**Report**

1. **Requirements for Phase 1:**

Scope of this phase is unchanged. Below are the requirements in Phase1.

**Functional requirements:**

Functional requirements are product features that focus on user demands and describe what a software system must accomplish and how it must work. It is a summary of the service that the software must provide. It refers to a software system or a component of one. Here in our application, we offer a separate dashboard for admin, designer and user, and the functional requirements are:

**Admin:**

* Login: Admin is provided with a facility to log into the application using the credentials that are registered with our application.
* Accept Designer: Admin is given access to accept requests from the newly registered designers with the application.
* View Users: Admin can view registered users where admin will verify details like their id, status and subscription plan.
* Add plans: Admin can add plans for the designer like a silver plan for three months subscriptions, gold for six months and diamond for an annual subscription.
* Manage Subscriptions: Admin has the feature to manage subscriptions on request from the registered users.
* View Designer: Admin will be able to view details like status, plan and the id of the registered Designers.

**External Interface Requirements:**

**Admin Dashboard:**

In Admin Dashboard, he will have access to different functionalities like login to dashboard, Accept Designers requests, View Designers, user’s profiles, and can manage subscriptions. Admin can add plans like in our application we provide three subscription plans silver for three months, gold for six months and diamond for annual subscription.

**Hardware Requirements:**

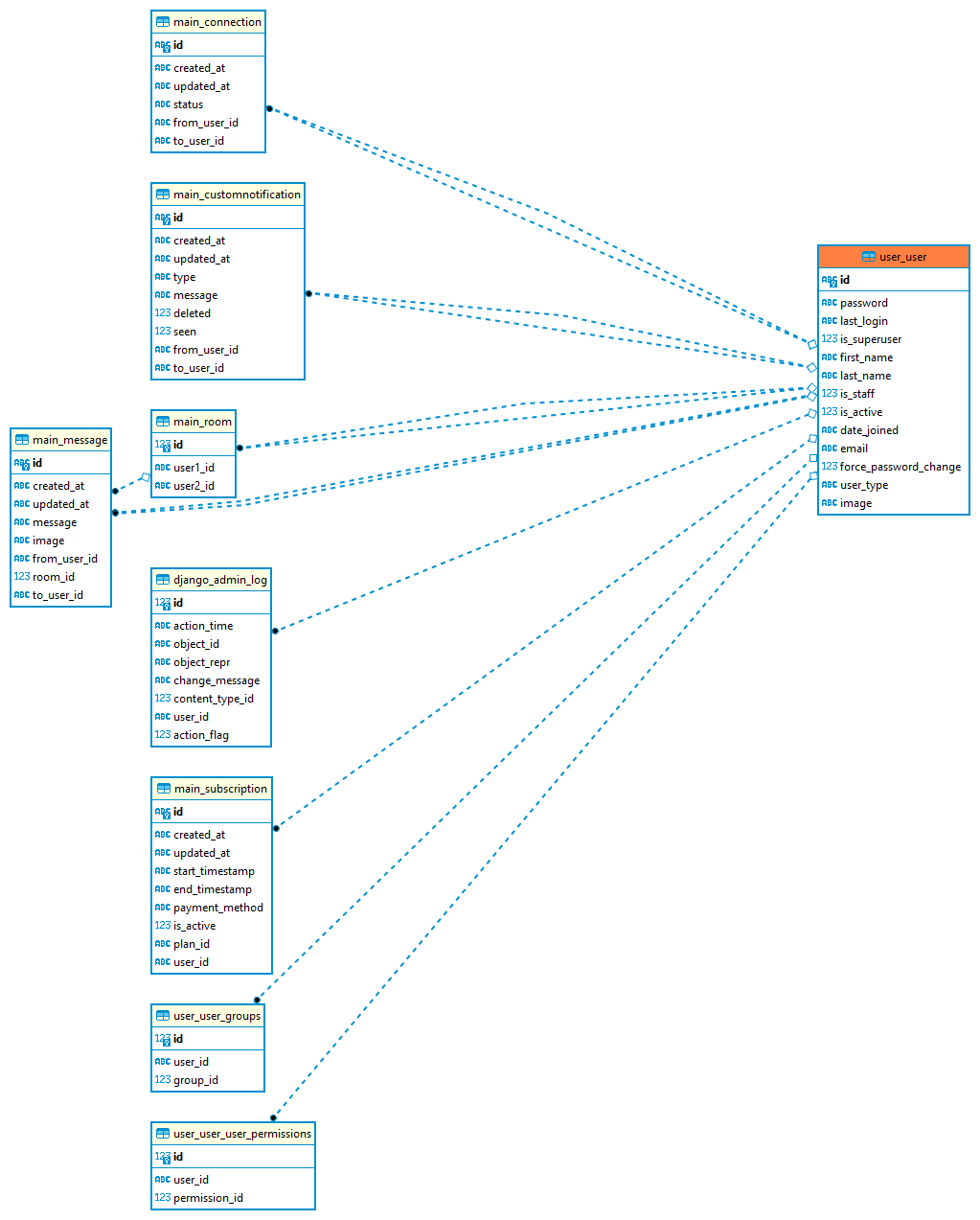
* Processor                                       :          Intel Pentium(R) Dual-Core Processor
* Speed                                             :          2.9 GHz
* RAM                                             :           6 GB RAM
* Hard Disk                                     :            40GB

**Software Requirements:**

* Operating System                         :         Ubuntu 18 Lts
* Browser                                        :          Chrome/Firefox
* Front End                                     :          HTML,CSS,JavaScript,Bootstrap5
* Back End                                     :           Python, Django
* Database                                      :           SQLite

1. **UML design for phase 1. You must include the following diagrams:**

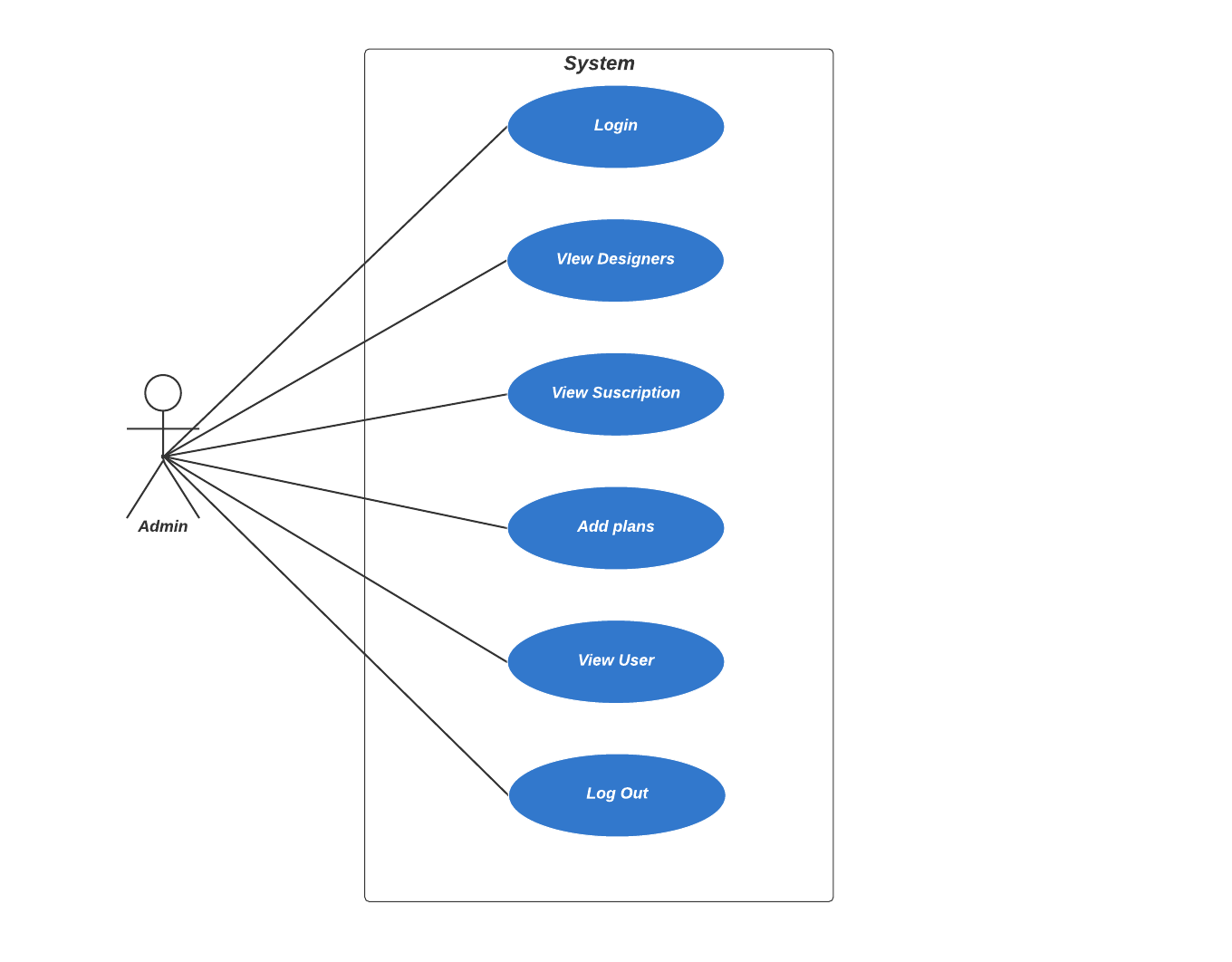
**UML Diagram:**

****

**Fig.1 Class Diagram**

**Use Case Diagram:**

A use case diagram is used in UML to identify system functionality during the analysis phase. It defines how users and external devices interact with the system under development. It does not go into great detail. Instead, it focuses on the connections between use cases, actors, and systems. A use case diagram usually comprises four components. Actors, systems, use cases, and relationships are all examples. The actors are anyone or anything which interacts with the system.

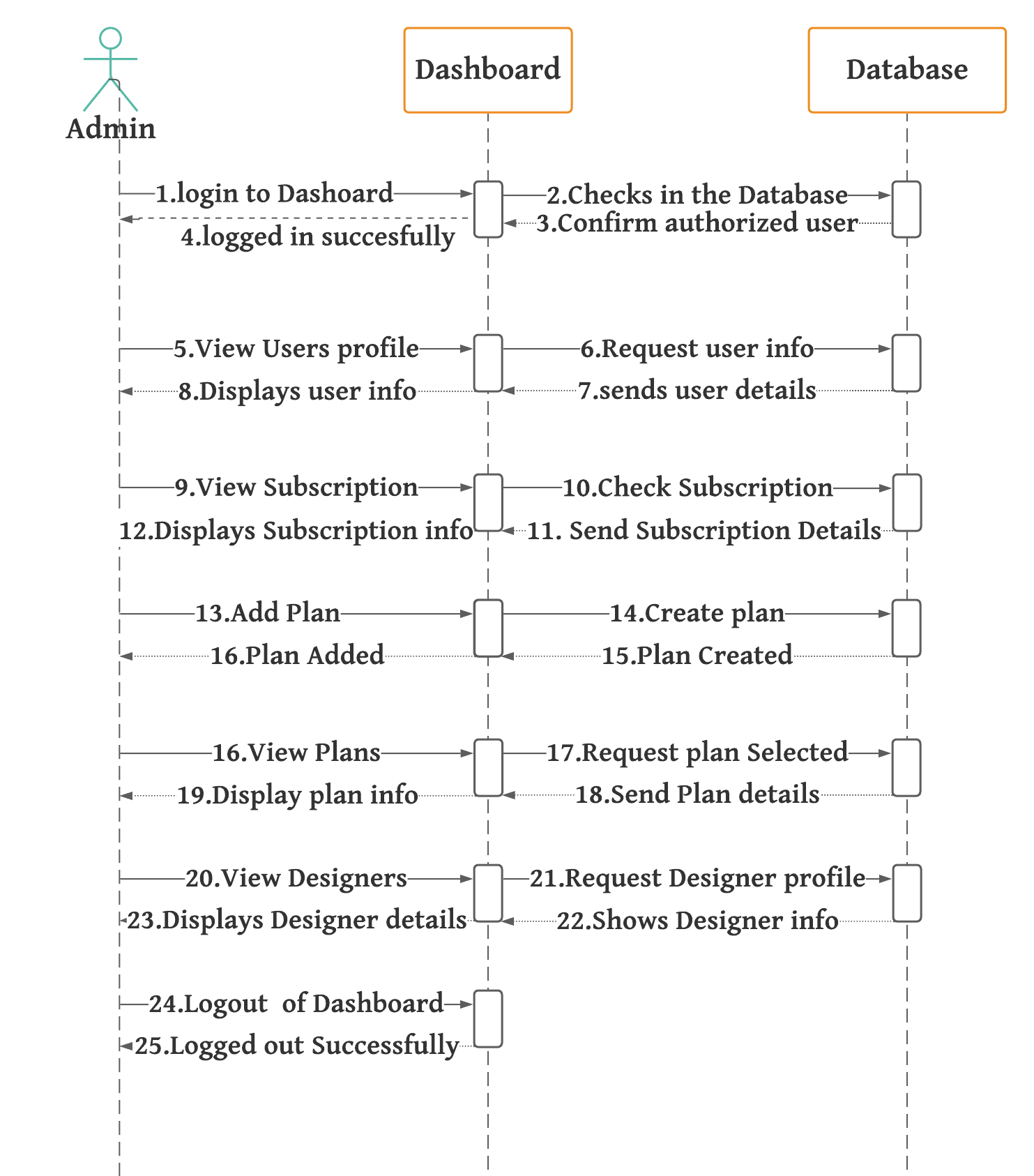


**Fig.2 Use Case Diagram**

**Description:** In the above use case diagram, the actors in the diagram are Admin who manages the application like adding plans, viewing subscriptions, and checking the status of the registered users.

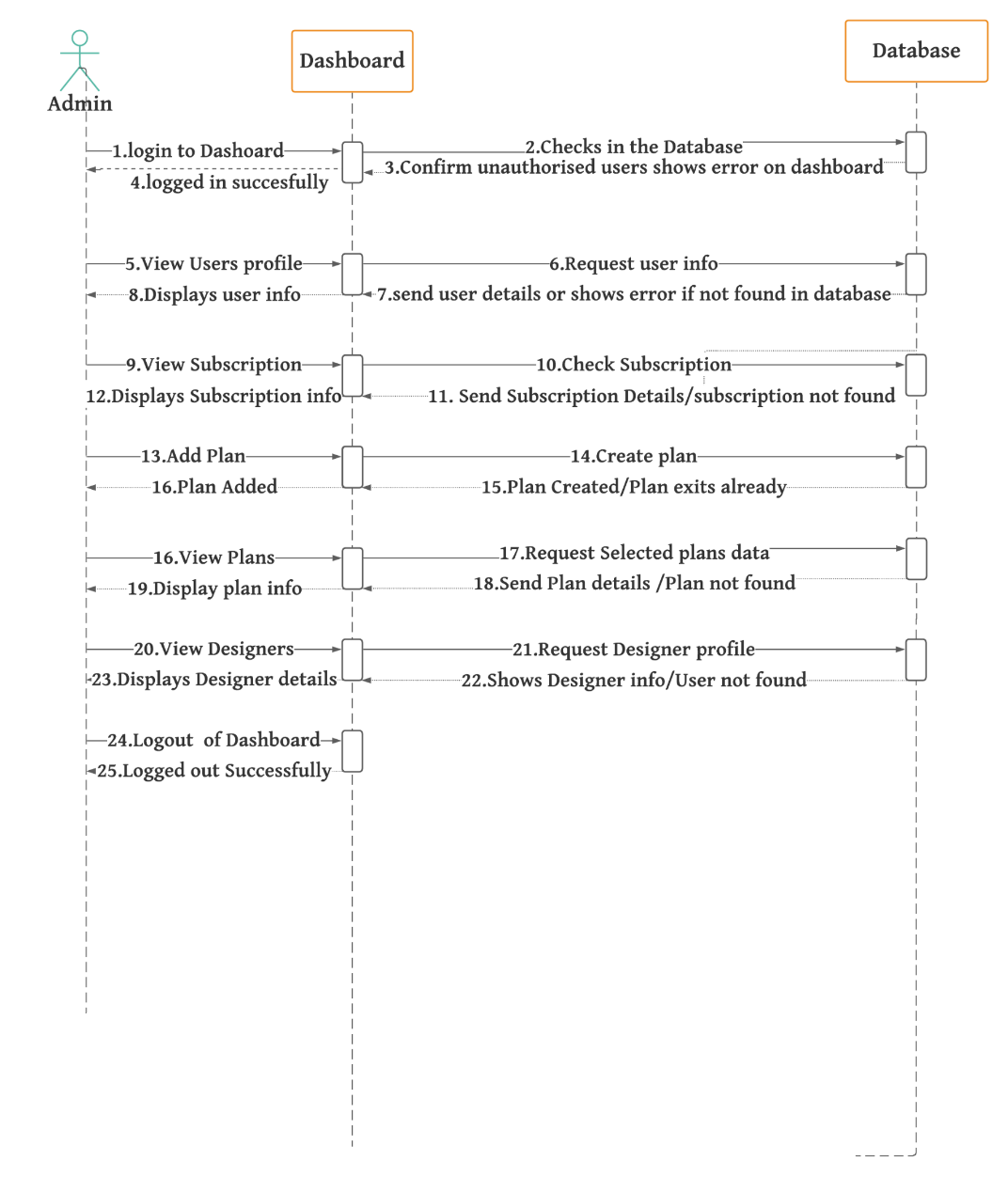
**Sequence Diagram:**

In UML, the analysis and design phases are represented by a sequence diagram. It is an interaction diagram that shows how operations are performed. A sequence diagram is frequently used to show a use case is chronologically structured event flow. It does an excellent way of highlighting how objects communicate with one another and what messages activate those communications.



**Fig.3 Sequence Diagram**

**Description:** The above sequence diagram depicts the orderly execution of actions between the admin dashboard and the database.



**Fig.4** Error Sequence Diagram

**Description:** The above sequence diagram depicts the orderly execution of actions between the admin dashboard and the database.

**C. Test cases for Phase 1:**

**Testing:** Testing is a process of determining whether the actual software product meets the expected requirements and is defect-free. It entails evaluating properties of interest by executing software/system components using human or automated techniques. Compared to actual requirements, software testing is used to find mistakes, gaps, and missing requirements. Program testing is critical because it allows any defects or errors in the software to be found and fixed before the final product is delivered. Software that has been thoroughly tested ensures dependability, security, and good performance, saving time and money.

**Test Case 1:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement\_id** | **Ticket\_**  **id** | **Requirement\_descrption** | **Expected Output** | **Actual Output** | **Requirement\_ticket\_status** |
| 1 | 11 | Admin tries to log in to the dashboard | Logged in successfully | Improper credentials and gives an error message user Not found | Failure |
| 1 | 12 | Admin login to the dashboard with proper Credentials | logged in successfully | Logged in to dashboard successfully | Success |
| 1 | 13 | After successful login Home Page will be displayed | admin logs in to the dashboard, the home page appears. | Home page displayed successfully | Success |
| 1 | 14 | Admin tries to view Users | User profiles will be displayed | User profile displayed successfully | Success |

**Description:** The authentication of the admin is tested in the preceding test case, and only the authorized user with appropriate credentials is allowed to enter the dashboard.

**Test Case 2:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement\_id** | **Ticket\_**  **id** | **Requirement\_descrption** | **Expected Output** | **Actual Output** | **Requirement\_ticket\_status** |
| 2 | 21 | Admin tries to access plan to check plan details | List of plans with validity and price are displayed | Displays list of plans with its details | Success |
| 2 | 22 | Admin tries to update the plans | The plan will be updated | Invalid operation admin doesn’t have access to update plan | Failure |
| 2 | 23 | Admin tries to add plans | Request to add plans is accepted and admin adds plan | Admin adds plan to users | Success |

**Description:** The admin’s access to manage plans is tested in the preceding test case, and only the limited operations are enabled for the Admin to protect the integrity of data.

**Test Case 3:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement\_id** | **Ticket\_**  **id** | **Requirement\_descrption** | **Expected Output** | **Actual Output** | **Requirement\_ticket\_status** |
| 3 | 31 | Admin views subscriptions | Subscriptions info is displayed on the dashboard | The dashboard shows subscriptions data | Success |
| 3 | 32 | Admin tries to update subscription | Subscription will be updated | Invalid operation admin cannot change subscription | Failure |

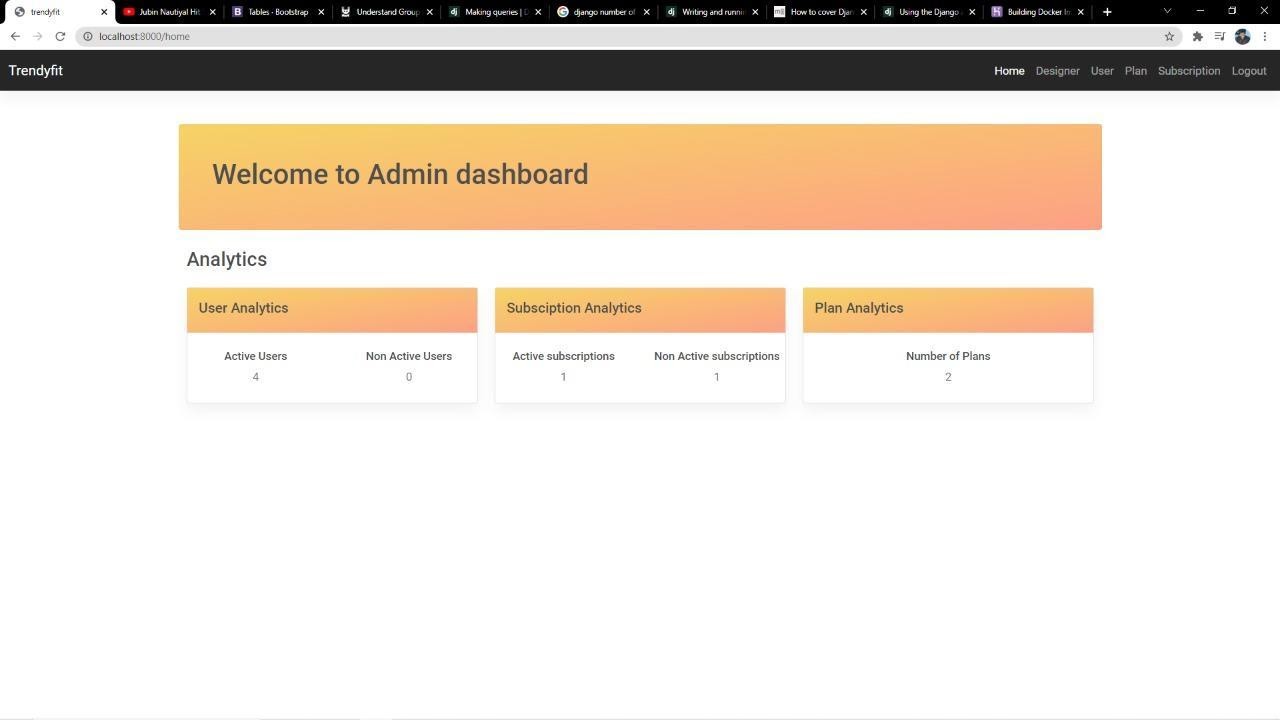
**Description:** The admin’s access to view and manage subscriptions is tested in the preceding test case, and only the limited operations are enabled for the Admin to protect the integrity of users data.

**d. A user manual that tells us how to install/use your program. This is meant for the end-user of the software. You may include screen shots, where appropriate.**

**User Manual:**

**Step 1:** To use our application, users must have access to the internet and a browser such as Chrome or Firefox. Admins can access the Admin Dashboard by visiting the login page and entering credentials (Mail Id and Password) that have been stored in our database.

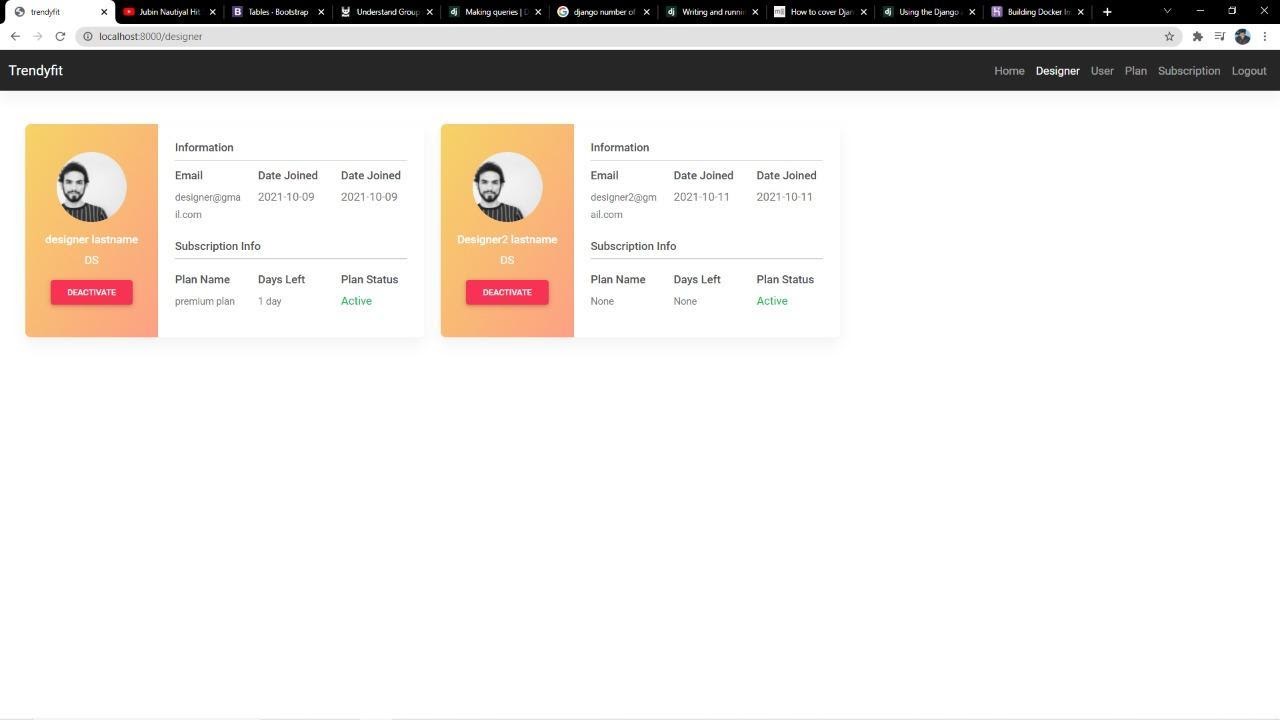
**Step 2:** Admin should be able to see a home page in their dashboard if the credentials entered are correct to check the number of active users, active subscriptions, and available plans.



**Fig.5 Home Page**

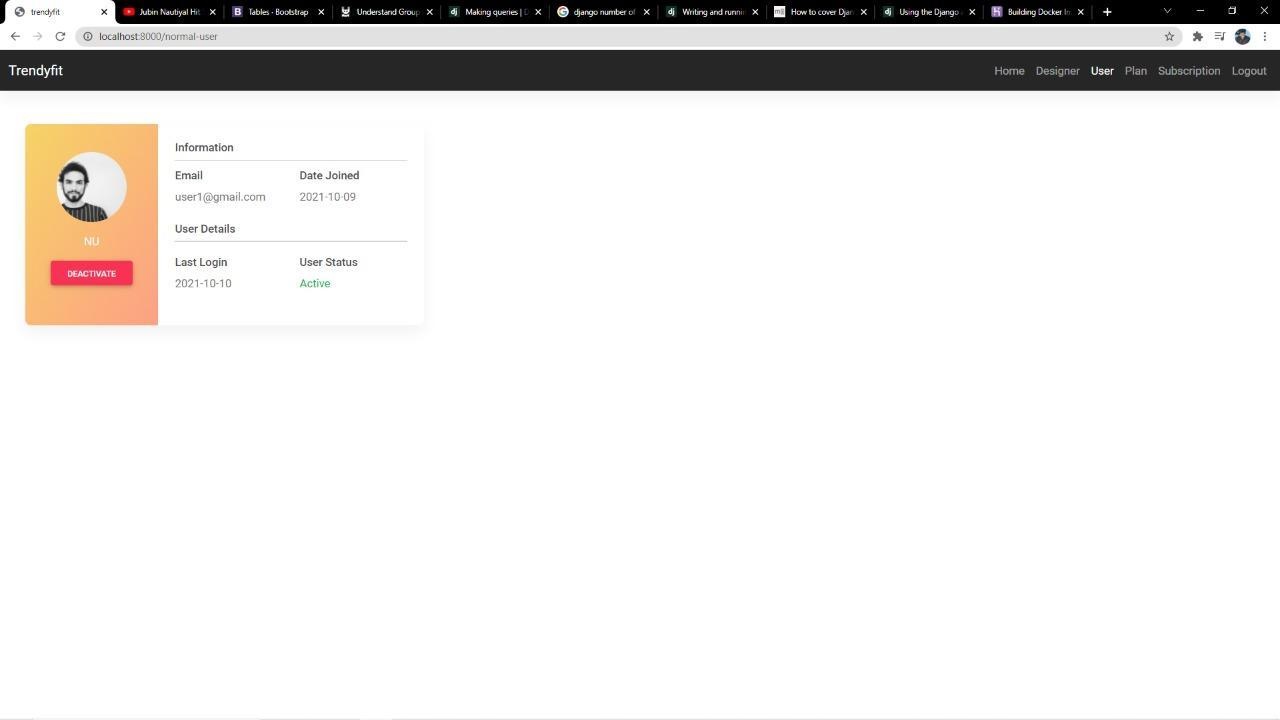
**Step 3:** There are several options for the user to explore in our application, including Designer, User, Plan, and Subscription, which can be found in the right-hand navigation bar.

**Step 4:** Select Designer to navigate to a page where you can see a list of all the Designers who have registered with the application and their information such as the Designer's email address, registration date, and subscription details, plan name, days left, and plan status.



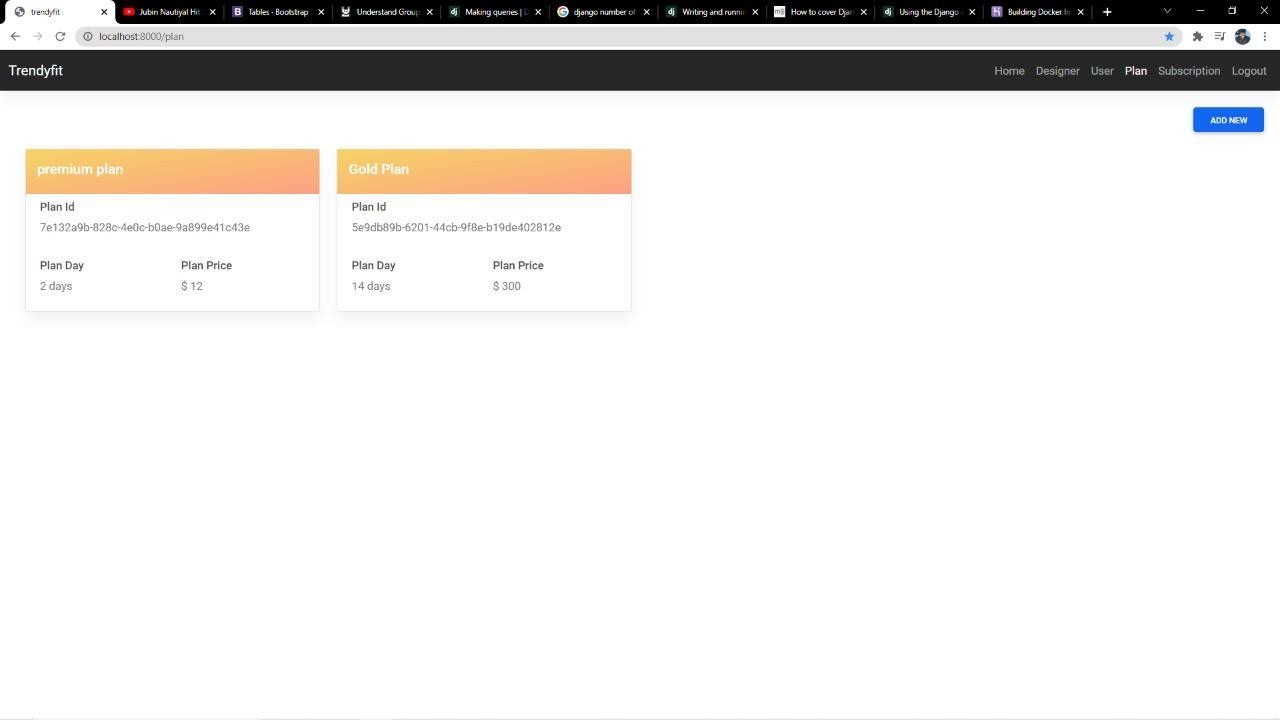
**Fig 6. Designer Page**

**Step 5:** Click on User to see the user's information, such as the email address they registered with, the date they joined, and other details such as their last login and status.



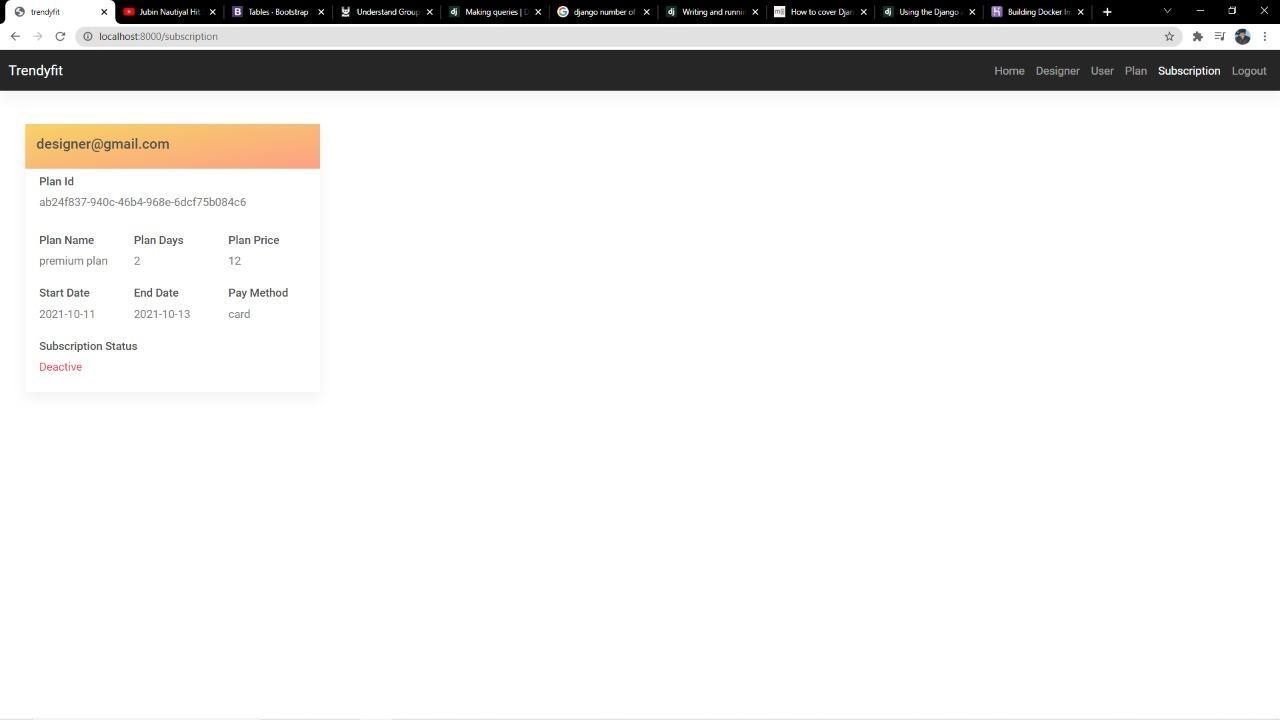
**Fig 7. User Page**

**Step 6:** Click Plan to view the available plans and their details, such as plan type, plan id, plan validity, and plan pricing.



**Fig 8. Plan Page**

**Step 7:** Select Subscription from the Navigation Bar to access subscription details such as UserId, plan id, plan name, Validity; Plan activated date and termination date, and Payment method.



**Fig 9. Subscription Page**

## **e. Clear instructions on how to compile/run both your program and your test cases (the program must compile/run).**

## **Instructions:**

The following are the 2 methods to setup our project

* using local run command
* using docker

Note: The following commands are for linux only

### Local Setup

The first thing to do is to clone the repository:

$ git clone https://github.com/SravanthiSilla/Mighty8.git

$ cd terndyfit

Create a virtual environment to install dependencies in and activate it:

$ python -m venv env

$ source env/bin/activate

Then install the dependencies:

(env)$ pip install -r requirements.txt

Note the (env) in front of the prompt. This indicates that this terminal session operates in a virtual environment set up by venv.

Once pip has finished downloading the dependencies:

(env)$ cd trendyfit

(env)$ python manage.py makemigrations

(env)$ python manage.py migrate

once you have migrated you need to

(env)$ cd trendyfit

(env)$ python manage.py runserver

or

(env)./manage.py runserver

And navigate to <http://127.0.0.1:8000/>.

**Docker Setup**

To execute the project in docker follow below commands. Make sure you have docker installed in your system or else please follow this link [click here](https://docs.docker.com/engine/install/)

$ cd terndyfit

$ docker-compose -f compose.yml up

**f. A section that briefly describes feedback received during the peer review session and actions taken based on the feedback.**

We have explained the overview and functionalities of our project to partner team and they have provided few suggestions. Explained all the functional, non-functional and external interface requirements.

**Features of the Project:**

* User Management (Login/Register)
* Admin
  + Admin manages designer and user
  + Admin can accept designer requests
  + Admin can view designers and users
  + Admin can provide plans and subscriptions to designer
* Designer
  + Designer can register and login to the application
  + View users
  + Designer can change password.
  + Also they can chat with users by providing suggestions related to fashion to the user.
* User
  + User can login to the application
  + They can view and chat with designers
  + Users have option to change their password in case of emergency
  + Users can provide feedback to designers

**Suggestions:**

Partner team told that all the requirements are defined properly and they suggested us to add feedback or ratings feature in the web application. Like customer can provide feedback to the designer.

**Actions Taken:**

We have accepted the suggestion provided by our partner team and will be adding feedback feature to our project. This feature will come under Phase 3 development which is User Dashboard.

**g. A brief reflection on what has been accomplished, what went well and could be improved.**

**Brief Reflection:**

Project phase one development has successfully completed we have implemented admin dashboard successfully feature that has been covered in the phase one development are listed below

* Admin login view
* Admin main dashboard view
* Admin designer view
* Admin user view
* Admin Plan view
* Admin Add plan view
* Admin Subscription View
* Admin logout

**Admin:**

The above features are implemented in the admin dashboard flow as part of the phase our target was to finish all the flow of the admin and there were successfully implemented

**Improvements:**

* A few Improvements can be done in the admin dashboard such as adding pagination which will not make perfect user experience, we will shift that to perfect scrolling in coming phases where user will not to click more on our dashboard that will improve our user experience
* Push notifications when a user/admin adds to our portal a push notification will be send to the admin so that he can keep an track of the new user at real time
* Filters can also be added to get the user/designer on search for an easy view of data.

**h. Member contribution table (should describe who wrote what components or classes of the system and what parts of the report). Add more rows as needed.**

|  |  |  |  |
| --- | --- | --- | --- |
| Member name | Contribution description | Overall Contribution (%) | Note  (If applicable) |
| Sravanthi | Added Designer, user and notifications, Member contribution table | 12.5 |  |
| Manish | Added user module, plan and subscription | 12.5 |  |
| Manohar | Added Login Page and basic layout of the application | 12.5 |  |
| Sharath | Added basic models and main models | 12.5 |  |
| Teja | Added test cases, Feedback from partner team during peer review | 12.5 |  |
| Harsha | Requirements for the projects, UI design, User manual | 12.5 |  |
| Praveen | Worked on UI/ UX designs, UML diagrams, Clear instructions | 12.5 |  |
| Yeshwanth | Added main module, Brief reflection on what has been accomplished, what went well and what had to be done part | 12.5 |  |