All subscriptions.**

**6.6.13 Membership Level**

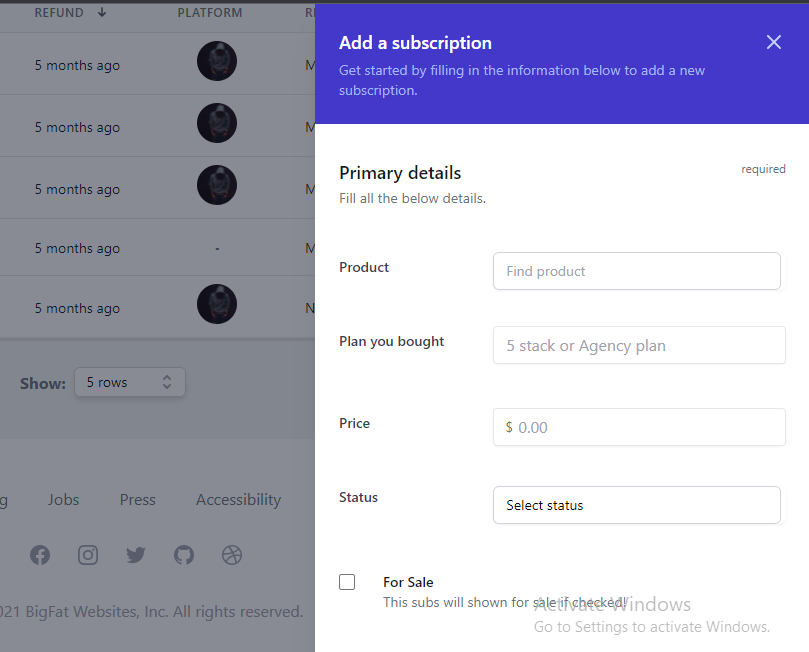
User can get a membership plan on our system.



**Figure 6.13. Membership Level.**

**6.6.14 User Subscription Add**

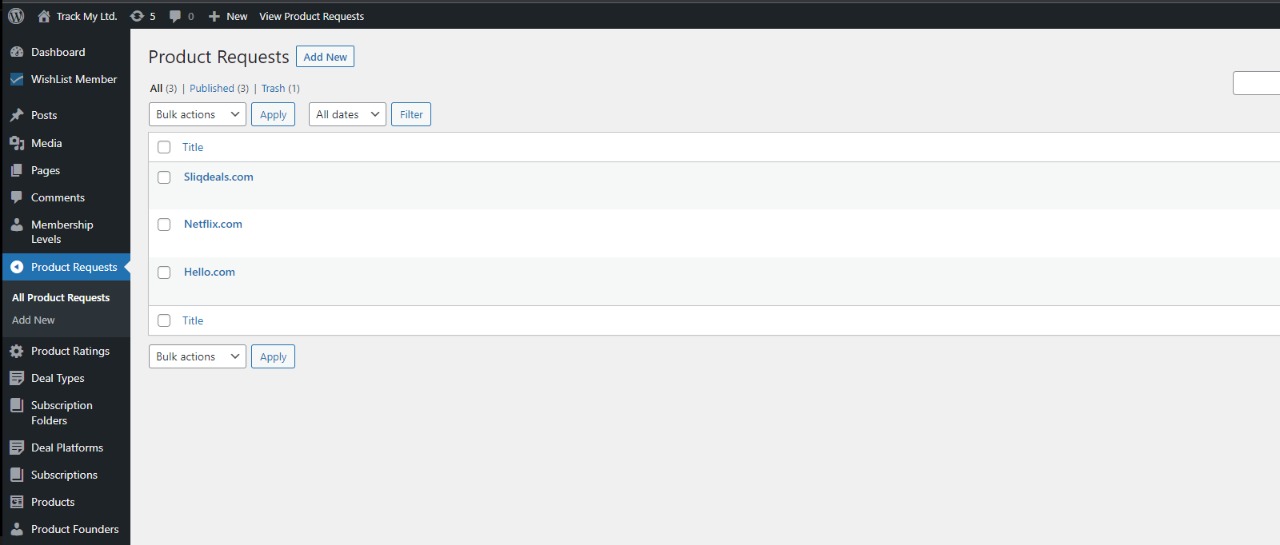
User can easily add subscription on our system.



**Figure 6.14. User Subscription Add.**

**6.6.15 User Product Request**

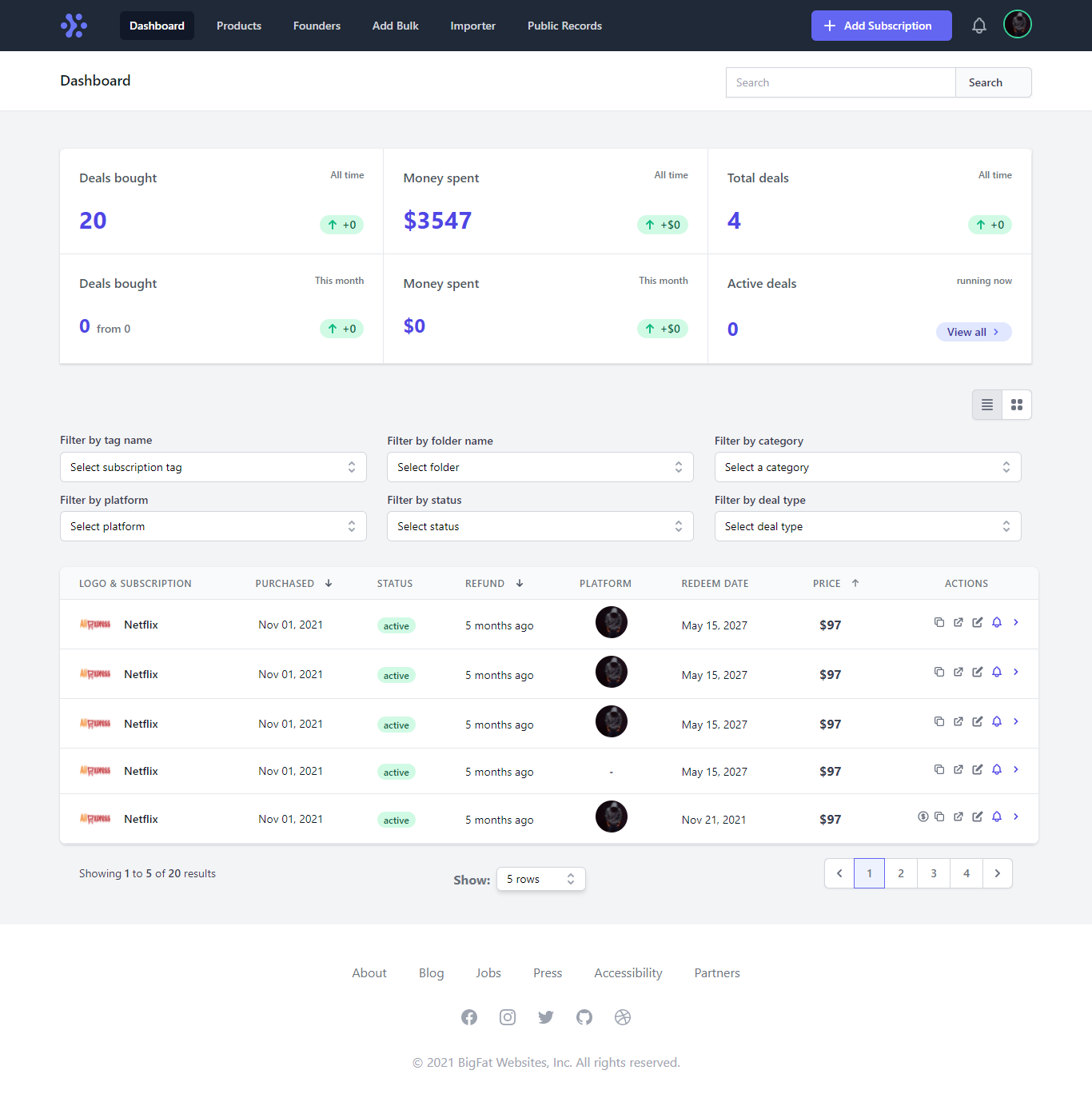
User can request product on our system.



**Figure 6.15. Product Request.**

**6.6.16 User Dashboard after Purchase**

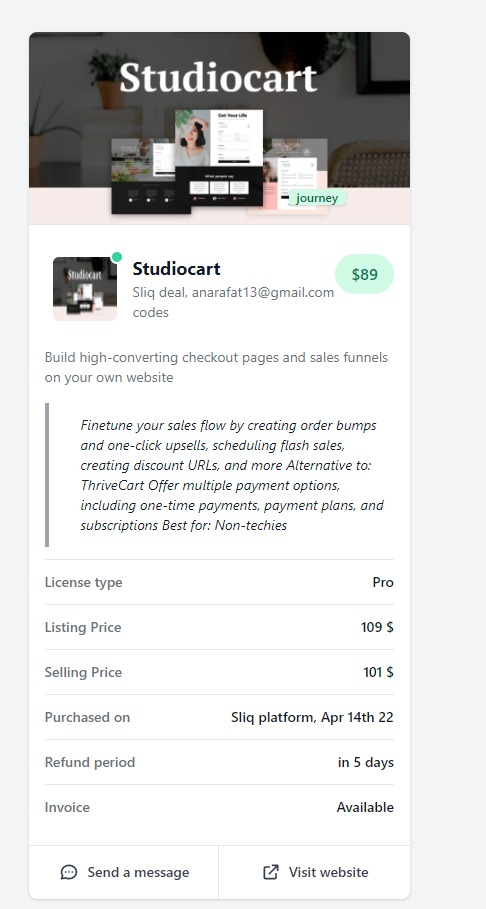
User can see updated dashboard on this system when they purchase a subscription.



**Figure 6.16. Dashboard after purchase.**

**6.6.17 Purchase Subscription Information**

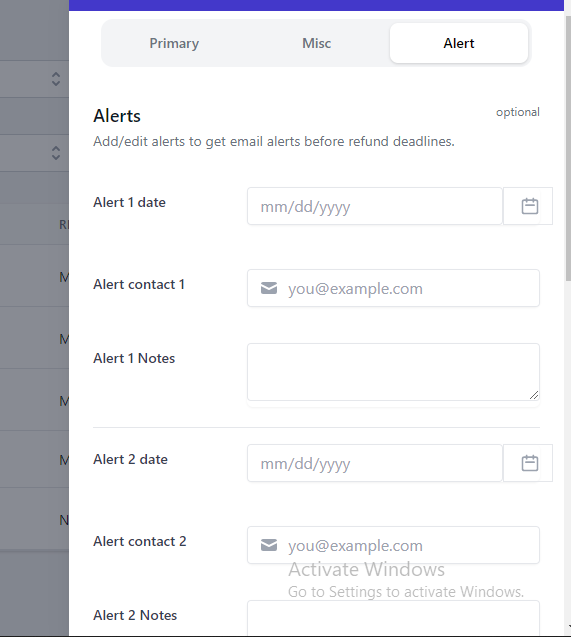
User can see purchase subscription information on this system.



**Figure 6.17. Purchase Subscription Information.**

**6.6.18 Subscription Alert**

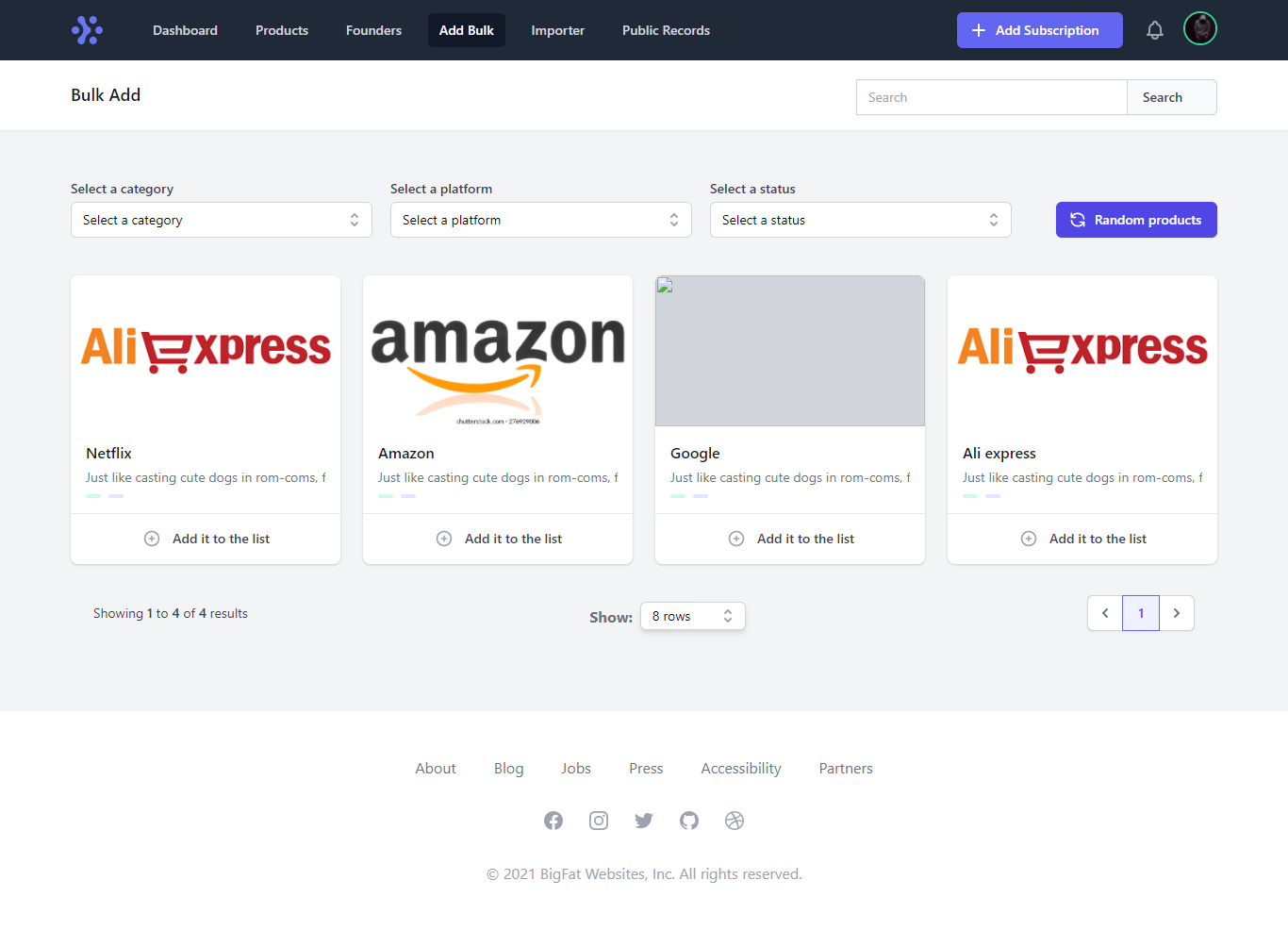
User will receive notification alert for subscription end period date on this system.



**Figure 6.18. Subscription alert.**

**6.6.19 Subscription Bulk Add**

User can add bulk subscription on this system.



**Figure 6.19. Subscription Bulk Add.**

**6.7 Summary**

The implementation of this project is discussed in this chapter. What technology did we utilize to create this website. We also go over each feature with screenshots.

# Introduction

**Chapter 7 DISCUSSION AND CONCLUSION**

After completing this project, we must state that we did our best to design this system in the most appropriate, useful, and user-friendly manner possible. We achieve the system's expected outcomes. The system is ready to use from anywhere, on any PC, and it can be accessed via the internet from any computer. Unauthorized users are well protected by us. Furthermore, because our system is user-friendly, we can claim that it has the best performance. Despite the fact that our technology has significant limitations, users will benefit from it. We will endeavor to overcome its restrictions in the future.

# Discussion

Our project is broken down into various sections. Introduction, literature study, and methodology, software requirements specification, normalization, and data dictionary, software design, and implementation are the items on the list.

We covered the project's goals, an overview of the current system, and a proposed system in the introduction.

We examined the software development life cycle and which software model is utilized in the system in the literature review and methodology.

We examined the software development life cycle and which software model is utilized in the system in the literature review and methodology.

We can learn about the requirements of a system that are needed by the end-user by looking into software requirements specifications.

We normalized several tables in the normalization and data dictionary section to handle data redundancy by reducing multiple value, partial dependency, and transitive dependency.

We've focused on the ER diagram, activity diagram, and user case diagram in software design.

We described the implementation tools that we used during the construction of our system in Implementation.

# Limitation of the system

Currently our project is for only desktop users. We didn't implement our project as a Single Page Application, so for this reason our page reloads after every action. There is no automatic system to change ownership of a subscription, User have to make it by communicating to each other. There is no panel to add deals by deals founder.

# Future work

1. We will add a mobile app for our project.
2. We will improve the UI.
3. We will make a system to stop recurring payments of any subscription from our system.

# Summary

In this chapter we discussed the limitations of our project and the future plan of our project.

# References

[1] <https://www.truebill.com/feature/manage-subscriptions> [Accessed on: 31 March, 2022].

[2] [https://money.usnews.com/money/personal-finance/saving-and-budgeting/articles/track-and- manage-subscriptions-with-these-apps](https://money.usnews.com/money/personal-finance/saving-and-budgeting/articles/track-and-%20%20%20%20%20manage-subscriptions-with-these-apps) [Accessed on: 31 March, 2022].

[3] [https://appsumo.com/deals/subscription-tracker-and-management-notion/](https://appsumo.com/products/subscription-tracker-and-management-notion/) [Accessed on: 31 March, 2022].

[4] <https://www.guru99.com/what-is-php-first-php-program.html> [Accessed on: 1 April, 2022].

[5] <https://www.techtarget.com/searchcontentmanagement/definition/WordPress> [Accessed on: 1 April, 2022].

[6] [https://community.codenewbie.org/lanrewaju/100daysofcode-chapter-1-html-and-css-basics-3 709#:~:text=What%20is%20HTML%3F,for%20web%20 pages%20and%20applications](https://community.codenewbie.org/lanrewaju/100daysofcode-chapter-1-html-and-css-basics-3709#%3A~%3Atext%3DWhat%20is%20HTML%3F%2Cfor%20web%20pages%20and%20applications). [Accessed on: 1 April, 2022].

[7] <https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS> [Accessed on: 1 April, 2022].

[8] <https://en.wikipedia.org/wiki/JavaScript> [Accessed on: 1 April, 2022].

[9] <https://v2.vuejs.org/v2/guide/?redirect=true> [Accessed on: 1 April, 2022].

[10] [https://www.siteground.com/tutorials/php-mysql/mysql/#:~:text=MySQL%20is%20a%20fre ely%20available,ease%20and%20flexibility%20of%20use](https://www.siteground.com/tutorials/php-mysql/mysql/#%3A~%3Atext%3DMySQL%20is%20a%20freely%20available%2Cease%20and%20flexibility%20of%20use) [Accessed on: 1 April, 2022].

[11] [https://www.geeksforgeeks.org/introduction-to-tailwind- css/#:~:text=Tailwind%20CSS%20is%20basically%20a,have%20to%20fight%20to%20override](https://www.geeksforgeeks.org/introduction-to-tailwind-%20%20%20css/#:~:text=Tailwind%20CSS%20is%20basically%20a,have%20to%20fight%20to%20override) [Accessed on: 1 April, 2022].

[12] <https://www.toppr.com/ask/question/web-server/> [Accessed on: 1 April, 2022].

[13] <https://www.wpbeginner.com/glossary/apache> [Accessed on: 1 April, 2022].

[14] <https://www.webopedia.com/definitions/browser/> [Accessed on: 1 April, 2022].

[15] <https://code.visualstudio.com/docs> [Accessed on: 1 April, 2022].

[16] <https://en.wikipedia.org/wiki/XAMPP> [Accessed on: 1 April, 2022].