Software Design Specification

By CleanFrame

Section: C

Group Name: CleanFrame

Group Members:

- *Aditya Aggarwal – IIT2019210*
- *Divy Agrawal – IIT2019211*
- *Anmol Bansal – IIT2019218*
- *Ambika Singh Kaushik – IIB2019017*

Table of Contents

1. Introduction

Software Design Document provides documentation which would aid and serve as a basis in software development by providing the key implementation details which the software must comprise of. It also helps us to recognise the regulations with which the software must comply with. Software Design Document comprises various models and diagrams.
This documentation is created to identify the key aspects which would be playing a crucial role during the development phase of the application. And would help the developers to know how this application was developed.

1.1 Document Outline

This document involves all the technical aspects and overviews over all the diagrams needed to design the web-based application portal for internship applicants and company professionals to be aided and this process of recruitment to be streamlined and transparency to be increased.
This document involves all the design considerations, architectural strategies and System Architecture which was kept in mind while designing the software's features and outlines all the things kept in mind to benefit the user experience.
This document involves Data Flow diagrams over various levels for the ease of reading which tells how the data flows over the database and devices, Sequence Diagrams for understanding the sequence of flow of features that have been applied in the software and State diagrams to tell the various states the users/stakeholders would find themselves in while using the software, Entity-Relationship diagram for understanding how the database works behind the scenes.
In the later part of this document the explanation of various components of various diagrams and models have been given so that any developer is fully able to understand the diagrams and model. Also Pseudocodes have also been provided to give an overview of how these key components would be implemented.

1.2 Document Description

Later on the explanation of every feature and design is provided in the detailed description from the start of the documentation. Every design consideration has been documented here, the architectural constraints and strategies have been explained.
The diagrams have been explained in detail in the further portion of this document describing all the nuances and every possible query has been explained along the flow of this document.

1.3 Scope

We will describe what features are in the scope of the application and what will be out of scope of the application which is to be developed by CleanFrame.

- In Scope
  - Company/Students would be able to create their account in order to take part in the internship process.
  - Users of the application will be able to get technical support incase of any query.
  - Company Professionals will be able to create announcements regarding any vacancy or the results of the interview.
- Out of Scope
  - Selection process information or platform will not be available on the application as it would be varying from company to company.

- ➢ There will not be much interaction between the company and the Institute through this application.

2. Design Considerations

Design considerations part involves all the aspects where the design has been involved all across the software and solve the issues or give a general guidance towards solving those issues. And makes users comfortable with all the data manipulation done in the code of the software for the ease of use for the user.
Design has to be made user friendly in order to make every stakeholder comfortable with the software and use it to its full potential.

2.1 Assumptions and Dependencies

Here we shall be discussing the various assumptions made while developing the software design:

1. It is implied that the person using the software shall be equipped with a device with a stable internet connection.
2. The device shall be optimized for running web browsers.
3. It is recommended that every user operating the software should practise ethical ways for providing information over the portal to maintain trust between the stakeholders.
4. Windows xp and above, android 6 and above,etc. Operating system is recommended for a smooth experience.

The dependencies over which the software lays it's foundation are implicit but are very important to be kept in mind:

1. The notifications shall be checked from time to time to keep track of them over the portal.
2. The technical support can take some time to resolve the issue and patience is demanded from the reporter over such an unfortunate event.
3. Permanent ban shall be done to the users not following the instructions over a defined period of time.

2.2 User Interface Issues

User interface has been defined keeping in mind the needs of both the parties involved and following are the key highlights:

1. Company Reviews and Blogs has been created for applicants to get a better glance at the past records of the companies enlisted.
2. A notification center has been created for applicants to get notified and a nice and tidy way for company professionals to post them.
3. A technical support system has been created for the help of both.

Possible User interface Issues and their addressal:

1. Suppose a User 'A' is new to the environment, so a tutorial would be posted at the introduction page for curbing that issue.
2. Suppose a User 'B' is facing an issue already asked by some other user, then this issue can be solved from a blog posted by the technical support.

2.3 General Constraints

1. Users can't login/ create accounts without indian number and google verified email.
2. Once banned can appeal for an unban within 2 days of ban. After 2 days, an unbanned request can't be made.

2.4 Goals and Guidelines

1. User must be used to the dashboard as he is using his mobile phone.
2. Login / Signup System should be as simple as possible.
3. In continuation to above point, this system must be fully secured
4. New updates in a few months with different themes and more feasibility.
5. In the new update the ban removal appeal deadline to extend (at least upto 1 week).

3. Architectural Strategies

Basically the system architecture depends on the various API's used while dealing with the project code. Basic API's used in the project:

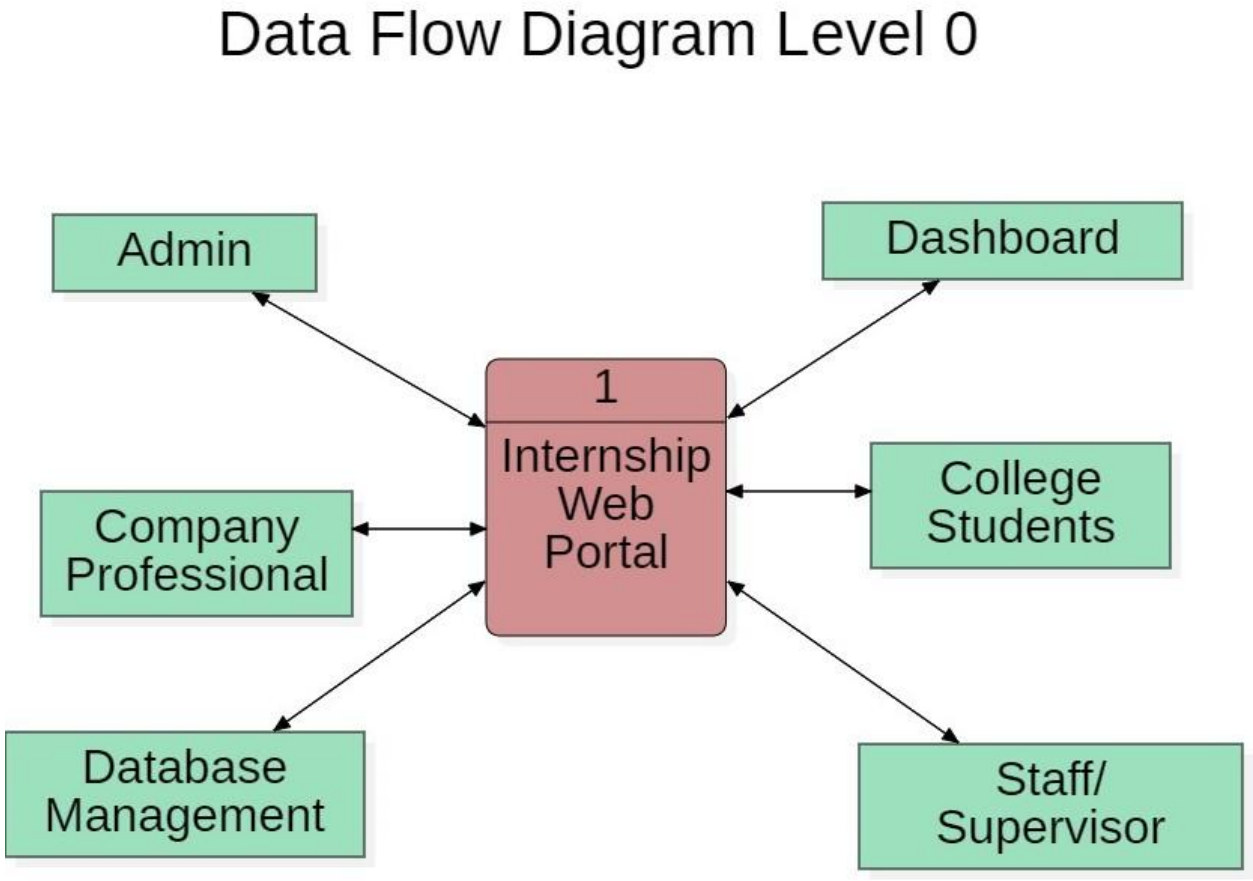
- ➢ Using Twilio API for sending OTP to phone number
- ➢ Using Django SEND MAIL API for sending OTP to emails.
- ➢ Using Rest Framework to create our own API's

Now since we are using some API so the user needs the following to create a account:

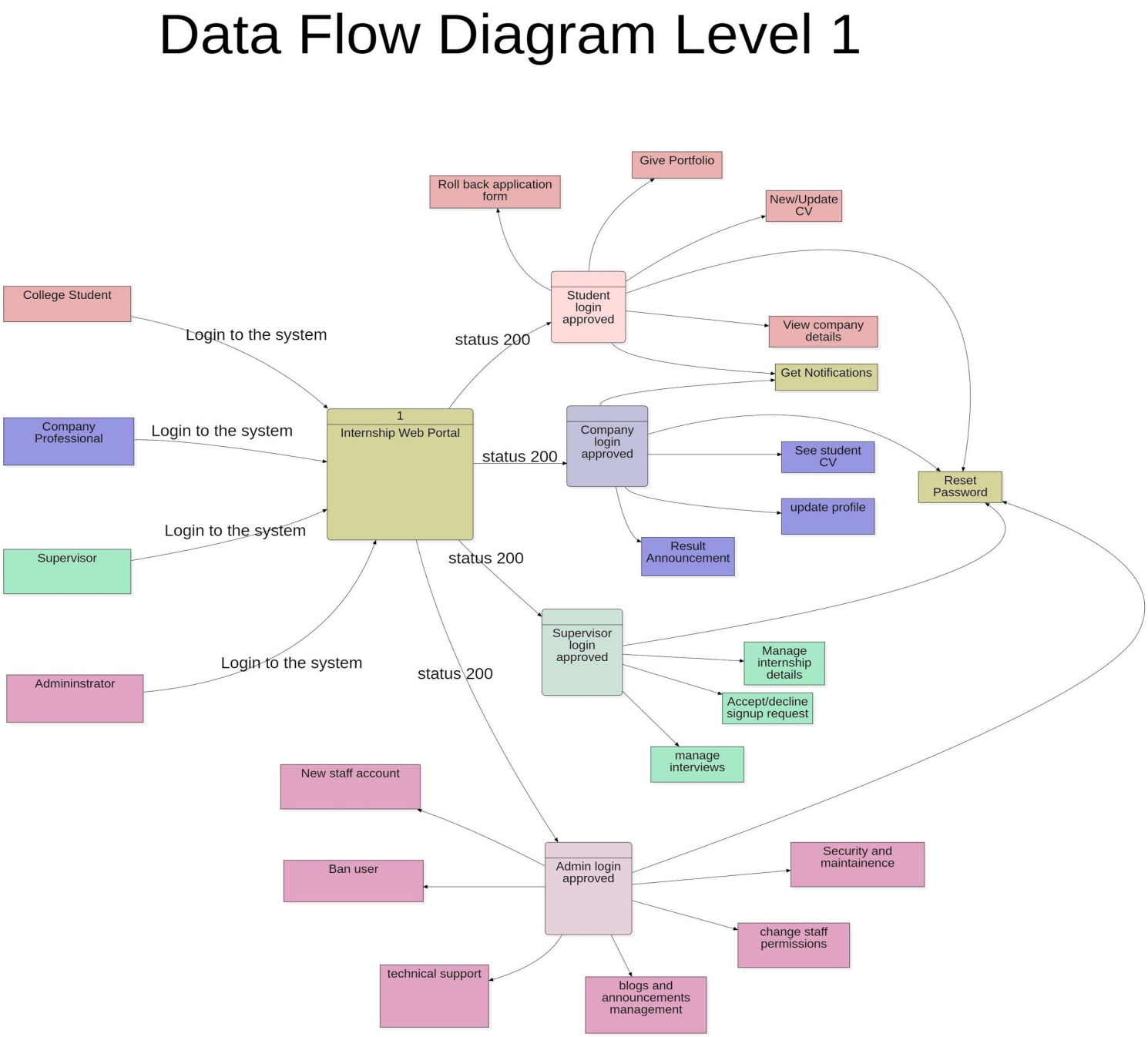
- ➢ Must have a valid Indian Phone Number
- ➢ Must have an google verified email address
- ➢ If email is provided with the IIITA then a student is allowed to create a student account.
- ➢ In continuation to the above point, no other gmail account or account related to another college should be allowed to register as a student.

4. Data Flow Model

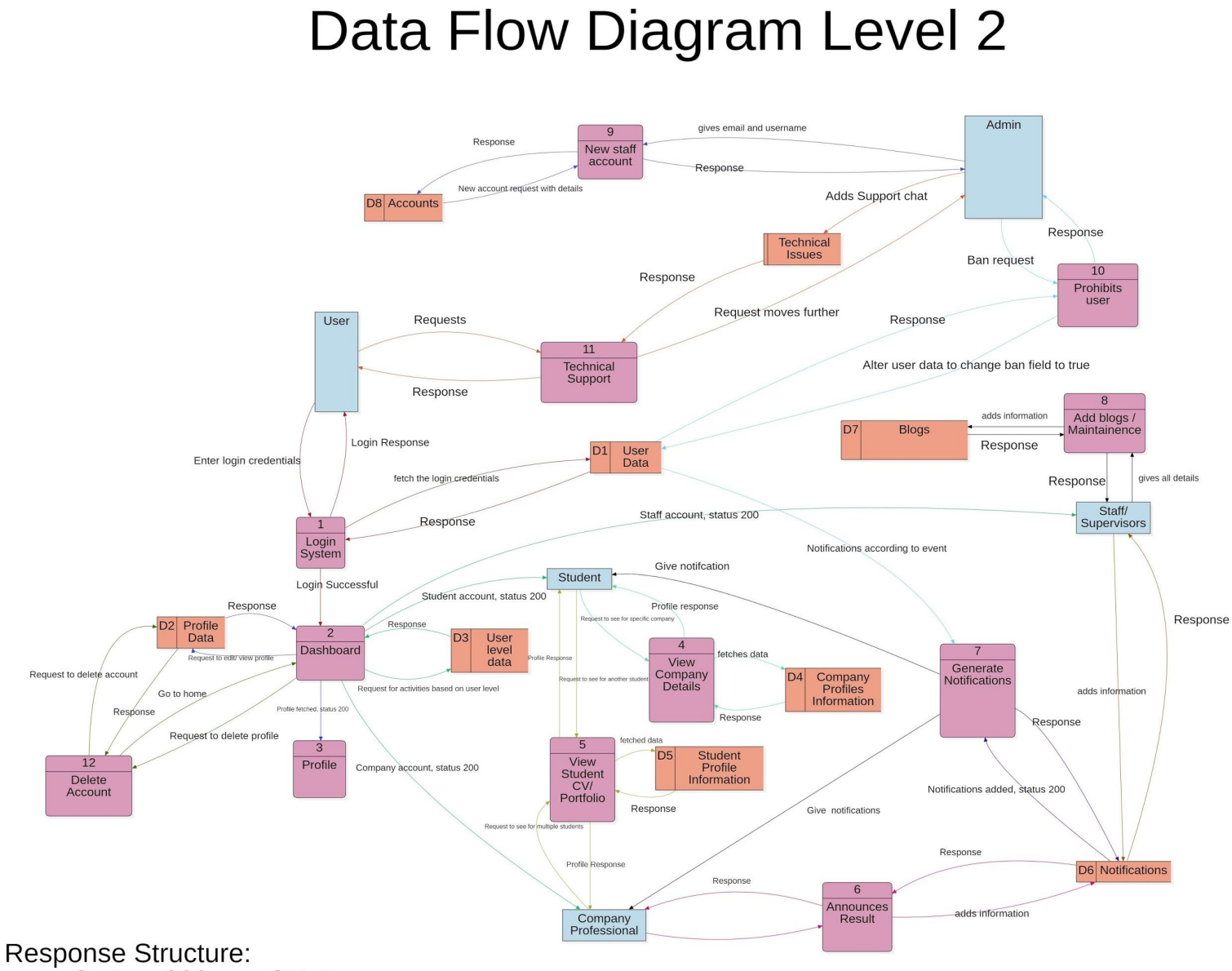
4.1 Level 0 Data Model



4.2 Level 1 Data Model



4.3 Level 2 Data Model



Response Structure:

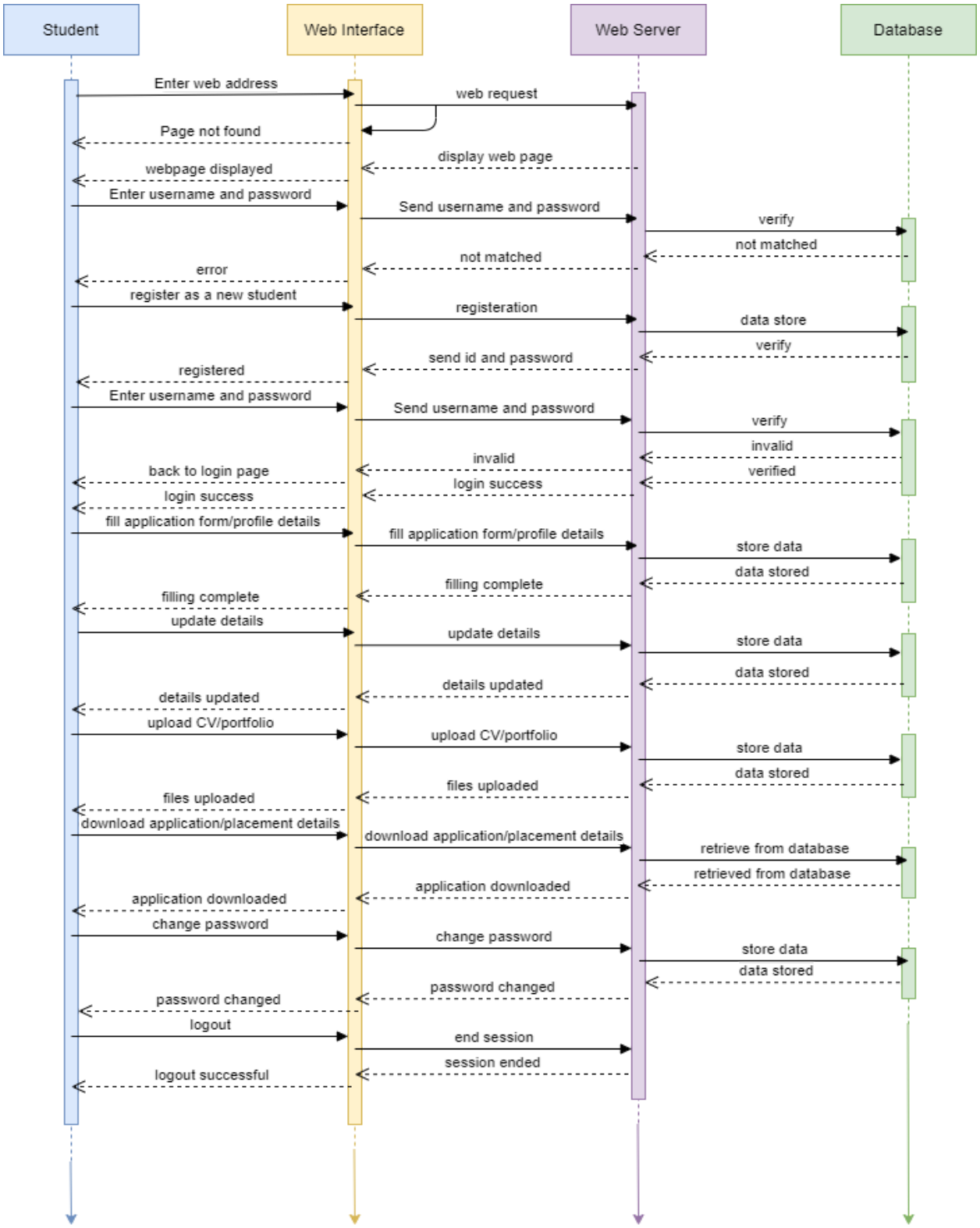
Status: 200 ---> OK, True

Status: 400 ---> No, False

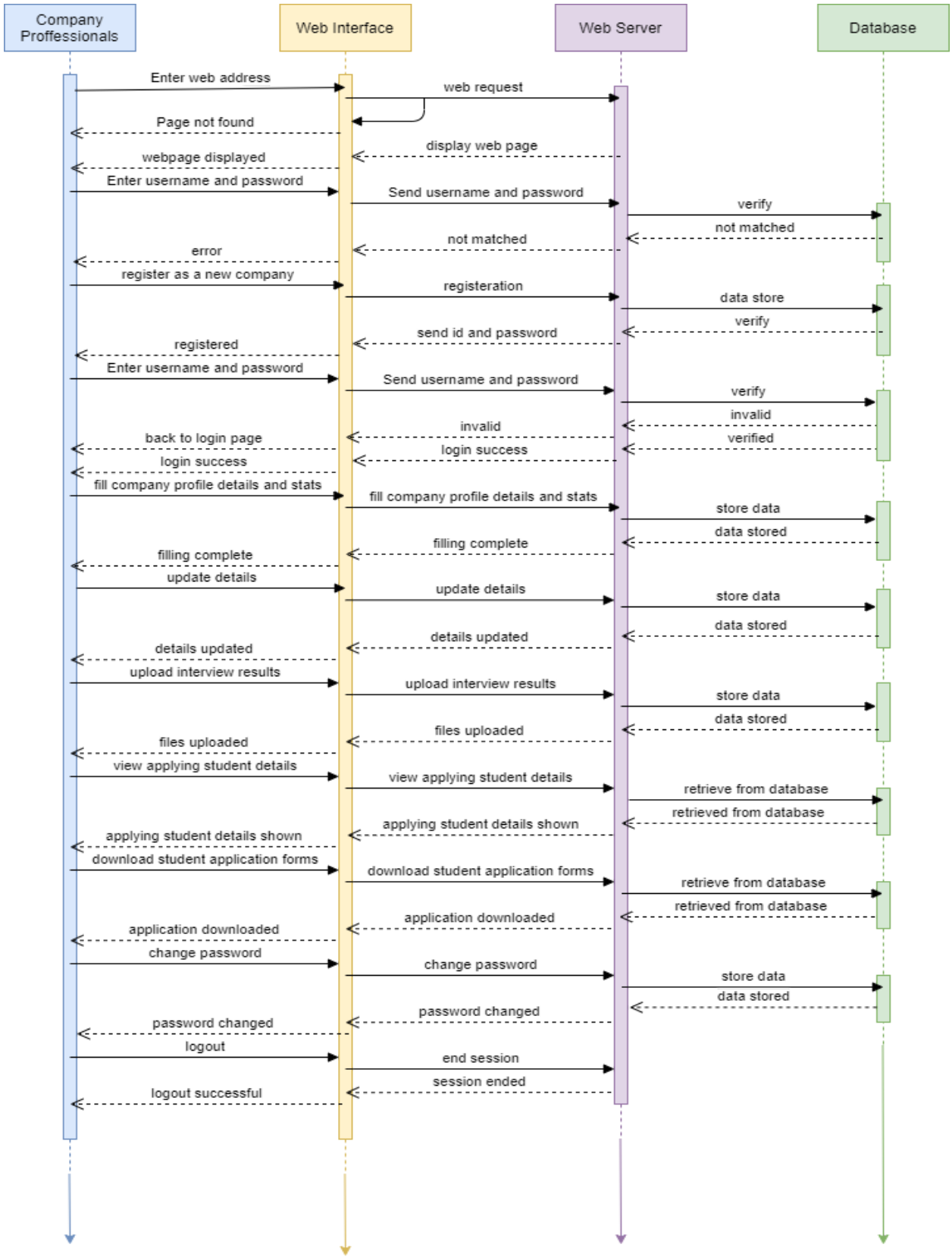
Status: 404 ----> Error Encountered

5. Sequence Diagram

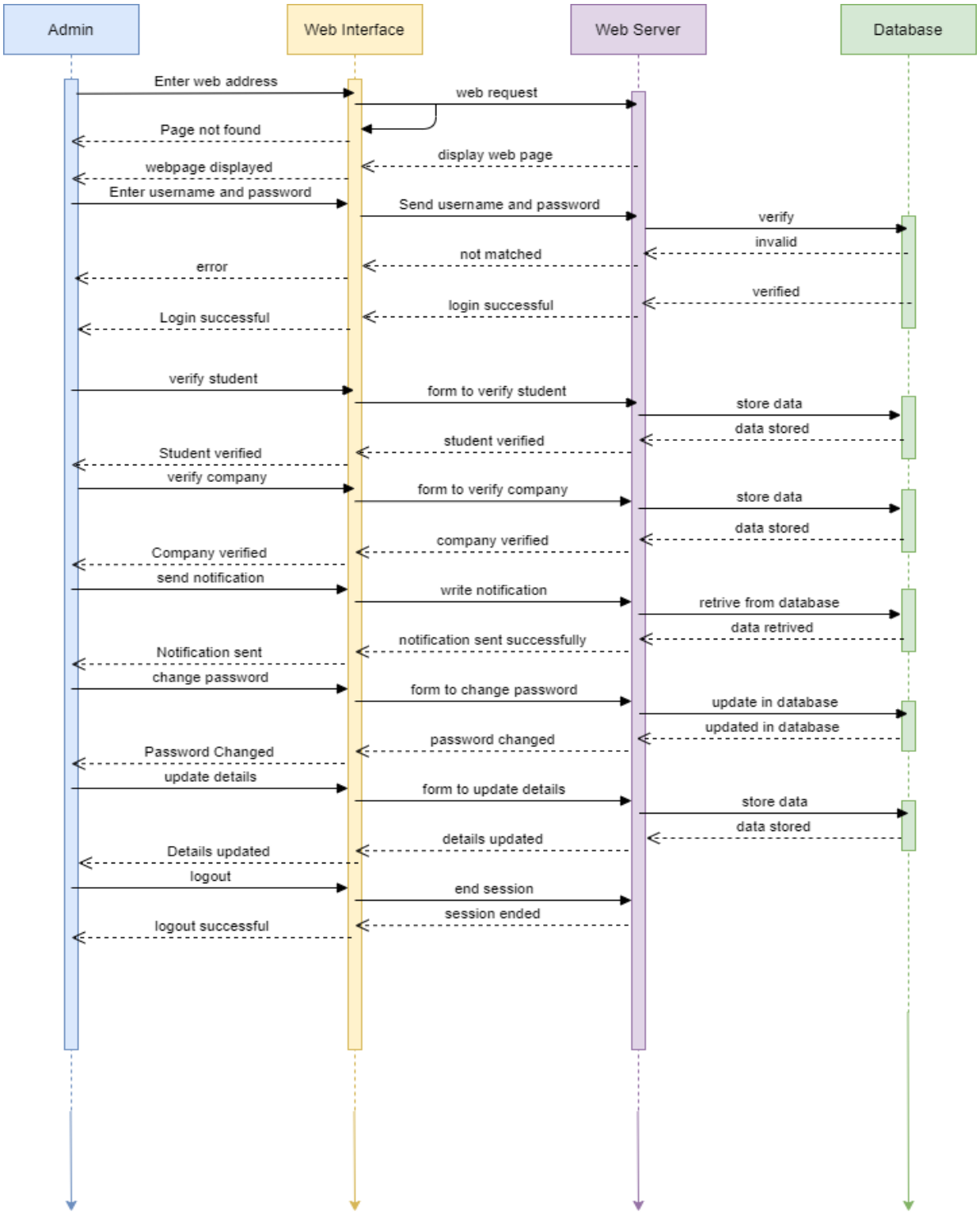
5.1 Sequence Diagram for Student



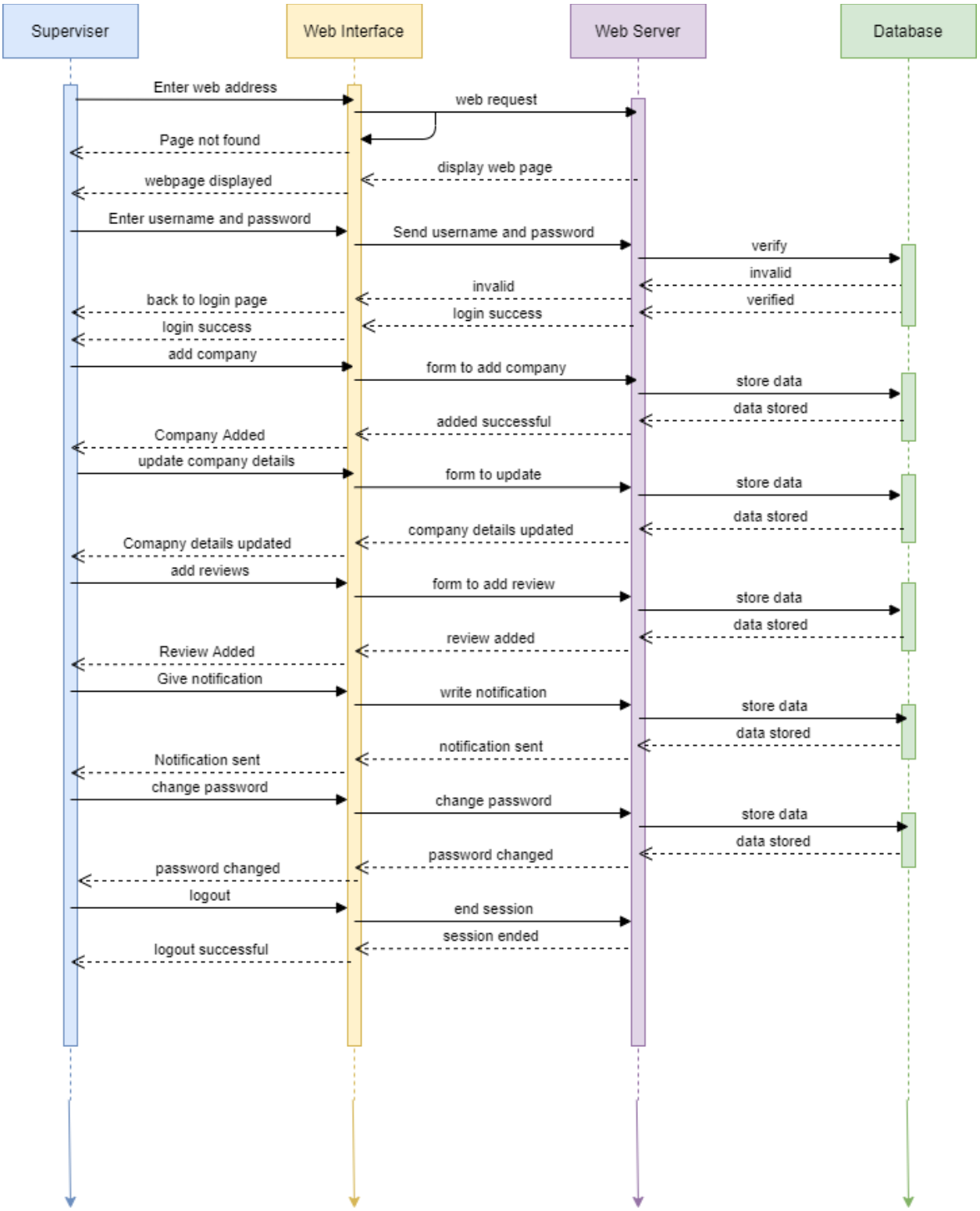
5.2 Sequence Diagram For Company Professional



5.3 Sequence Diagram For Administrator

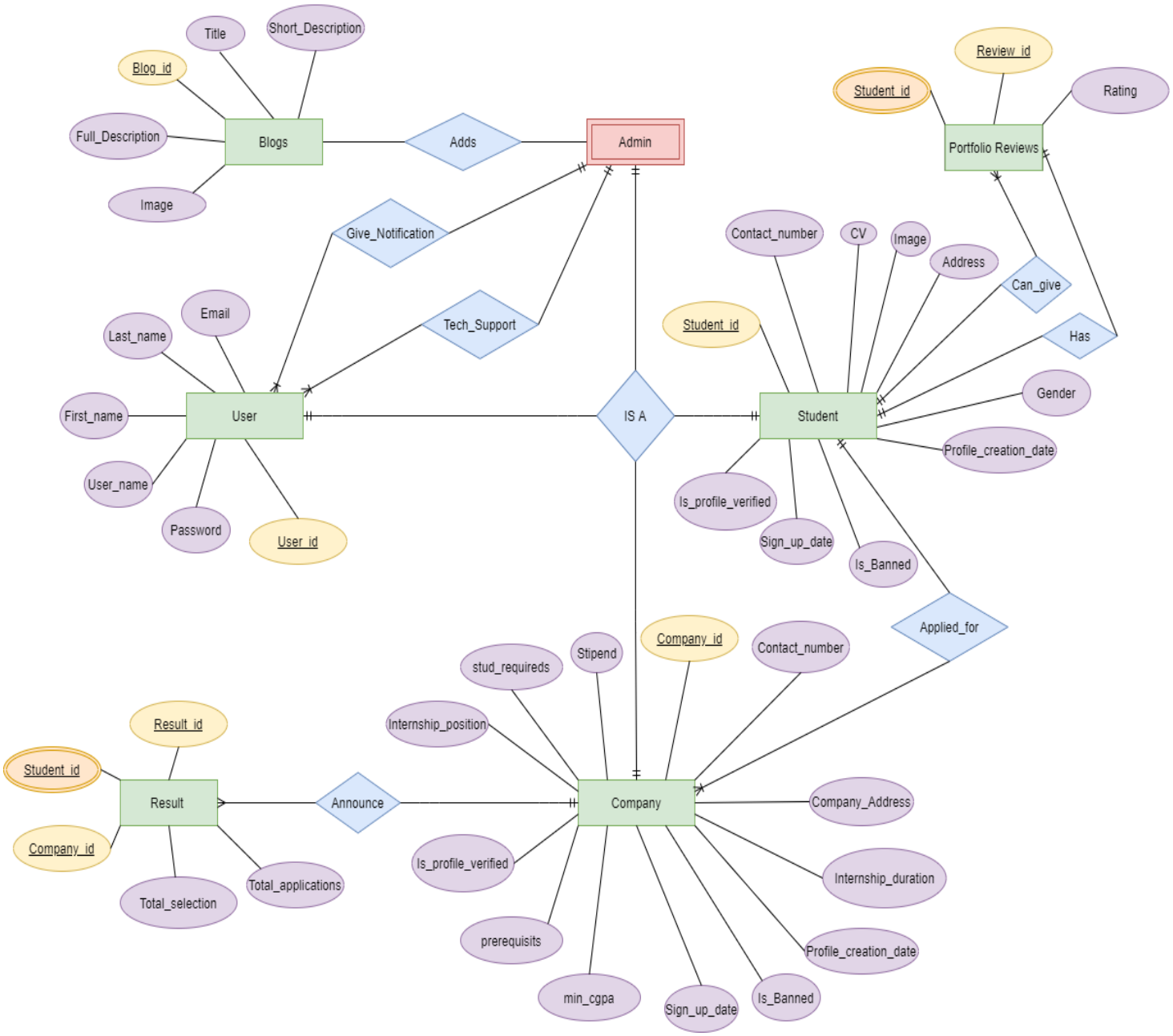


5.4 Sequence Diagram For Supervisor



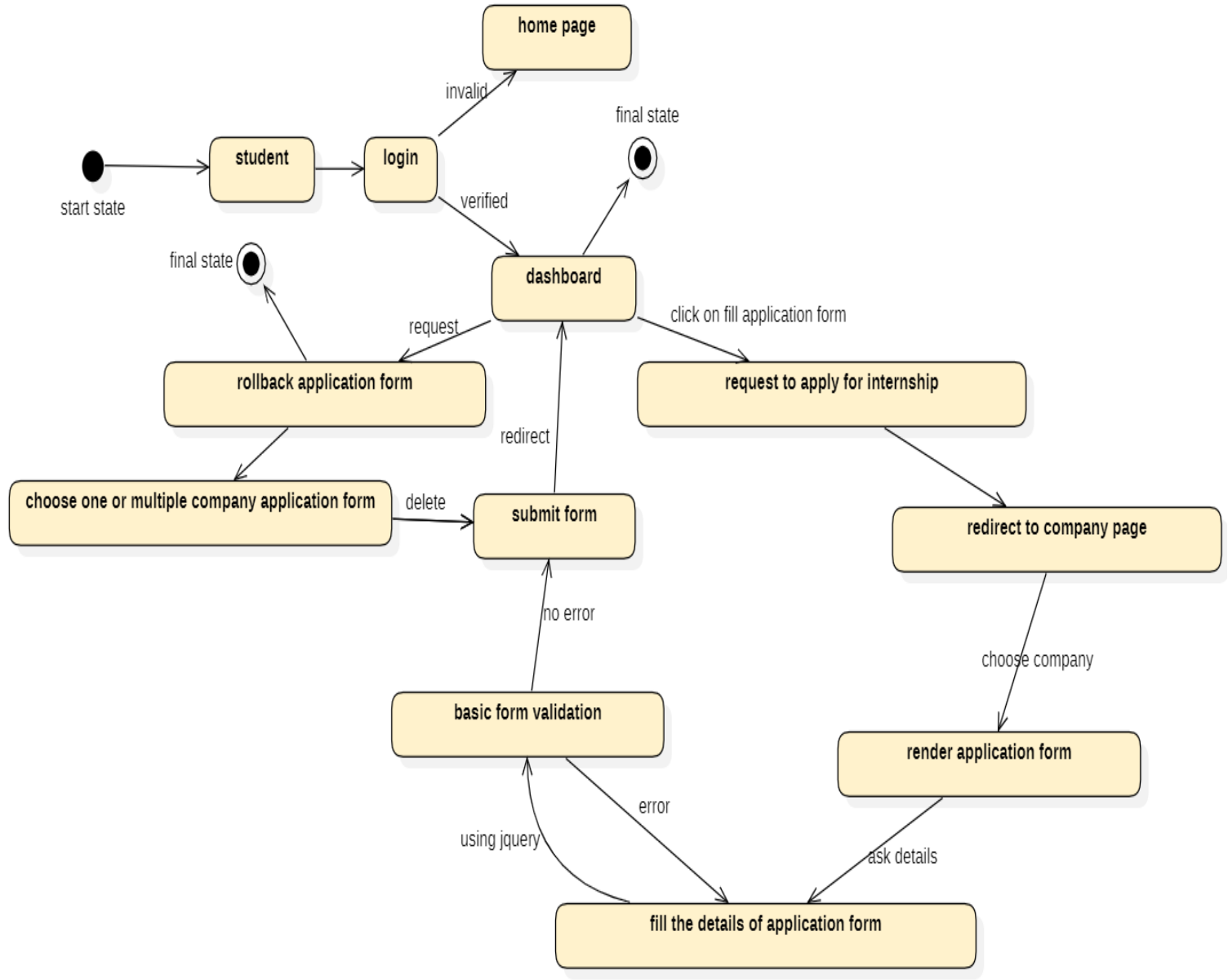
6. Entity Relationship Diagram

Clean Frame's ER - Diagram

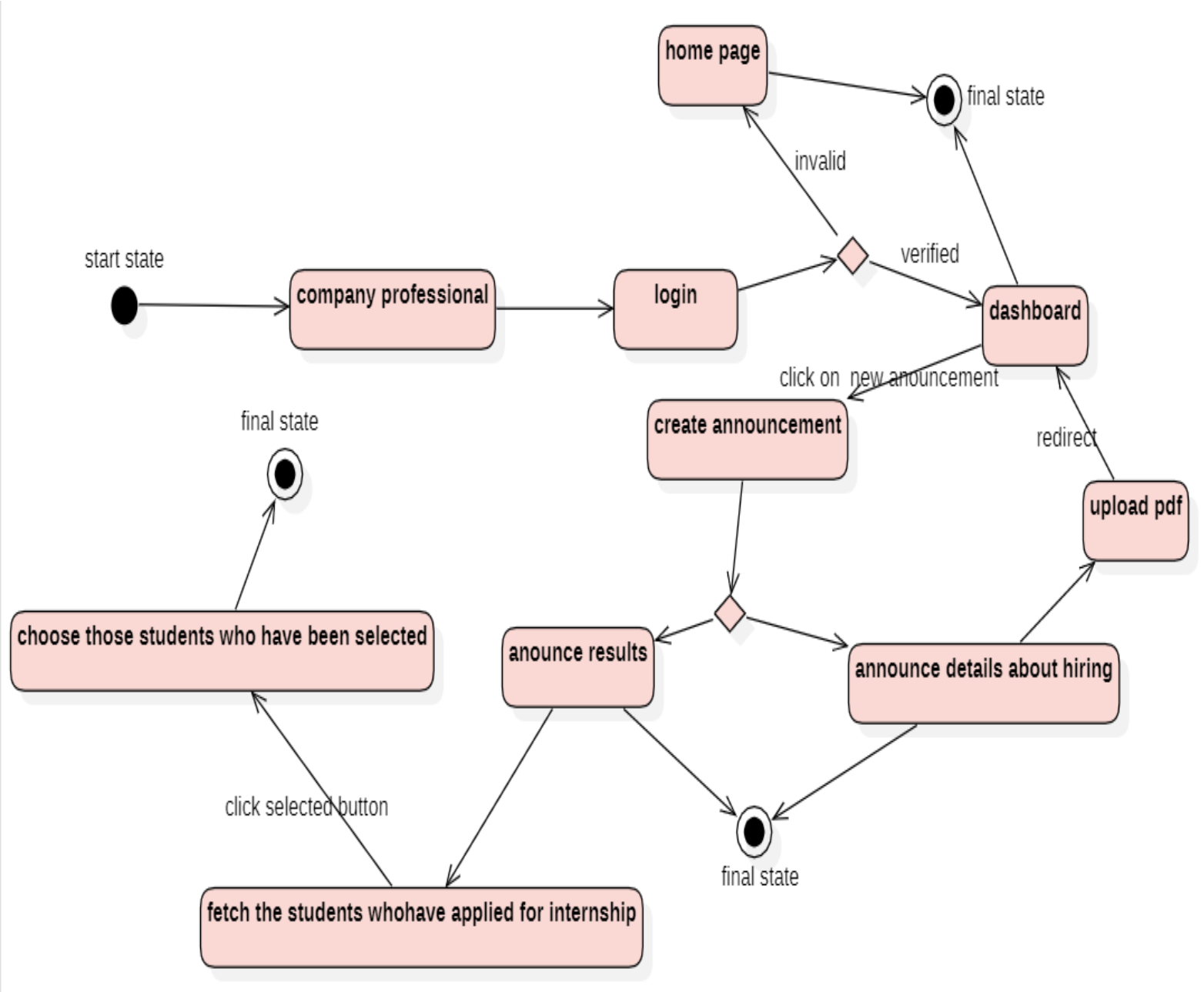


7. State Diagram

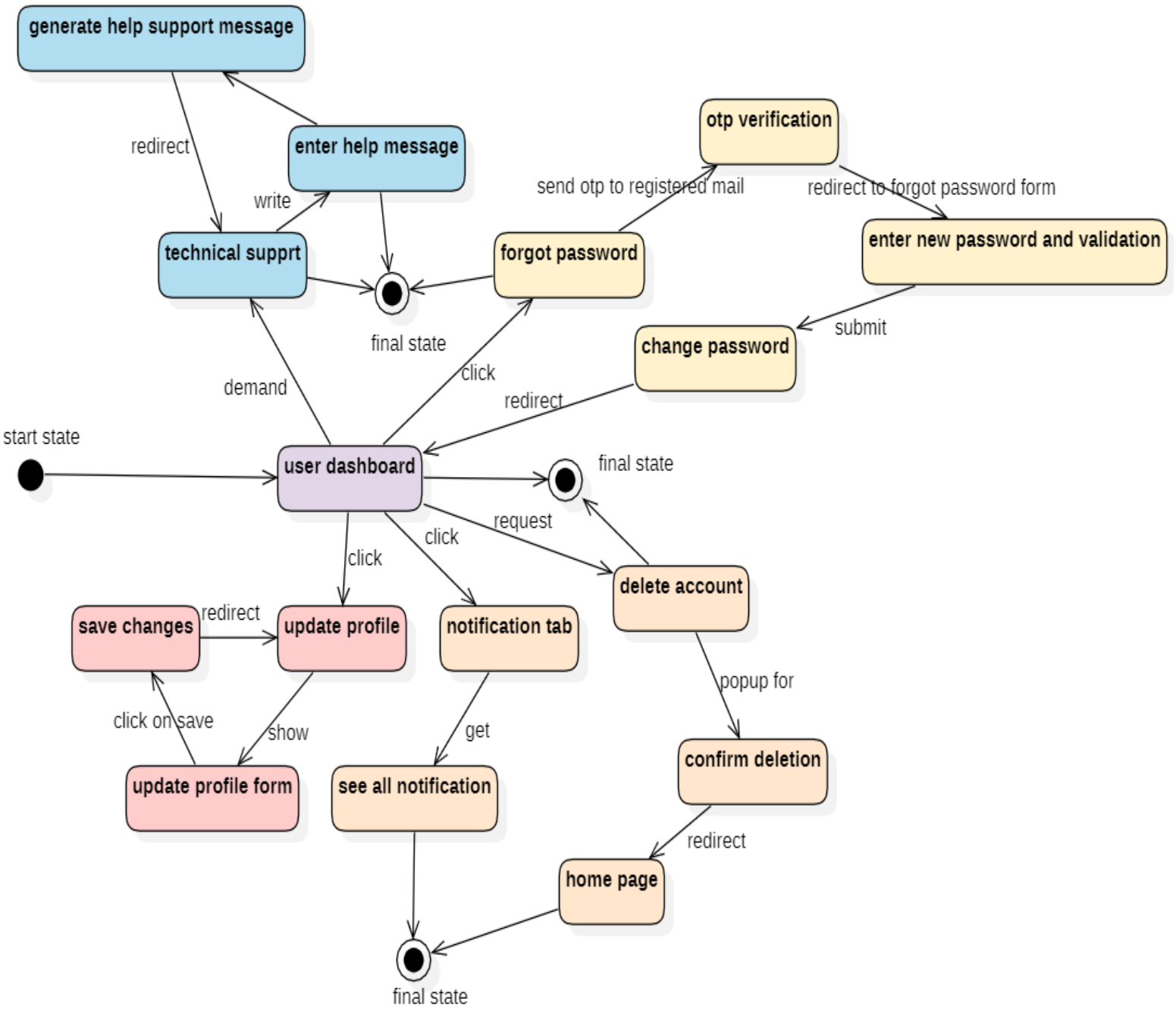
7.1 StateDiagram For applying for internship and rollback application



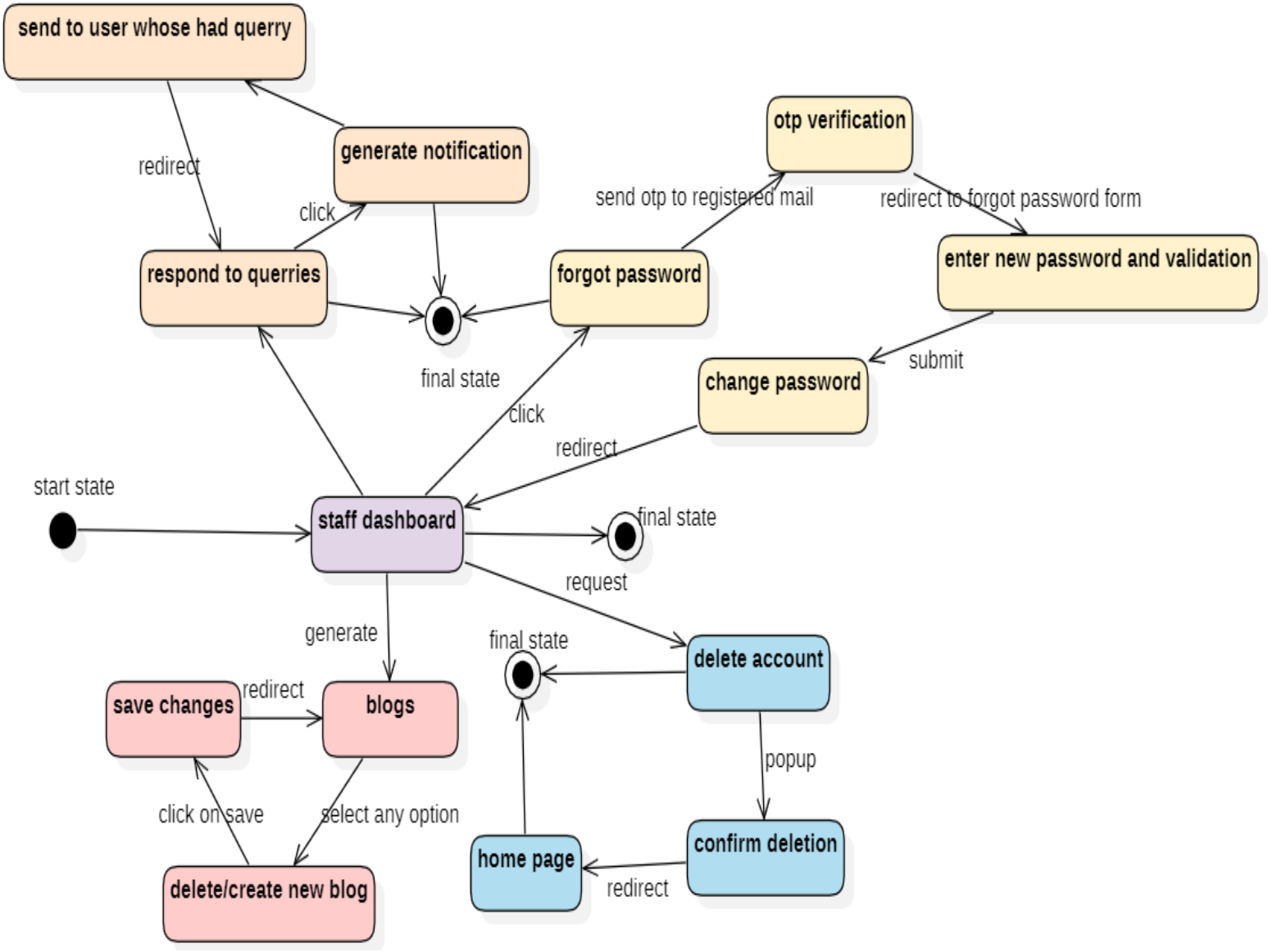
7.2 StateDiagram For announcements by company professionals



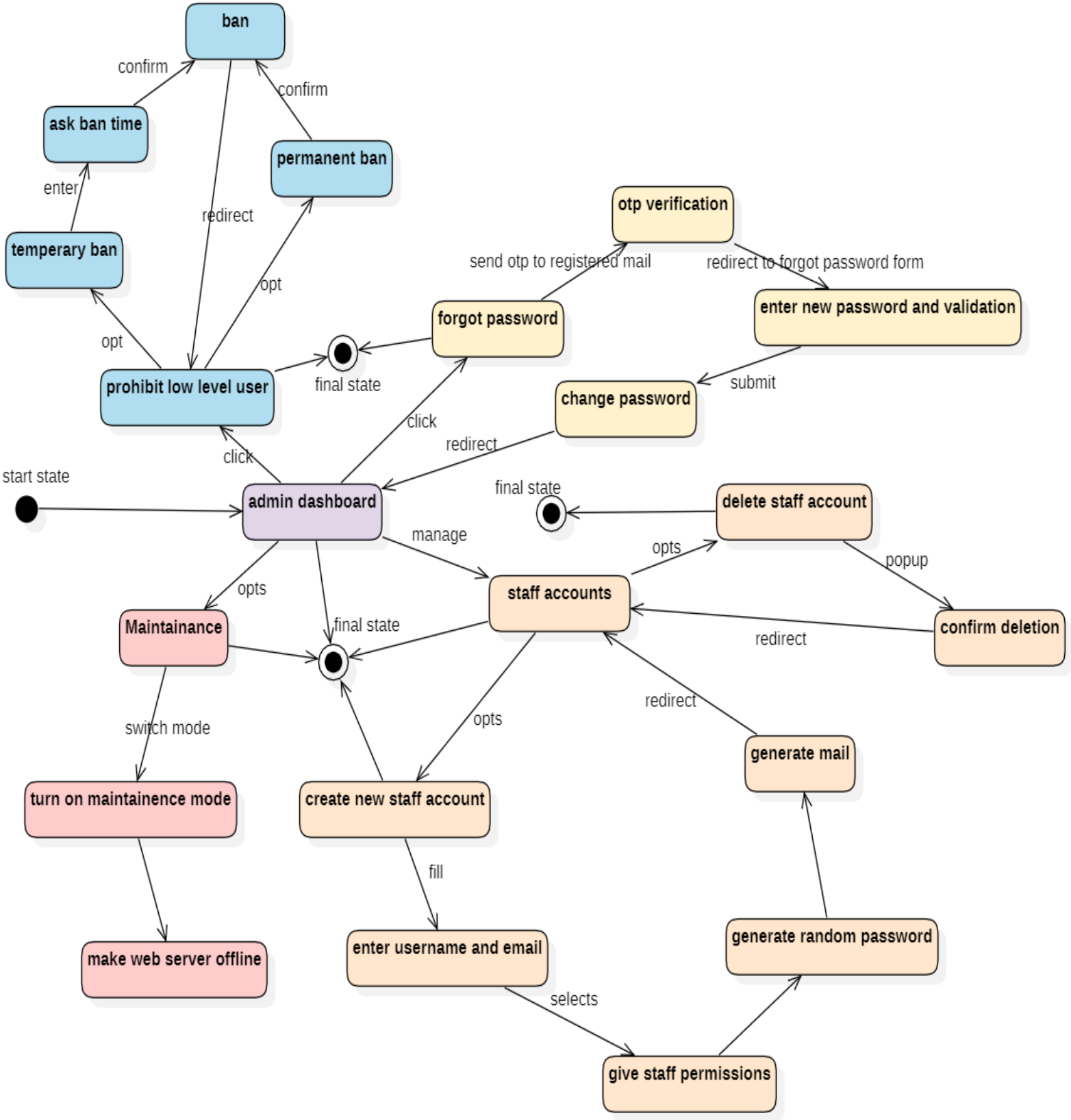
7.3 StateDiagram For student/company professional dashboard facilities



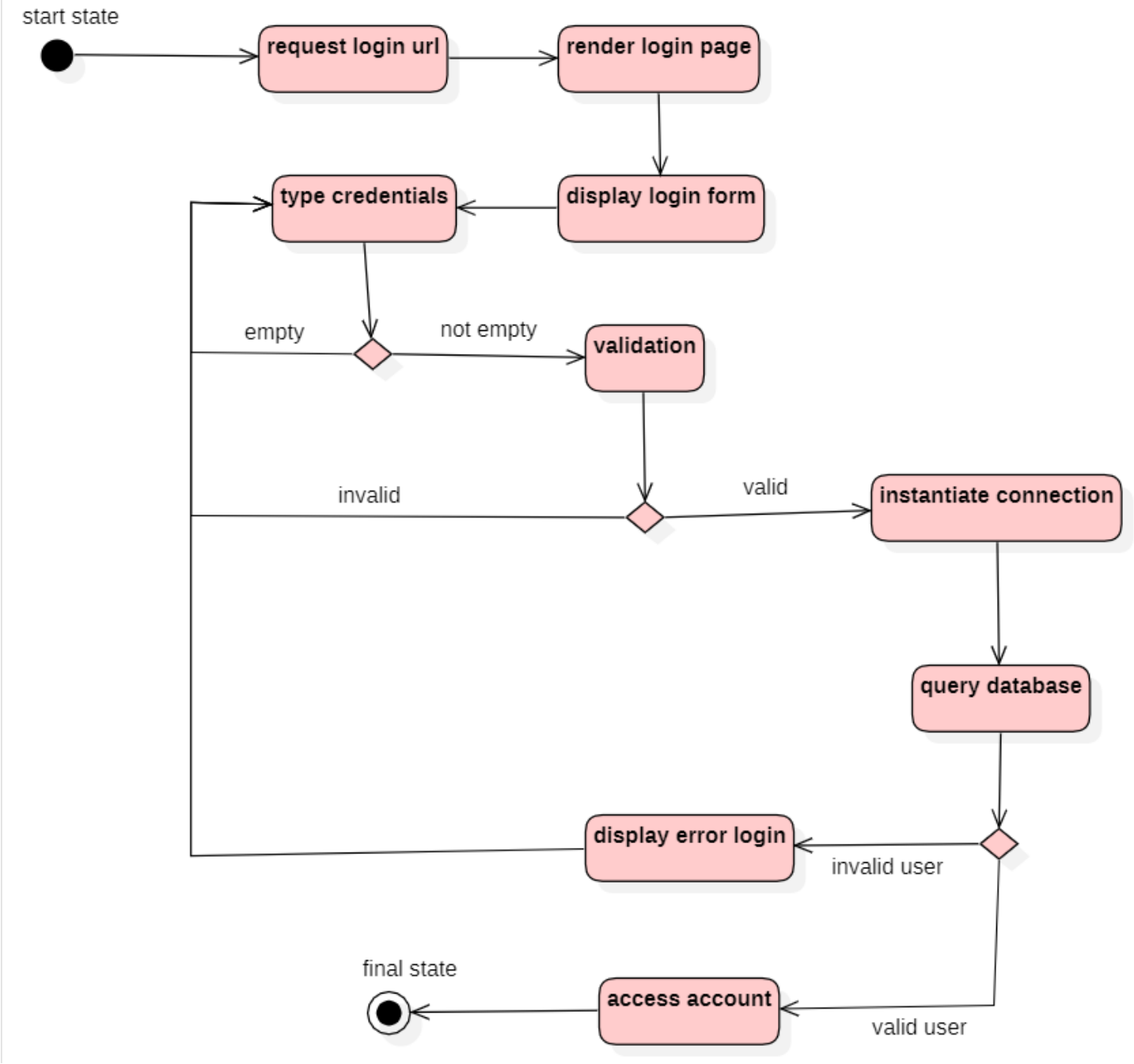
7.4 StateDiagram For supervisor/admin dashboard facilities



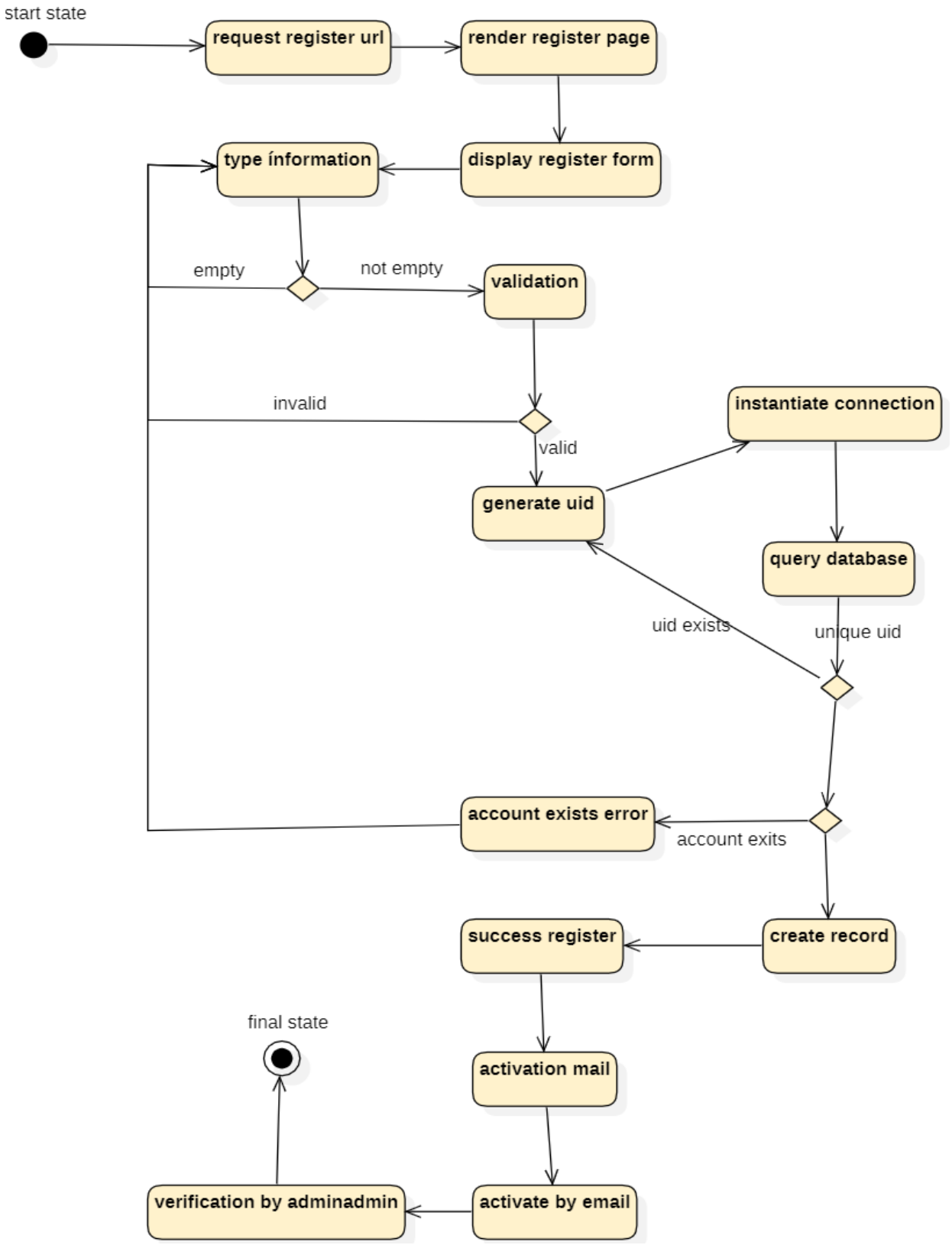
7.5 StateDiagram For admin roles



7.6 StateDiagram For login

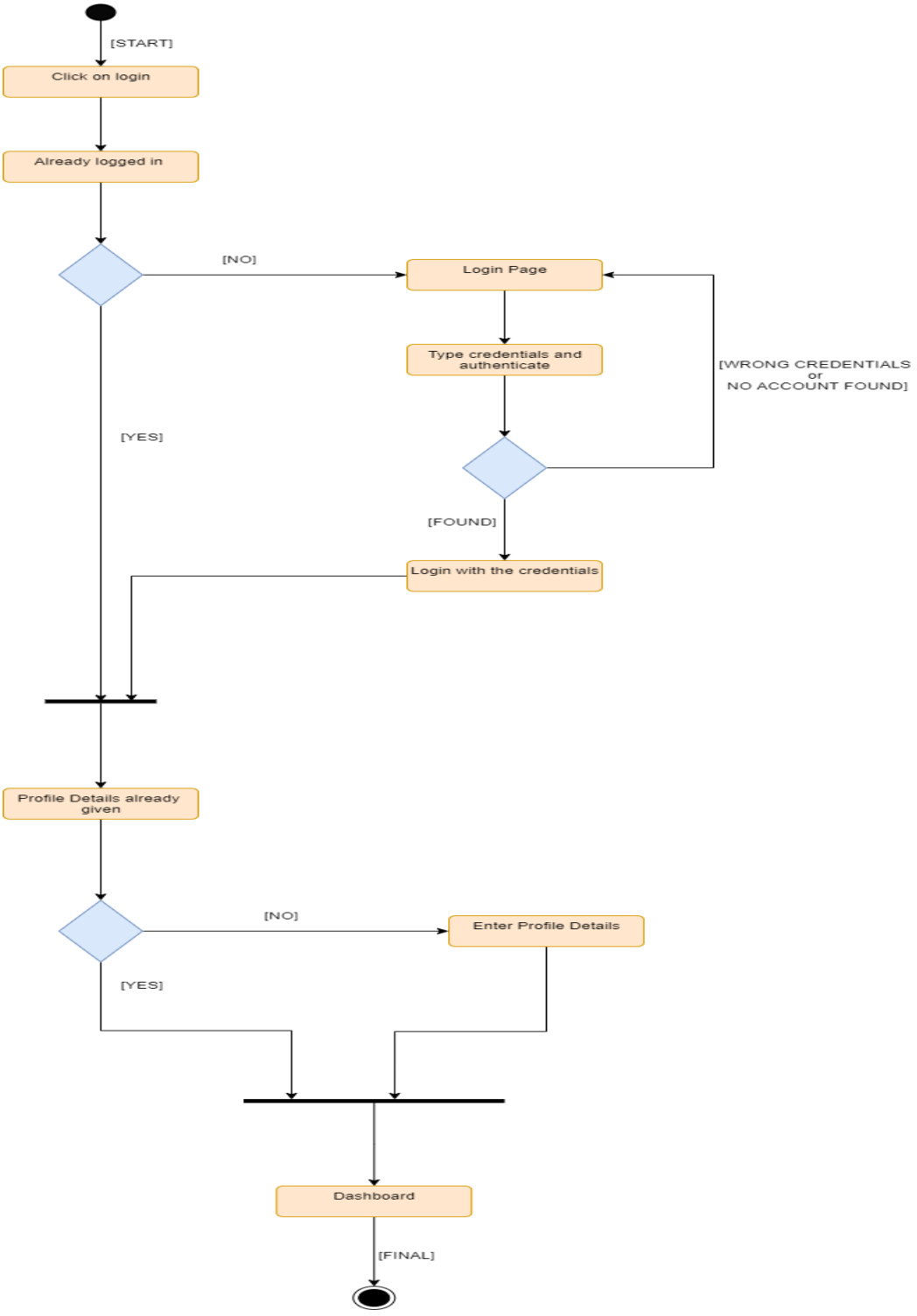


7.7 StateDiagram For registration

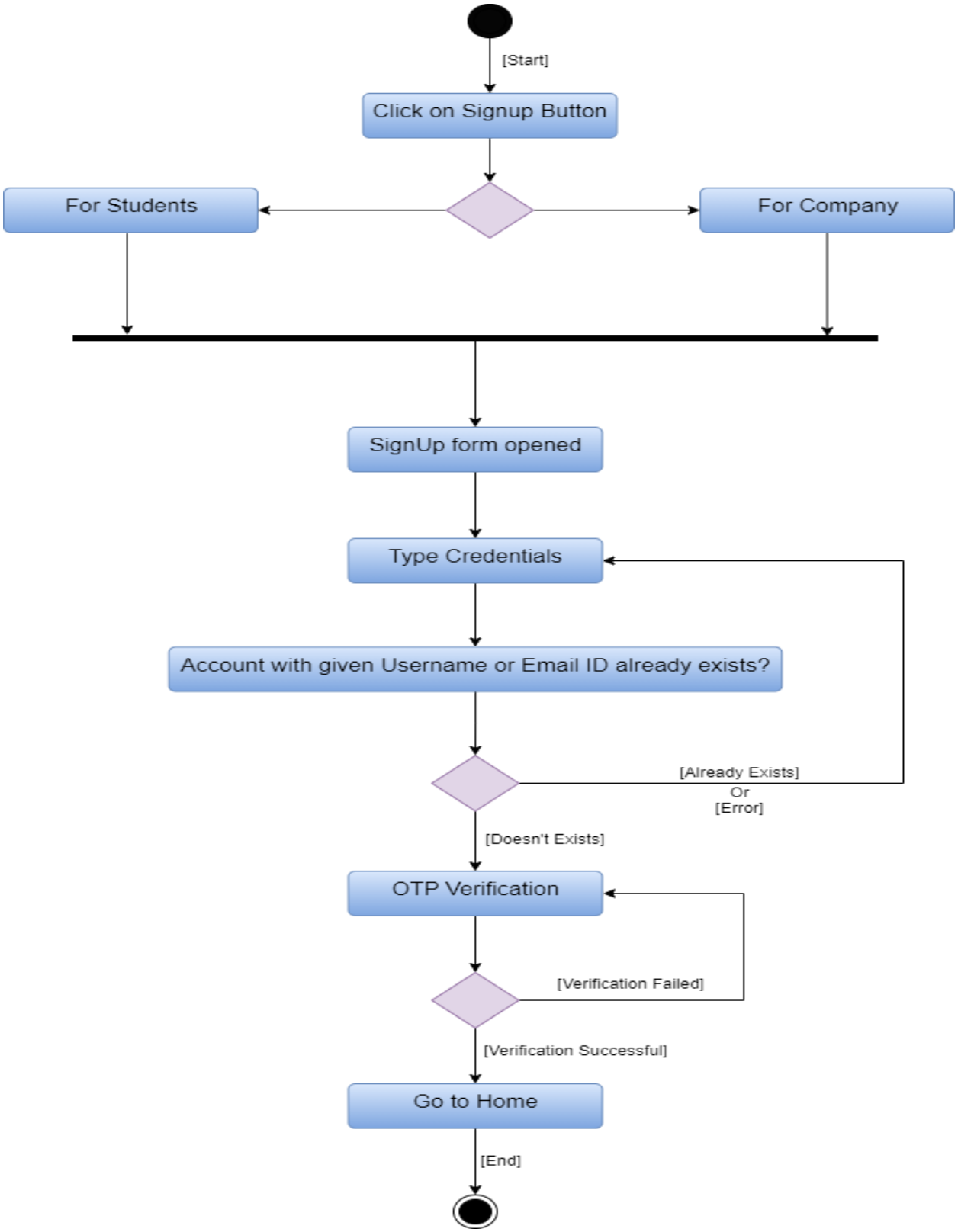


8. Activity Diagram

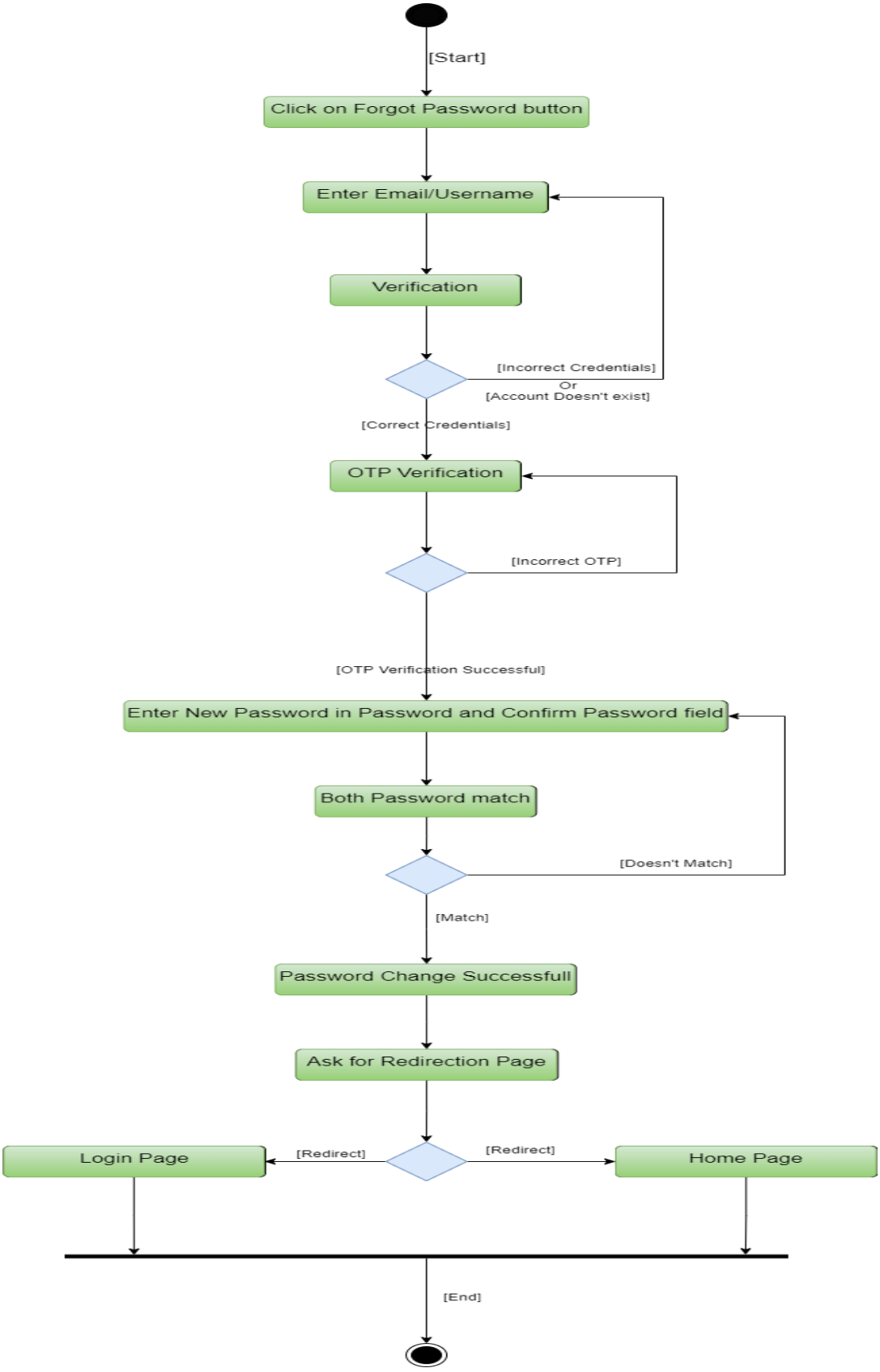
8.1 Login



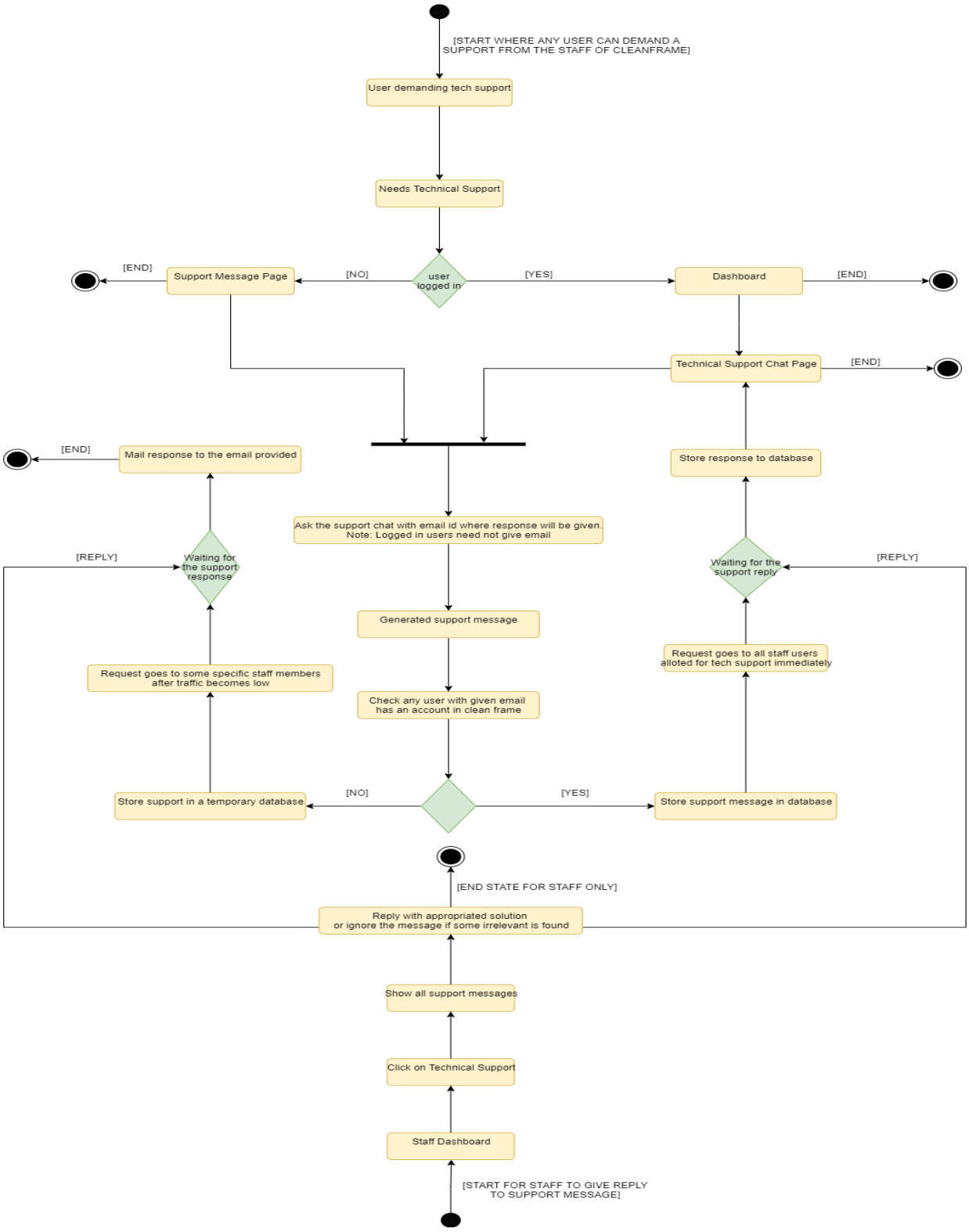
8.2 SignUp


[Start]
Click on Signup Button
For Students
For Company
SignUp form opened
Type Credentials
Account with given Username or Email ID already exists?
[Already Exists]
Or
[Error]
[Doesn't Exists]
OTP Verification
[Verification Failed]
[Verification Successful]
Go to Home
[End]

8.3 Forgot Password

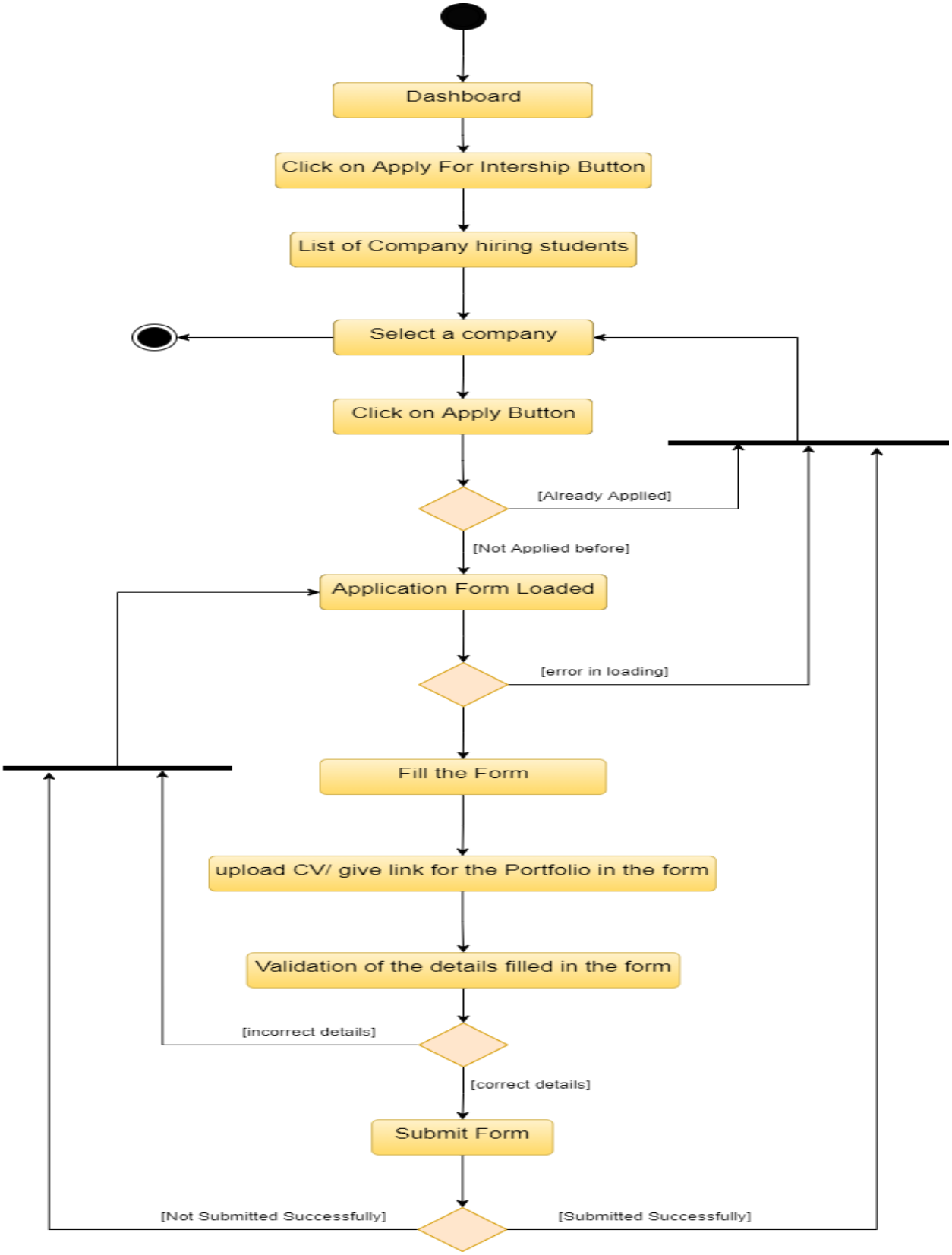


8.4 Technical Support

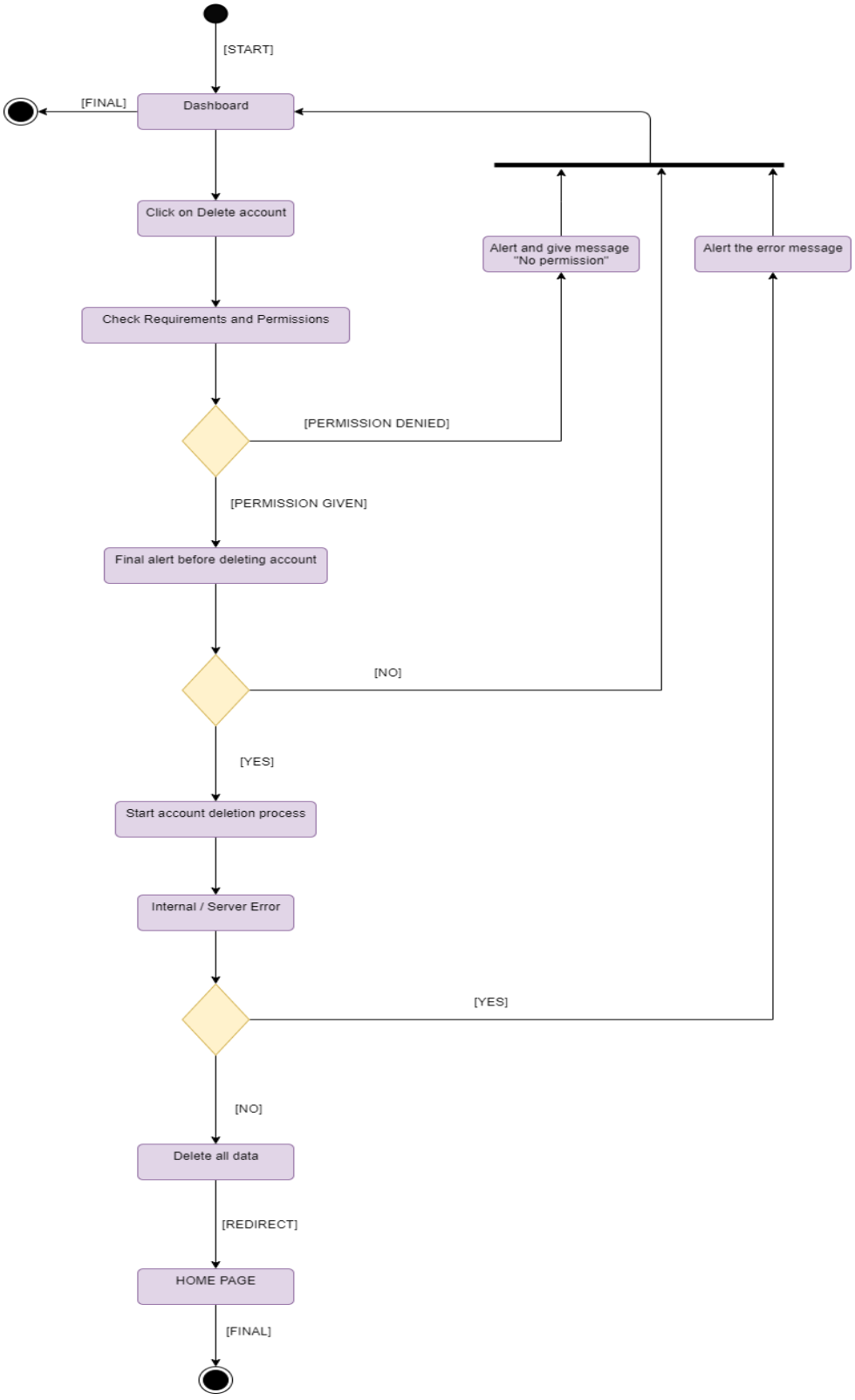


8.5 Update Profile

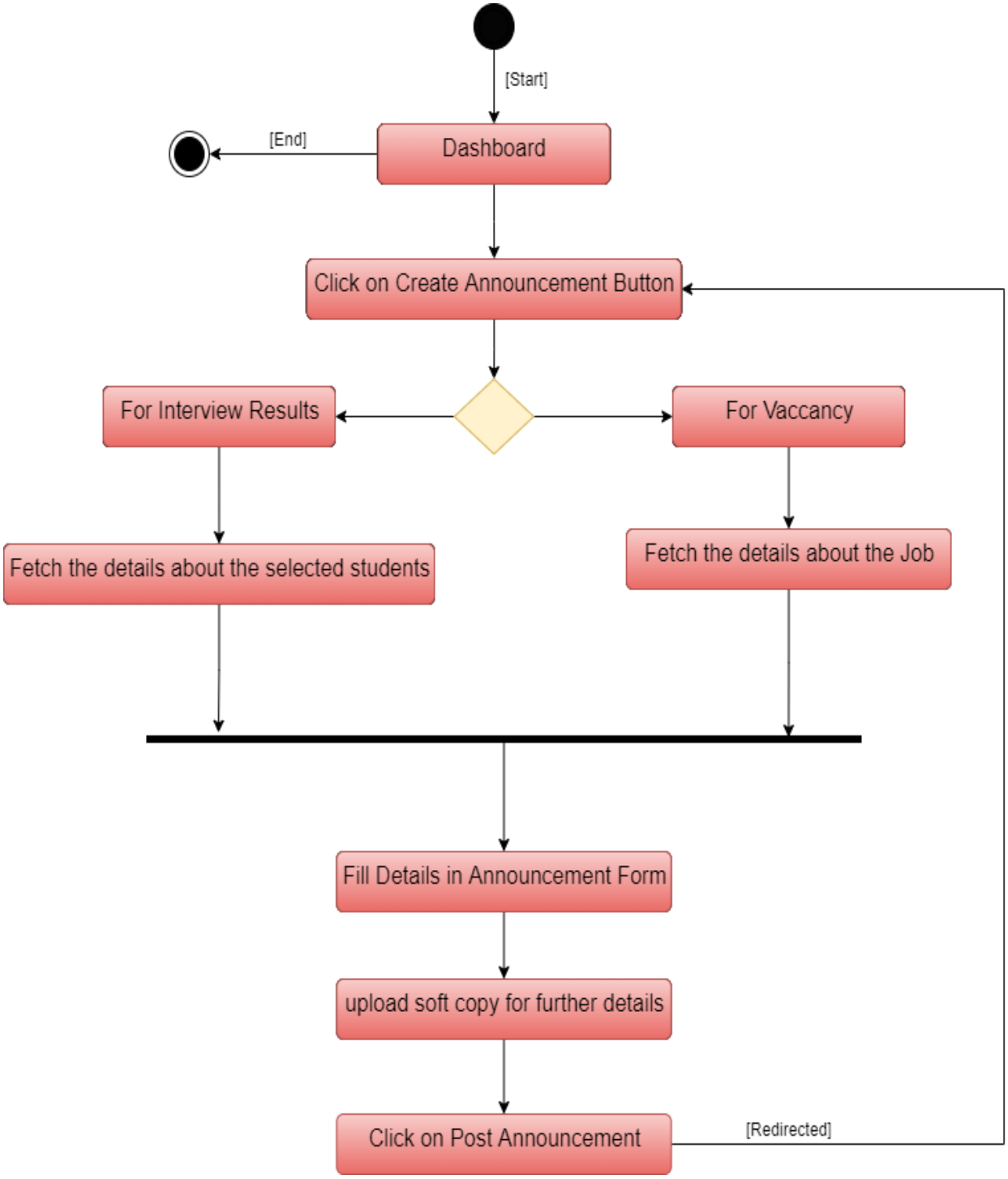
8.6 Apply For Internship



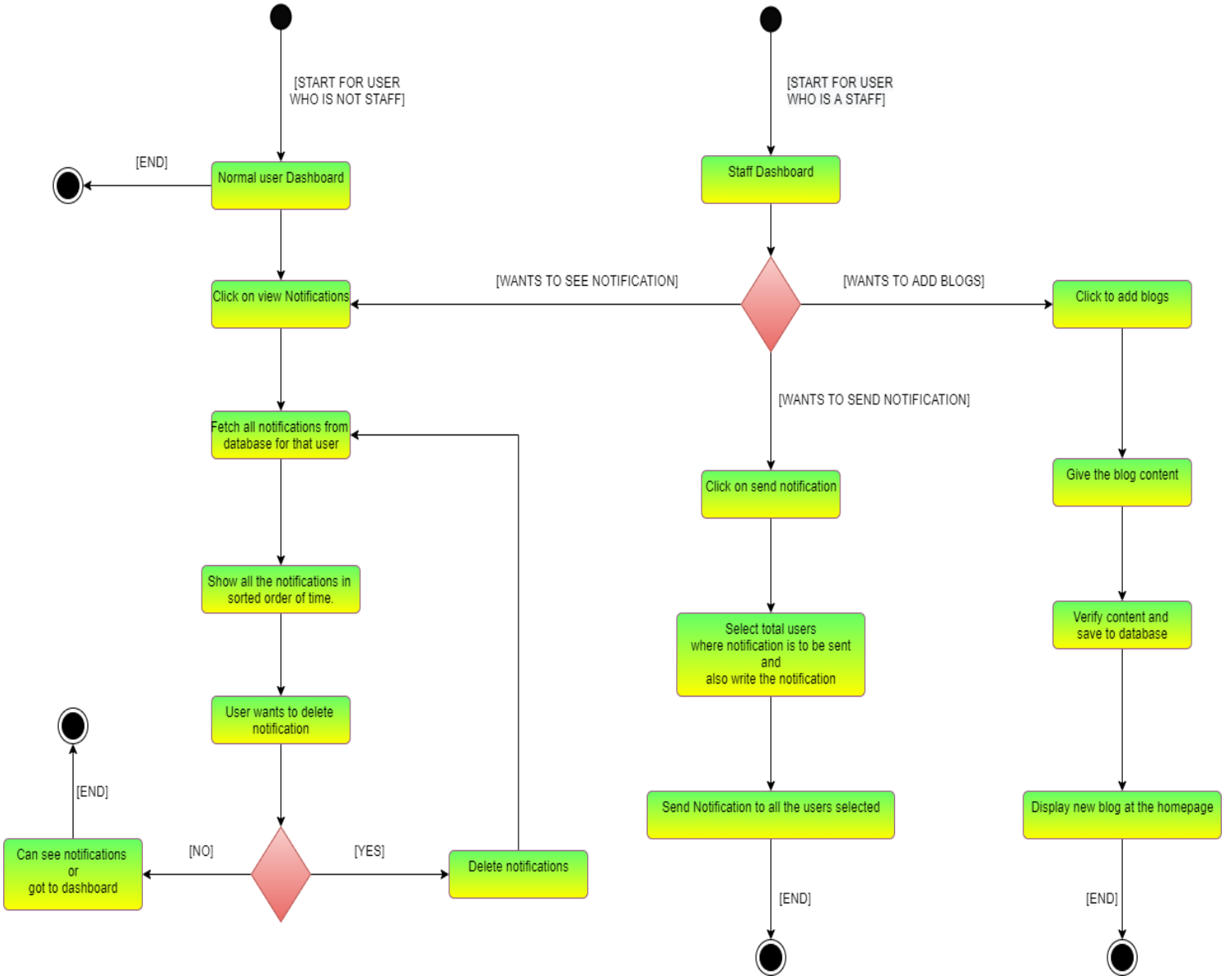
8.7 Delete Account



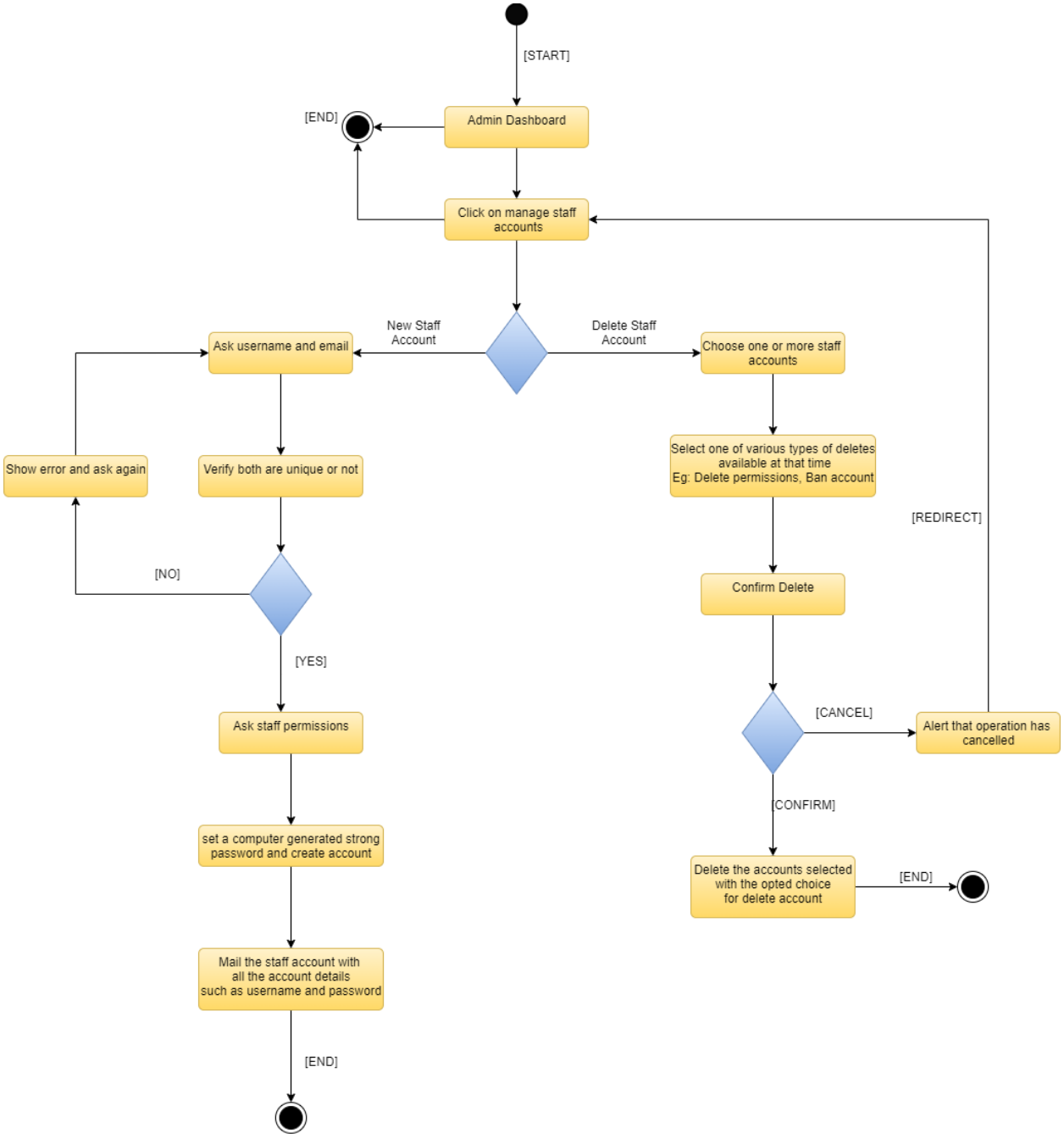
8.8 Company Announcement



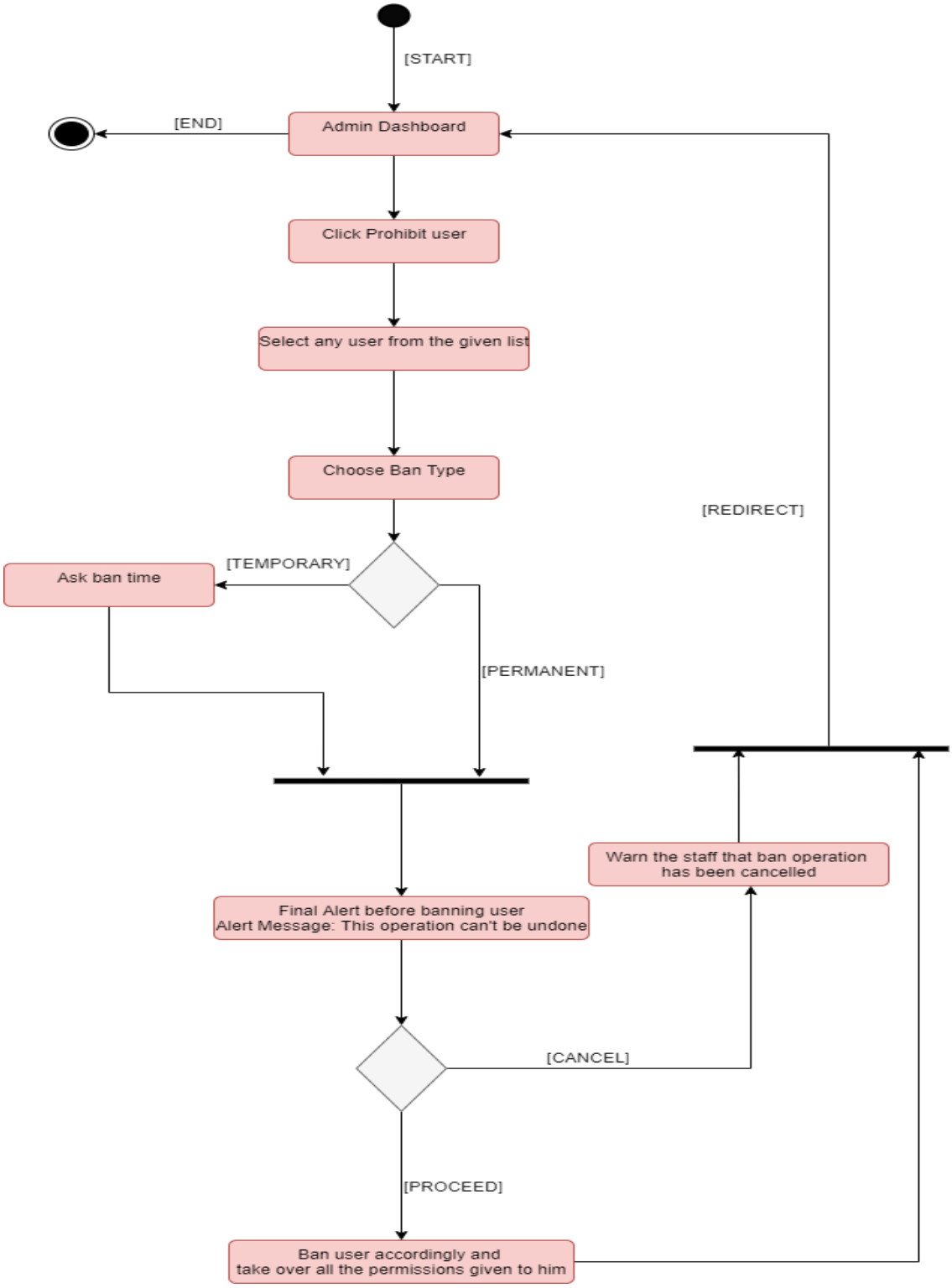
8.9 Create or View Notification and Add Blogs



8.10 Manage Staff Accounts



8.11 Account Prohibition



9. Explanation of DataFlow Diagram

9.1 Level 0 Data Flow Diagram

- ➢ Typically known as Context Diagram.
- ➢ In this level only single process flow is shown.

In this level, the process (P1) is the whole Internship Portal. Now main 6 external entities (E) are shown that are Admin, Company Professional, Database Management, Dashboard, College Students and Staff/ Supervisor. These 6 external entities are linked with the P1 in the manner such that data flow takes in both the directions i.e. From P1 to E and E to P1.

9.2 Level 1 Data Flow Diagram

- ➢ Typically known as Overview Diagram
- ➢ In this level we use external entities, multiple processes as well as data storages.

In this level basically we shown the main data flow that where is it flowing not how is it flowing. It has 4 phases on the whole which are well explained as above:

1. College Student -> Internship Portal -> Student Login Approved (P).
   Now further data flows from P as:
   P -> Roll Back Application Form
   P -> Give Portfolio
   P -> New/ Update CV
   P -> View Company Details
   P -> Get Notifications
   P -> Reset Password
2. Company Professional -> Internship Portal -> Company Login Approved (P).
   Now further data flows from P as:
   P -> See Student CV
   P -> Update Profile
   P -> Result Announcement
   P -> Reset Password
3. Staff/ Supervisor -> Internship Portal -> Staff Login Approved (P).
   Now further data flows from P as:
   P -> Manage Internship Details
   P -> Accept/ Decline Signup Request
   P -> Manage Interviews
   P -> Reset Password
4. Admin -> Internship Portal -> Admin Login Approved (P).
   Now further data flows from P as:
   P -> New Staff Account
   P -> Ban User
   P -> Technical Support
   P -> Blogs and Announcement Management
   P -> Change Staff Permissions
   P -> Security and Maintenance
   P -> Reset Password

9.3 Level 2 Data Flow Diagram

- ➢ Typically known as Detailed Diagram
- ➢ Basically this level is the further explanation of level 1.

In this level a total of 12 processes are shown, with 5 external entities and 9 data Storage with further extension to level 1. The detailed explanation is given below with each and every flow of data:

1. Data flow going from the user into the login system (P1) when he enters login credentials. From here this data flows into the User data (D1) and in return D1 gives a response which is a form of data that flows back into the login system. Now if login is successful data flows in the form of success and takes the user to its dashboard. Otherwise data flows in the form of error and error in the data is shown to the user.
   Data Flow Overview : User -> P1 -> D1 -> P1 -> User -> P2 (on success)
   In this also when P2 is reached on success (that is when dashboard is achieved) then in the backend data flows from D2 to User level Data (D3) which has the level of every user and users are shown the data according to it. So, data flowed shows only that information that is in its permitted zone.
   Data Flow Overview : P2 -> D3 -> P2 -> Student/ Supervisor/ Company (E)
   (E) is the external entity and its value will depend on the type of user.
2. Data flow going from the Dashboard (P2) to Profile Data (D2) when a user requests to edit a profile. In this if the profile is fetched then success is returned as response in the form of data which flows back to P2. If success is met then data again flows to Profile (P3). Otherwise data flows in the form of error and error in the data is shown to the user.
   Data Flow Overview: P2 -> D2 -> P2 -> P3(on success)
3. Data flow going from the Dashboard (P2) to Delete Account (P12) when a user requests to delete the account. Now further again the data from P12 flows into Profile Data (D2) to fetch the profile. In response error or success would be given in the form of data which flows back to P2 through P12. If success is met then the user account is deleted. Otherwise data flows in the form of error and error in the data is shown to the user.
   Data Flow Overview: P2 -> P12 -> D2 -> P12 -> P2
4. Next case is when the student wants to view any company details, then data flows from Student to View Company Details (P4). Further data flows from P4 to Company Profile Information (D4). In return the response is given which is the same as in the above cases i.e. if success then company details would be shown else error would be shown.
   Data Flow Overview: Student -> P4 -> D4 -> P4 -> Student -> Profile Page(on success)
5. Next case is when any student or company wants to see the profile/ CV of some student, then data flows takes place from student/ company to View Student CV (P5) and then data flows from P5 to Student Profile Information (D5). In return the response is given to both the P5 and the user.
   Data Flow Overview: Student/ Company -> P5 -> D5 -> P5 -> user -> Profile Page (on success).
   Note : Here a student can't search for his own profile. Obviously he can access and update his profile. This is for other students who wants to see other's profiles.

6. Now the data flow takes place when the company does any announcement whether related to results or new vacancy etc. In this case data flows from company to Announces Result (P6) and then from P6 to Notifications Table (D6). Then the response comes and if success is met then new notification is generated in the notification panel of only those users who are related to it which is our P7 or Generate Notifications (explained in next point).
   Data Flow Overview: Company -> P6 -> D6 -> P6 -> Company -> P7 (on success)
7. Now whenever a successful Notification (D6) is generated then data flows from D6 to Generate Notifications (P7). Finally data flows to all the users associated with it. If error is encountered then it's displayed on the webpage from where the request was generated.
   Data Flow Overview: D6 -> P7 -> D6 -> Users (on success)
   Note : Whenever we reach D6 or P7 it is automatically assumed that P7 or P' is executed respectively where P' is the process of message delivering to the users.
8. Whenever a high level user wants to float notification then data flows from user to D6 and response is returned.
   Data Flow Overview: High Level User -> D6 -> user -> (on success continue with point 7)
9. When staff wants to add blogs then data flows from staff to Add Blogs (P8) and then it flows from P8 to Blogs (D7) and response is returned which comes to Staff.
   Data Flow Overview: Staff -> P8 -> D7 -> P8 -> staff -> (blog is added)
10. Next case is when the admin wants to add a new staff and wants to create an account for it. In this case data floors from admin to New staff account (P9). Data further flows from P9 to Accounts (D8). As usual a response is returned. If successful then a mail is sent to the staff email with a high security password.
    Data Flow Overview: Admin -> P9 -> D8 -> P9 -> admin -> mail (on success)
11. Next Case is when the admin requests for a ban of an user account, when he finds something wrong inactivity in the account. Now the data flows from admin to Prohibit user (P10) and then further to User Data (D1) and then further to D7 if ban is successful.
    Data Flow Overview: Admin -> P10 -> D1 -> P10 -> admin -> P7 (on success)
12. Last data flow case is when any user needs technical support from the admin/ staff users. In this case data flows from user to Technical Support (P11) which further moves to admin/ staff user. In return admin adds the support chat and data flows from admin to Technical Issues (D9) and finally response is given to P11 and the final support help answer message is reached to the user if no error is encountered in between.
    Data Flow Overview: user -> P11 -> admin -> D9 -> P11 -> user -> (support reply on success)

10. Explanation of Sequence Diagram

This diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

10.1 Web Page Display

To exploit the application to the full extent the user would be sending a request to access the web page to the server after entering the address of the desired website. If because of some reason the request was not processed or the entered address of the website by the user was wrong then the error would be prompted on the user's screen.
But if there was no error i.e., that is the address entered was correct and the request was processed successfully then the web page will be displayed on the user's screen.

10.2 Registration

1. **Student Registration**

   The student who wants to get an internship needs to register themselves on this application as a prior requirement to be eligible for the internship process. The students choose the option "Register as a student" on the web interface. And the registration process request would be sent to the server. Now the Registration Credentials entered would be stored into the database and saved after verification of some of the fields have been done by the IIITA personnel. After the verification has been done if all the details were correct then the user would be notified about the account creation. If any of the fields were not correct then the registration needs to be done again. One of the boundations on the students is that the email id should be that provided by the college which is not there on the company registration.

2. **Company Registration**

   The company who wants to hire students for various positions also needs to register themselves on the portal as a prerequisite. For registration they need to choose the option "Register as Company" on the web interface. And the registration request would be sent to the web server. Now the Registration Credentials entered by the company professionals would be verified and then stored in the database. If any of the details entered is wrong then they would need to register again.
   For both of the user's verification processes would include several types of verification such as OTP verification and all.

10.3 Login

For accessing the various facilities the user needs to login first, for that he would need to enter details such as username and password into the fields on the web interface which would be later sent to the web server for further verification and login process. The details entered by the user would be compared with the user details in the database. If the details do not match then the user will be asked to login again. But if the login details match then the user would be notified about the successful login.

10.4 Change Password

For changing the password there should be two cases-

1. If user has already logged in (i.e he/she want to change the password) then he need to enter details like old password and new password, after that if old password is correct then the new password is stored in the database while if old password is not correct then then he need to do the same process again.
2. When the user doesn't remember the password then he needs to verify email first, if the verification step has been done successfully then he has to enter the new password after that the password has been stored in the database.

10.5 Upload Portfolio/CV

For uploading the portfolio/CV, users need to fill it in their dashboard where he/she need to upload the portfolio/CV, then the uploaded portfolio/CV is stored into the database.

10.6 Fill application form/profile detail

After the registration user needs to update his profile for that he/she needs to fill the form in the given field, after that the filled data is stored into the database.

10.7 Uploading interview result

After the interview which has been held by the company professional, the company needs to declare the result of the students. So, the company uploaded the interview result, after that the result is stored in the database for further use.

10.8 View Applicant detail

After uploading the details of students, companies are able to see the profile detail of the applicant. So when companies need this information, the student's details are fetched from the database.

10.9 Student/Company Verification

During the registration process of the company and student we need to verify their accounts. For that the verification process is required. In this process a verification code has been sent to the email of the user/company when the verification code is the same then the next step of verification is moved to the Admin system. Where the admin manually verifies the account. After the verification step the verification status has been stored in the database.

10.10 Notification/Announcement creation

If the company wants to send the notification then they go to the notification section, after that they need to write the notification and send it, after that the notification is stored into the database and the notification is sent to the students.

10.11 Reviews

If a user wants to give a review on someone's profile then he/she should give it by viewing their profile and after that the review is stored in the database.

10.12 Update user detail

If a user wants to update the detail then he needs to fill the field which he wants to be uploaded after that the uploaded detail has been stored into the database.

11. Entity's and their relationship in the ER Diagram

11.1 User

Attributes:

- User ID (Primary Key)
- Email Address
- First Name
- Last Name
- Username
- Password (Encrypted)

Relationships with explanation:

- 2 Many to One relationships with Admin.
  - **Tech_Support:** Users take technical support from a single admin and one admin gives tech support to many users.
  - **Give_Notification:** One Admin can give notifications to many users.
- Admin, Student and Company are in an IS-A relationship with the user because these 3 are special types of users.

11.2 Student

Attributes:

- Student ID (Primary Key)
- Contact Number
- CV
- Image
- Address
- Gender
- Profile Creation Date
- isBanned (boolean)
- SignupDate
- isProfileVerified (boolean)

Relationships with explanation:

- 1 One to One Relationship(**Has**) with Portfolio Reviews since every student will have only 1 combined or average portfolio review.
- 1 One to Many Relationship(**can_give**) with Portfolio Reviews since one student can give review on others portfolio.
- 1 One to Many Relationship(**applied_for**) with Company since in the context of student one student can apply for multiple companies.
- IS-A Relationship with Users.

11.3 Portfolio Review

Attributes:

- Student ID (Foreign Keys as Multi Valued Attributes)
- Review ID (Primary Key)
- Ratings

Relationships with explanation:

- 1 One to One Relationship(**Has**) with Students since every student will have only 1 combined or average portfolio review.
- 1 Many to One Relationship(**can_give**) with a Student since one student can give review on others portfolio.

11.4 Company

Attributes:

- Company ID (Primary Key)
- Contact Number
- Company Address
- Internship Duration
- Profile Creation Date
- isBanned (boolean)
- SignupDate
- Minimum CGPA Required
- Prerequisites
- isProfileVerified (boolean)
- Stipend
- Internship Position
- Total Students Required

Relationships with explanation:

- 1 One to Many Relationship(**applied_by**) with Students since in the context of a company one company's application form can be applied by multiple students.
- 1 One to Many Relationship(**announces**) with Result since a single company could announce multiple results of various rounds including vacancy and final list.
- IS-A relationship with User.

11.5 Result

Attributes:

- Result ID (Primary Key)
- Student ID (Foreign Keys as Multi Valued Attributes)
- Company ID (Foreign Key)
- Total Selections/ Vacancies
- Total Applications

Relationships with explanation:

- 1 One to One relationship with a company(**announced_by**) 1 result would only be announced by 1 company.

11.6 Admin

Attributes:

No such extra attributes required because admins don't have any profile.

Relationships with explanation:

- 1 One to Many relationship(**adds**) with Blogs because one admin can add multiple blogs.
- 2 One to Many relationships with Admin.
    - **Tech_Support:** Users take technical support from a single admin and one admin gives tech support to many users.
    - **Give_Notification:** One Admin can give notifications to many users.
- IS-A relationship with User.

11.7 Blogs

Attributes:

- Blog ID (Primary Key)
- Title
- Image
- Short Description
- Brief Description

Relationships with explanation:

- 1 One to One Relationship(added_by) with Admin because a single blog can be added by only one admin.

12. Explanation for State Diagrams

The state diagram generally depicts the state of a particular feature in this web based software to achieve an aim. The filled circle depicts the start state and the filled circle with a ring depicts the final state. The arrows depict an action performed on a particular state to arise at a particular state.

12.1 Rollback Application Form

First the student login through his/her credentials and is redirected to the home page if found invalid. If verified, the student's dashboard will be displayed. From there on, all the student's applications would be displayed and the student can choose the application which he wishes to rollback.
After choosing the application, he/she will be redirected to a form where all the companies would have to be selected from the list from which the student wishes to rollback the application. The student will submit the form, the applications would be deleted and he will be redirected to the dashboard.

Source: 7.1

12.2 Applying for Internship

First the student login through his/her credentials and is redirected to the home page if found invalid. If verified, the student's dashboard will be displayed. Now the student will be able to click on a button to fill an application as a request to apply for an internship among the list of companies at the portal.
He/she would be able to see the company profile as he would be redirected to that page and choose all the companies he wishes to apply for and an application form would be rendered.
He/She would be asked to enter details into that form and basic form validation would be done using jquery. If an error is detected, he/she would be redirected to the application form again to change wrong entries.If no error is found, the application

would be submitted and dashboard page would be rendered.

Souce: 7.1

12.3 Announcement Creation by Company Professional

First the company professional would be asked to login to the system. If found invalid he/she shall be redirected to the home page. If verified, his/her dashboard would be rendered, from where a new announcement can be created.

He/She can announce results or announce details about a new vacancy available for an intern at their company. For announcement for a new hiring, a pdf related to it can be uploaded and he/she would be redirected to the dashboard and for displaying results of a test, an algorithm would fetch the students who had applied and given the test.

From this list the selected students can be check listed and an announcement would be generated accordingly for that and hence, the aim achieved.

Source: 7.2

12.4 Technical Support and Response

First the general login shall take place and if verified, the user's dashboard will be rendered. If the user is facing some technical issues, he/she can demand technical support from the team for which a help message would have to be written. Accordingly a help support message would be generated on the platform and would be sent to the team concerned. The user would be redirected to the dashboard after this.

Source: 7.3

12.5 Forgot Password

First the general login shall take place and if found invalid, the user can click on forgot password to change his/her password. After clicking on the button, an OTP would be generated and sent on the registered email which needs to be entered within a specified time window.

After that, a forgot password form would be displayed where the user would be demanded to type in the new password and it would be validated there. If all went well, the user can click on the submit button and password would be changed and the user would be redirected to the homepage.

Source: 7.3, 7.4, 7.5

12.6 Profile Updation

First the general login shall take place and if verified, the user's dashboard will be rendered. To update the profile, a button would have to be clicked on the dashboard which would take the user to a form where all the necessary details would be taken from the user regarding the updation required. The changes would be rendered and if the user likes the changes made, the user can click on the save button to save the changes made. He would be redirected to the dashboard after this.

Source: 7.3

12.7 Account Deletion

First the general login shall take place and if verified, the user's dashboard will be rendered. The user can delete his/her account after clicking the delete account button, a "confirm deletion" pop-up would be displayed and since his/her account credentials have been verified, the account would be deleted. The user would be redirected to homepage.

Source: 7.3

12.8 Notifications

First the general login shall take place and if verified, the user's dashboard will be rendered. To view all the past notifications, the user can click on the notifications tab, the software will fetch all the notifications related to the defined credentials.
All the notifications would be displayed and the user would be able to see them and the aim achieved.

Source: 7.3

12.9 Blogs

First the staff's login shall take place and if verified, the staff official would be redirected to the staff dashboard. From here, the official can generate a blog which could benefit the users of the software.
The official has the power to create/delete the blogs at the portal by selecting an option under the blogs section. After making the necessary changes, the official can click on the save button to make the changes permanent and would be redirected to the dashboard.

Source: 7.4

12.10 Prohibition of a low level user

First the admin's login shall take place and if verified, the admin official would be redirected to the admin dashboard. The admin has the full flexibility to ban a certain user applied there is a specific and accepted reason for that.
He/She would have to click on a button named " prohibit low level user " from where he would have the option to permanently or temporarily ban a user from the access to use that software. He would opt for either of them, in case of temporary ban a specified time would be asked to enter from the admin.
Then the admin would be asked to confirm the ban and the user would be banned and the admin would be redirected to the dashboard.

Source: 7.5

12.11 Maintenance

It is an optional feature present in the software which can be availed by the a admin.
First the admin's login shall take place and if verified, the admin official would be redirected to the admin dashboard from where the admin can switch on the maintenance mode.
This will eventually lead to the web server being offline till the admin wants and some specific problem has not been dealt with and resolved.

Source: 7.5

12.12 Creation/deletion of Staff Account

First the staff's login shall take place and if verified, the staff official would be redirected to the staff dashboard. Then the official can delete his/her account after clicking on the delete button. A confirmation would be asked which if checked correct and since the details have been already verified will lead to deletion of account.

Source: 7.4

Another way is from the admin's end. The admin after logging in if the verification succeeds have two options i.e. to create or delete staff accounts.
If the admin opts for creating an account, then he/she would be asked to add the username and email of the individual. Then the admin would give certain staff permission to him/her according to the needs of the position. A random password would be generated for the account and an email regarding the confirmation of the creation would be sent to the registered email.
If the admin opts for deleting a certain account, he would be asked to enter the credentials and a pop-up would be generated regarding the confirmation of the same. If confirmed, the account would be deleted.

Source: 7.5

12.13 Login

The process would be started by requesting the login url in the internet and then the login page would be rendered. The login page would be displayed and credentials would be asked to be typed in.
The basic credential validation would be run, if the credentials are found valid a connection would be instantiated with the database of the software. A query would be run at the database of the software.
If found invalid at the database, invalid error would be displayed and the user would be redirected to the form. I found valid, the login would be successful and access granted.

Source: 7.6

12.14 Registration

The process would be started by requesting the registration url in the internet and then the registration page would be rendered. The registration page would be displayed and credentials would be asked to be typed in.
The basic credential validation would be run, if the credentials are found valid, a uid would be generated for the user. Correspondingly a connection would be instantiated with the database of the software. A query would be run at the database of the software.
If the uid generated is unique, then the record would be created. Else, the "account exists error" would be displayed and the user would be redirected to the registration page.
After creating the record, success registration would be displayed and an activation mail would be generated and sent to the registered user email and also to the admin for security reasons.

Source: 7.7

12.15 Responding to Queries

First the staff's login shall take place and if verified, the staff official would be redirected to the staff dashboard. From here, the staff official can respond to the queries asked by the users over the platform. He/She can do so by clicking on the 'respond to queries' button and writing text over it.Then a notification would be generated and sent to the user having the query resolving the issue.

Source: 7.4

13. Explanation for Activity Diagrams

13.1 Login

First user clicks on login and the login page appears. The user types in the credentials and authentication for those credentials is done. If the credentials are correct, then the user is instructed to enter profile details and the dashboard of the user is rendered based on those details.

Activity Overview: Click on Login-> Login Page -> Type Credentials ->Enter Profile Details -> Dashboard

13.2 Signup

First user clicks on the sign up button then he has two options, whether he wants to sign up as a student or as a company. After that a sign up form is opened and then users have to give their information. After that we check whether the given email already exists or not. If email already exists then users have already existed in the database so he has already signed up, so he needs to login or it is also possible that the given email is wrong and it is already in the database, so in this case users need to be given the correct email and have to fill the form. After the form submission an OTP is sent to the user email id through which he has to authenticate himself. If the authentication was not successful then again OTP is sent to the user email id. After the successful authentication user redirects to the home page.

Activity Overview: Click on Signup-> Signup Page -> Type Credentials ->Verification page -> Home page.

13.3 Forgot Password

First the user clicks on the 'Forgot Password' button in order to obtain a new password for his/her already present account over the platform. He/She is asked to enter email address/username for verification. If credentials are found incorrect then the user is redirected to enter username/email address page and 'incorrect credentials'/'account not found' is displayed. If credentials are correct then OTP verification is sent over the registered email. If it doesn't pass then the OTP is resent. If it passes then New password is asked and confirm password field is filled and checked to be the same. The password has now been changed and the user is given an option to be redirected to either login or home page.

Activity Overview: Forgotten Password button -> Enter email/username -> Validation -> OTP verification ->Enter new password in password and confirm password fields -> Both password match -> Redirection to home/login page

13.4 Technical support:

For the technical support user need to the support page, if the user has not login then he redirect to the support page whether he has to give his email for the response if the user has already login then no need to provide the email. Then the user goes to the support message page where he needs to write a query. Then the support message has been generated, then it is checked whether the given email is already present in the database or not. If email is not present in the database then the support message is stored in the temporary database else if email is present in the database then the support message is stored in the database. Then the message goes to the technical support team. And the query goes to the waiting state from the support team. While at the same time in the dashboard of technical staff all the queries are shown, and he has the option to reply or ignore the message (if irrelevant is found). After that when technical staff reply the query then the response is sent to the given email for non login users and for the login user the reply is stored in the database and sent to the technical support chat page.

Activity Overview: Technical support page -> Generate message -> Send request -> Reply -> Mailed reply.

13.5 Update profile:

First user goes to the dashboard after the login page and then he goes to the profile page. There is a button to edit the profile, so after clicking on the edit button the user has to select the fields which he/she wants to edit. Then if the field which the user needs to edit requires verification whether the user is an authenticated person or not, in that case the user has to verify himself by his phone number, then fields are edited. Whether if the changing field does not require the user authentication then the fields are edited. After that if the user presses on the save button then the updated data is stored into the database and then the user redirects to the profile page where the updated profile is visible.

Activity Overview: Dashboard -> Profile Page -> Edit button -> Select field -> Verify authentication page (if required) -> Save -> Profile page

13.6 Apply for Internship

First the applicant logs into the system and reaches to the dashboard and clicks on ‘Apply for Internship’ button. Now the companies list who are hiring would be displayed. Now at least one company name should be selected by the applicant and then he/she can click on the apply button. If the user has not applied in that company earlier, an application form will be loaded which needs to be filled correctly. A CV needs to be uploaded for the Portfolio and valid details need to be filled in the form. If the details entered are incorrect then the user will be redirected to application form. If correct then the submit form button shall be clicked to confirm application.

Activity Overview: Dashboard -> 'Apply for internship' button -> Display list of hirings in companies -> Select companies of choice -> 'Apply' button -> Application form -> Fill the form and upload CV -> Validation of filled details -> Submit form

13.7 Delete Account

First the user needs to login and reach the dashboard. Then the user will click on the delete account button. Then requirements shall be checked and needs to be fulfilled. The permission needs to be checked. If the appropriate permissions are not present with the account owner, then an error message would be displayed i.e. "No permission". If the appropriate permissions are granted, then a final alert will be given to the user regarding deletion of his/her account, if the user agrees to it. Then the account deletion process will be started. If the web server is online during the process, then all the data will be deleted and the user will be redirected to the home page. He is now open to make a new account or leave the portal.

Activity Overview: Dashboard ->Click on Delete account ->Check requirements and permissions ->Final alert before deleting ->Start account deletion process if server is online ->Delete all data

13.8 Company Announcement

The activity starts from the dashboard. We are assuming that activities for successful Login have already been done. Now when the user has come to the dashboard he has a choice to do a new activity from here. Dashboard is always an option to quit the activity in between so we marked Dashboard an exit point in the diagram. Whenever the user (type: company) opts for Create a new announcement, he must have to click the Create announcement button. Now the user will have 2 options : announcement for result or announcement for vacancy. In both options some details will be fetched from the database which will be fetched and the user will have to fill the announcement form in which he has to tell some fields (eg: Topic, Round Number, Final List, Vacancies left etc-). He will be redirected to a next page where he will be asked to submit a soft copy of the result or any pdf/ word file regarding the announcement (This option will also be optional to the user since it's the user's choice to give the file or not). Finally when the user clicks the Post Announcement button then a new announcement will be created and the user will be redirected to the Create Announcement Page from where he had started.

Activity Overview: Dashboard -> Create Announcement Page -> Ask announcement type -> fetch user data required -> ask details about the announcement -> ask for soft copy (if any) -> POST Announcement -> Create Announcement Page

13.9 Create/ view notifications and Add blogs

Since we are going to merge 2 activities (notifications and blogs) in one diagram, we need to create 2 start points not because of two starting points but the reason is quite different : Normal User (student and company) can only see their notification, But High level User can not only see notifications but also give notifications. Moreover Blogs will

be added by the high level users. So, there are 3 parts in this activity diagram: view notification, give notification and add blogs which are discussed below:

- View Notification
  - **User Dashboard -> Click on view notifications -> (Automatically fetches the user notifications from the database) -> Show all the notifications (with sort and delete options) -> (May go to Dashboard)**
  - Now in this When notifications will be shown the notification tab would be the end state and before activity ends the user may also return back to the dashboard which itself is an end state. Moreover, on deleting a notification, the fetching process will begin once again.
- Give Notification
  - **Staff Dashboard -> Click on send notification -> fetch the users to send -> Ask the notification message -> Send notifications -> END**
  - It's quite a simple activity where the staff user will have to select the users whom he wants to send the notification and then he has to type the message. On clicking the send button, notification will be sent.
- Add Blogs
  - **Staff Dashboard -> Click on Add Blogs -> Fill the blog contents -> (Verification of the blog content anif verified then saving it to database) -> Post the Blog -> END**
  - This is very similar to the above case, the difference is that here blog content such as title, description etc needs to be filled and basic validations will be done such as no Non Empty content or Restricting some particular type of content etc. On clicking the Post button the blog would be posted.

13.10 Manage Staff Account

This activity includes 2 types of processes, one is creating a new staff account and second is delete staff account. Basically, for both the activities only the admin user is eligible. No other user would be allowed to access this webpage. So the 2 parts of this activity : new staff account and delete staff accounts are discussed as below:

- Delete Staff Accounts
  - **Admin Dashboard -> Manage Staff Accounts -> Choose one or more staff accounts -> Select the delete permissions -> Click on confirm delete -> (Delete operation to be applied on the selected accounts) -> END**
  - Whenever the delete activity is done, it's the admin's choice whether to delete the account permanently, ban the account, or shift the usability to another staff. Moreover on the final delete warning, admin can also opt to cancel the operation because once the account is deleted, it can’t be undone..
- Add New Staff Account
  - **Admin Dashboard -> Manage Staff Accounts -> New Staff Account -> Ask username and email -> (verify them) -> Ask the staff permissions -> Set the computer generated strong password and create the account -> Mail will go at the staff’s email about his account details -> END**

- When the verification is successful then the rest of the activity will be processed, until a unique username and email aren't entered, the account can't be created.

13.11 Account Prohibition

Only the admins have the access to prohibit any user (including other admins). The activity starts from the admin dashboard where admin goes to the user ban tab. Where he could select any user and ban him. While banning the webpage will ask the admin to choose ban type. Now if admin chooses to ban temporarily then admin also needs to give the number of days for which the user will be banned. Finally the admin would be prompted before the final ban. On cancel user would not be banned, but on clicking ban button user would be banned and would not be able to login until he is unbanned. Moreover if he's already logged in, automatically he will be logged out.
Activity Overview : Admin Dashboard -> Click Prohibit User -> Select a user to ban -> Choose ban type -> (Ask ban time if needed) -> Final Alert before banning the user -> Take over all permissions -> Ban the user -> END

Note: In some awkward situations, the superuser account comes into picture which can purely access anything. So, the superuser account is not given to a single user rather it could be accessed when a certain defined user permits its use. The result of this note is that superuser can prohibit the admins too but viceversa is not possible.

14. Traceability Matrix

Requirement ID	Use Case ID	High Level Design Diagrams			Low Level Design Diagrams		Test Case ID	Function
					Level 1	Level 2		
		State ID	Data Flow ID	ER ID	Activity Diagram	Sequence ID / Sequence Instance ID		
R1	U4, U5, U19	ST13	D1	ER1, ER2, ER4	A1	SE1, SE3, I1, I2, I3, I4, I5, I6, I7	T1, T2, T3, T4, T5, T6, T7, T8, T9	P5, P6, P10, P11, P12
R2	U4, U5, U19	ST14	-	ER1, ER2, ER4	A2	SE1, SE2, SE6, SE9, I1, I2, I3, I4, I5, I6, I7	T1, T2, T10, T11, T12, T13	P2, P3, P6, P7, P9, P15
R3	U5, U6	ST5	-	ER1, ER2, ER4	A3	SE1, SE4, SE9, SE12, I1, I2, I5, I6, I7	T1, T2, T10, T11, T12, T13, T14	P2, P3, P8, P10
R4	U3, U7, U15, U16, U20	ST1,ST2,ST6, ST3	D4	ER4	A9	SE1, SE6, I1, I2, I5, I6, I7	-	P20
R5	U5, U19, U22, U23	ST5, ST12, ST15	D3, D5, D10, D11	ER6	A10	-	T1, T2, T8, T9, T15	P17
R6	U15, U20	ST3, ST8, ST9	D6, D9	ER5, ER7	A8, A9	SE7, SE10, SE11, I2, I3, I5, I6, I7	T1, T2, T18	P16, P19
R7	U21, U22	ST10	D11	ER6	A7, A11	SE1, SE12, I3, I5, I6, I7	T1, T2, T7	P18
R8	U2, U8, U9, U10, U11, U16, U17	ST6	D2	ER2, ER4	A5	SE1, SE5, SE6, SE9, SE12, I1, I2, I5, I6 , I7	T1, T2	P2, P4, P10, P13, P14, P21
R9	U8	ST8	D7(Minor), D8	-	A9	SE10, I1, I2, I3, I4, I5, I6, I7	T1, T2, T16	P19
R10	U18	ST4, ST15	D12	-	A4	SE10, I1, I2, I3, I4, I5, I6	T1, T2, T17	P23
R11	U5, U23	ST7,ST12	D3	-	A7	SE12, I1, I2, I3, I4, I5, I6, I7	T1, T2	P1, P10, P22
R12	U14	ST2, ST14	D5	ER3	A5	SE8, I2, I5, I6	-	P10, P21

Reference Tables

ID	Requirement
R1	Login
R2	Signup
R3	Forgot Password
R4	View company details/ fill application form
R5	Manage Staff account
R6	Blogs and announcements
R7	Prohibit user
R8	Edit profile
R9	View notifications
R10	Technical support
R11	Delete my own account
R12	Access student CV

ID	Use Case
U1	Portfolio Reviews
U2	Update CV/Portfolio
U3	Roll back application form
U4	Login/Signup
U5	Password Authorisation
U6	Forgot password
U7	View Company Detail
U8	View/Give notification
U9	Manage internship details
U10	Add company
U11	Update company detail
U12	Manage Students Interview
U13	Add/Delete review
U14	Access student CV/Portfolio
U15	Result announcement
U16	Interview information
U17	Update detail profile
U18	Technical support
U19	Account authorisation
U20	announcement/blog
U21	Prohibit Student/Supervisor/Company Professional
U22	Manage account
U23	Delete account
U24	Security & Maintenance

ID	ER Diagram
ER1	User
ER2	Student
ER3	Portfolio Review
ER4	Company
ER5	Result
ER6	Staff/ Admin
ER7	Blogs

ID	Sequence Diagram Instance
I1	Student
I2	Company Professional
I3	Administrator
I4	Supervisor
I5	Web Interface
I6	Web Server
I7	Database

ID	DFD
D1	Login
D2	Edit Profile
D3	Delete Account
D4	View Company Details
D5	View CV of other student
D6	Company Announces Result
D7	Generate Notifications
D8	Float Notifications
D9	Add Blogs
D10	Create new staff account
D11	Prohibit user
D12	Technical Support

ID	State Diagram
ST1	Rollback application form
ST2	Applying for internship
ST3	Announcement creation by company professional
ST4	Technical support and response
ST5	Forgot password
ST6	Profile updation
ST7	Account deletion
ST8	notifications
ST9	blogs
ST10	Prohibition of a low level user
ST11	Maintenance
ST12	creation/deletion of Staff Account
ST13	login
ST14	Registration
ST15	Responding to queries

ID	Sequence Diagram
SE1	Web page display
SE2	registration
SE3	login
SE4	Change password
SE5	Upload portfolio/cv
SE6	Fill application form/profile detail
SE7	Uploading interview result
SE8	View applicant detail
SE9	Student/company verification
SE10	notification /Announcement creation
SE11	reviews
SE12	Update user detail

ID	Activity Diagram
A1	login
A2	signup
A3	Forgot password
A4	Technical support
A5	Update profile
A6	Apply for internship
A7	Delete account
A8	Company announcement
A9	Create / view notifications and add blogs
A10	Manage staff account
A11	Account prohibition

ID	Unit Test Case
T1	setUp
T2	tearDown
T3	test_correct_username_password
T4	test_correct_email_password
T5	test_wrong_username
T6	test_wrong_password
T7	test_account_ban
T8	test_staff
T9	test_admin
T10	test_email_already_exists
T11	test_username_already_exists
T12	test_time_verification_for_otp
T13	test_send_otp
T14	test_change_password
T15	test_check_permissions
T16	test_create_notification
T17	test_technical_support
T18	test_create_blogs

ID	Functions
P1	Redirect to home page
P2	Generate OTP
P3	Send OTP to email
P4	Send OTP to phone
P5	Login
P6	Logout
P7	Signup
P8	Forgot Password
P9	Resend OTP to email
P10	User Identification
P11	Dashboard detection error check
P12	Redirect to dashboard page
P13	Redirect to profile page
P14	Resend OTP to phone
P15	Profile verification
P16	Blogs
P17	Manage staff account
P18	Ban Users
P19	Manage Notification
P20	View Company Details/ Fill Application form
P21	Profile
P22	Delete My Account
P23	Technical Support

Pseudocode

15.1 Redirect To Home Page

Method home (request):

Request for home page
Load the home page with the details required in it.
Return the requested page

15.2 Generate OTP

Method generate_otp():

Store the digits, lowercase and uppercase alphabets in an array
Create a empty string
Initialize a variable with 0
Initialise a object of class random
while(variable is less than 7):
    Insert a random element from the array into the string using the object
    Increment variable by one
endwhile
return string generated

15.3 Send OTP to Email

Method SendMail (mail_subject, message, recipient_email):

try:

Fetch any available host email address from hosts list in the settings file.

Reset the recipients list to NULL.

Add the recipient_email in the recipients list.

Get the dummy (before successful verification) / verified profile from the database.

Generate a new OTP using generate_otp method.

Store the new otp into the database.

Store the time of otp generation.

Give a time limit after which it will automatically expire.

Lock the recipient list until OTP has not sent to recipient_email in order to avoid multiple emails and otps.

Initiate an email object from the inbuilt library.

Call its inbuilt method send_mail.

Pass the mail_subject, message and recipient_email as the parameters.

It will automatically send the email to recipient_email.

Unblock the recipient_list so that it can be edited later by other methods.

return True

except:

return False

15.4 Send OTP to Phone

Method Send_OTP_to_Phone (request, mobile_number, country_code, message):

try:

Taking help of Mobile OTP Sender Twilio,
Get the Sid and Auth Token of the twilio account from settings file.
Reset the recipient list to NULL.

Add the mobile number passed in the method as a one and single
Recipient.

Generate a new OTP using generate_otp( ) method.
Store the OTP to the database.
Store the time of otp generation.
Give a time limit after which it expires.

Lock the recipient list until OTP has not been send to the recipient phone
number in order to avoid multiple messages and otps on phone number.
Create the message using the object of twilio library.
Pass its body as a message and otp.
Pass recipient as the complete recipient list with country code.

Fetch an available host number from settings file,
Add it to the sender phone number with country code.
Send the message.

Unblock the recipient list so that it can be edited later by other methods.
Return True

except:

Return False

15.5 Login

Method login_request(request):

```
if (user id already logged in):
    redirect the user to the dashboard
endif
if (request type is 'POST')
    fetch the username and password from the form
    if (username is entered)
        if(username or password is incorrect)
            redirect user to login page
            display error on the screen
        endif
    else
        if (email id is entered)
            if (email id and password are incorrect)
                redirect user to login page
                display error on the screen
            endif
        else
            redirect user to login page
            display error on the screen
        endif
    endif
    if (user credentials have not been verified)
        redirect to login page
        Display error message on the screen
    endif
    if (user account is permanently banned by authorities)
        redirect user to login page
        notify the user about the ban
    endif
    if (user account is temporarily banned)
        calculate the time since the account was banned
        if (this time is less than the ban time)
            redirect user to login page
            notify the user that the ban is not ended
        else
            remove ban from the account
            save the account settings
        endif
    endif
    call the inbuilt function login
    pass the request and the user credentials as parameters
```

redirect user to his dashboard
else
redirect the user to the login page
display the error message on the screen
endif
return NULL

15.6 Logout

Method logout_request(request):

if (user is logged in)
call the inbuilt logout function
Pass the request to this function as parameter
endif
return home(request)

15.7 Signup

Method signup(request):

if (user is authenticated == True):
Redirect to Dashboard
else:
Render signup page.
Ask the user whether he wants to register as a student or as a company.
endif
return NULL

Method signup_verify(request):

Check if this request is POST or not.
if (request != “POST”):
Redirect to the signup page and ask them to fill in the details.
endif

Invoke the email_already_in_use method.
It checks whether an account with a given email already exists or not.
if (email_already_in_use(email) == True):
Render signup page and ask for a different email address.
endif

Invoke the username_already_in_use method.
It checks whether an account with a given username already exists or not.
if (username_already_in_use(email) == True):
Render signup page and ask for a different username address.
endif

Fetch the form details using the form method.
Check form is valid or not.
if (form is invalid):
    render the signup page and show the error encountered.
endif

Save the form entry.
Create a dummy user account by inactivating the account until email verification has been done.

Call the Send OTP to Email method and send an OTP to it.
if (send_otp==False):
    Render signup page and show that the email servers are busy
    So, otp can't be sent at this time.
endif
Render Signup Page.
Move on to the OTP Verification Phase by rendering its form.

Method signup_verify_otp(request):

Check if this request is POST or not.
if (request != "POST"):
    Redirect to the signup page and ask them to fill in the details.
endif

Fetch the data from post request
Invoke the email_already_in_use method.
if (email_already_in_use(email) == False):
    Render signup page and ask for a registered email address.
endif

Get the profile of the user from its email address from the data
Match the otp stored in the profile with otp in the post data.
if (otp is not matched):
    render otp verification page and again ask the otp
endif

Store the otp send time in a variable (say prev_time)
Store the current time in a variable(say current_time)
Store their difference in a variable(say time_delta as current_time - prev_time)
Convert it time_delta to minutes.
if (time_delta>(OTP Expire Time Limit)):
    Resend OTP to the email address
    Render the otp verification

Tell the user that Time limit has been exceeded.
A new OTP has been sent
endif

Activate the user account.
Set the OTP to NULL so that it can't be misused.
Update the registration date time to current_time.
Send the request to backend staff that a registration has come.
Save the changes to the database and commit them.
Render the last phase of signup
Tell the user that the account has been successfully created.
They can login once backend staff permits their account

15.8 Forgot Password

Method forgot_password(request):

```
if (request type is 'POST')
    fetch the email id of the user from the form
    try:
        if (user account is not active)
            redirect the user to forgot password page
            display error message on to the screen
        endif
        if (user account with such mail id does not exist)
            redirect the user to forgot password page
            display error message on to the screen
        else
            retrieve the user profile
        endif
        if (user account is not verified)
            redirect the user to forgot password page
            display error message on to the screen
        endif
        if (user account is permanently banned)
            redirect the user to forgot password page
            notify the user about permanent account suspension
        elseif
        if (user account is temporarily banned)
            calculate the time since the account was banned
            if (this time is less than the ban time)
                redirect user to forgot password page
                notify the user that the ban is not ended
            else
                remove ban from the account
```

save the account settings
redirect the user to OTP verification page
endif
endif
if (forgot_password_send_otp(email,user) returns false)
redirect user to forgot password page
display error message on to the screen
endif
except:
if (account with such details doesn't exist)
redirect user to forgot password page
display error message on to the screen
endif
else
redirect user to forgot password page
display error message on to the screen
endif
return NULL

Method forgot_password_send_otp(email,user):

try:
generate the OTP using the generate_otp() method
store the subject of the mail on to a string
store the message of the mail on to a string
concatenate the message with the OTP generated
call the inbuilt function sendmail
pass the subject, message, and email id as arguments to this function
store this generated OTP in to the user credentials for further verification
save the changes in the user credentials
return true
except:
return false

Method forgot_password_verify_otp(request):

if (request type is 'POST')
fetch email id and the otp entered by the user from the form
try:
if (OTP sent to user is equal to OTP entered by the user)
if (duration between when OTP was sent to user and
entered by him is greater than 1 minute)
redirect the user to forgot password page
display the error message on to the screen
else

encrypt the OTP present in user credential
save the changes in user credentials
redirect the user the password change page
endif
else
redirect the user to forgot password page
display the error message on to the screen
endif
except:
redirect the user to forgot password page
display the error message on to the screen
else
redirect the user to forgot password page
endif
return NULL

Method forgot_password_resend_otp(request, email id):

Convert the email id to string
try;
if (OTP is not received by the user)
redirect the user to forgot password page
display error message on the screen
else
send OTP again using using forgot_password_send_otp() function
endif
except;
redirect the user to forgot password page
return NULL

Method reset_password(request):

if (request type is 'POST')
fetch the email id of the user from the form
fetch the password entered by the user on the form
try;
reset the user password to the new password entered by user
save the user credentials
except;
redirect the user to home
return NULL
store the subject of the mail for resetting the password in a string
store the message of the mail for resetting the password in a string
call the inbuilt function sendmail to send a mail
pass subject, message and email id as arguments

redirect the user to forgot password page
else
redirect the user to forgot password page
endif
return NULL

15.9 Resend OTP to Email

Method Resend_OTP_to_Email(email):

Search for the email address in the list of all users type.
if (search==True):
Get the profile of the user from the Profile table in the database.
Call the Send OTP method.
Pass mail subject, message and email as the parameters.
if (send_otp(*parameters) == False):
Render the page from where resend otp request has been generated.
Show the error no email is free to send the otp.
Ask them to wait for a while.
else:
It needs to go for the next process of OTP Verification.
return render(html web page)
(render the next phase to ask them otp)
endif
else:
Redirect to Signup Page
Show error that no email has been given where otp can be sent.
(This error comes when attackers try to signup without emails or someone just created a get request to send otp to emails)
endif
return NULL

15.10 User Identification

Method user_type_checker(request, user, email):

Fetch the user using email if user is not NULL or user is always a user as:
if (user == NULL):
user = fetch user from email
endif

Fetch the profile of this user
if (error encountered == True):
Give a http-response of error that”profile not found”
endif

if (user is Student):
return “an user account”

```
elif (user is Staff):
        return " a Staff account"
else:
        return " a company account"
endif
return NULL
```

15.11 Dashboard Detection Error Check

Method error_detection(request,request_form_profile_page):

```
if (requested user id is not authenticated == True):
        redirect to login page
endif

if (requested user is associated with staff account):
        cancel all error detections
        return false (meaning np error detected)
endif

if (email associated with account is not verified == True):
        Logout()
        redirect to login page
        Notify the user to first verify his/her email address.
endif

if (profile associated with email not found == True);
        Give http-response of profile not found
endif

Fetch the profile according to the company or student account.
(not admins do not profile)

if (user account is banned permanently == Ture):
        Logout()
        Redirect to login page
        Notify user than account has been banned permanently
        They can challenge the ban in the ban section by filling the form and pay
        verification fees.
endif

if (user account is banned temporarily == False)
        Give http-response page
```

Suggested them to logout themselves and login again if they think at this time the ban should be removed.

endif

if (profile has not filled == True AND request_from_profile_page==False):

Redirect to profile page

Ask them to fill tehri profile first before accessing dashboard

It includes phone number verification

endif

return False

15.12 Redirect to Dashboard Page

Method dashboard():

Call the Dashboard Redirection Error Check method.

if (error_detection(* parameters) == False):

Render the Dashboard page.

else:

Return/Render the error detected by error_detection method.

endif

return NULL

15.13 Redirect to Profile Page

Method Profile(request):

if (error_detection(request, True)==False):

return profile_with_data_fetcher(request)

else:

return the error detected by error detection method.

endif

Method Profile_with_data_fetcher(request):

Assume that contact number has been given by the user and it's verified too.
In order to do it initialize a variable (say contact_give as True)
Now recursively fetch in Company Profile that user exists or not
if (found then check its contact) :

Fetch and store the user data

else:

recursively fetch in Student Profile that user exists or not

if (found then check its contact):

Fetch and store the user data

else:

give httpresponse that profile not found

endif

endif

render the profile page
if (profile filled==True):
    then give option to edit profile
else:
    ask them to verify the sensitive profile details(example : mobile number)
endif

15.14 Resend OTP to Phone

Method Resend_OTP_to_phone(phone_number):

Search for the phone numbers in the list of all users profiles(verified or unverified).
if (search==False):
    Get the profile of the user from the Profile table in the database.
    Call the Send OTP to phone method.
    Pass mail subject and phone number as general parameters.
    Pass the country in string format.
    if (send_otp_to_phone(*parameters) == False):
        Render the page from where resend otp request has been generated.
        Show the error no phone number is free to send the otp.
        Ask them to wait for a while
    else:
        It needs to go for the next process of OTP Verification.
        return render(html web page)
        (render the next phase to ask them otp)
    endif
else:
    Give an http response or redirect to page from where request had come.
    Tell them that the phone number has already registered.
Endif

15.15 Profile Verification

Method Fetch_Profile(request):

if (user is logged not in):
    Redirect to home page
endif

Check if this request is POST or not.
if (request != “POST”):
    Redirect to the profile page and ask them to fill in the details.
endif

Fetch the profile details according to the user account.
Phone Number will be common to every user profile.
In the user database search that any user has same phone number or not
if (same number found):
    Redirect to profile page and ask them to enter another phone number

endif

Call the otp sender to phone method and pass phone number as parameter.
if (error_occured==True):
    Redirect to dashboard.
endif

Update the profile according to the user account and data passed in the form.
Still profile_filled will be False because phone number is not verified.
Save and Commit changes in the database.
Render the next phase that is otp verification on the number given.

Method Profile_Phone_OTP_verify(request):

if (user is logged not in):
    Redirect to home page
endif

Check if this request is POST or not.
if (request != “POST”):
    Redirect to the profile page and ask them to fill in the details.
endif

Match the OTP entered by user and OTP sent to phone number
if (matched==False):
    Redirect second phase of OTP Verification
    Display error that OTP not matched
endif

Compare the time limit of when otp was sent and current time.
if (time_delta>time limit):
    Resend a new otp to same phone number and save its time.
    Redirect second phase of OTP Verification
    Display error that Time limit has exceeded, new OTP has been sent
endif

Fetch the profile again
Change the value of profile_verified to True
So that dashboard can be accessed.
Finally Redirect user to dashboard
Display success message that profile has been verified.

15.16 Blogs

Method manage_blog(request):

if (user is admin):
    redirect the user to the dashboard

Else
    Return

Display " Blogs"
Display box "Show Entries" with a drop down facility
    Fields : 10,25,50,100
Display backend data table with attributes : Blog ID, Title, Short Description, #
Display Pages for the requested no. of entries

Method create_new_blog(request):

if (user is admin or staff account with required permissions):
    redirect the user to the dashboard
Else
    Return

if(click on '+New Blog' == true)
    Render form(Fill Out Blog Details)
    if((all details != NULL) and (click on "create blog" == true))
        Copy the details to the backend database
        Redirect to manage_blog page
    Endif
Endif

Method delete_blog(request,item):

if (user is admin or staff account with required permissions):
    redirect the user to the dashboard
Else
    Return
search(Blog ID);
if(search == true)

    if(click on 'delete' == true)
        Delete the details from the backend database
        Redirect to manage_blog page
    Endif
Endif

Method edit_blog(request,item):

if (user is admin or staff account with required permissions):
    redirect the user to the dashboard
Else
    Return
search(Blog ID);

```
if(search == true)
    if( click on 'Blog ID' == true)
        Render form(Update Blog Details)
        if((all details != NULL) and (click on "update blog" == true))
            delete the details from the backend database
            Copy the details to the backend database
            Redirect to manage_blog page
        Endif
    Endif
Endif
```

15.17 Manage Staff Accounts

Method manage_staff_accounts(request):

```
if (user is admin):
    redirect the user to the dashboard
Else
    Return

Display " Accounts"
Display box "Show Entries" with a drop down facility
    Fields : 10,25,50,100
Display backend data table with attributes : Account ID, Username, Email, Name
Display Pages for the requested no. of entries
```

Method edit_staff_permissions(request,item):

```
if (user is admin):
    redirect the user to the dashboard
Else
    Return
search(Account ID);
if(search == true)
    if( click on 'Account ID' == true)
        Render form(Update Account Details)
        if((all details != NULL) and (click on "update permissions" == true))
            delete the details from the backend database
            Copy the details to the backend database
            Redirect to manage_staff_accounts page
        Endif
    Endif
Endif
```

Method create_new_staff_account(request,item):

```
if (user is admin):
```

redirect the user to the dashboard
Else
Return

if(click on '+New Account' == true)
Render form(Fill Staff Details)
if((all details != NULL) and (click on "create staff account" == true))
Copy the details to the backend database
Redirect to manage_satff_accounts page
Endif
Endif

15.18 Ban Users

Method restrict_users(request):

if (user is admin):
redirect the user to the dashboard
Endif
Else
Return

Display " Restrict Single user at a time"
Display box "Show Entries" with a drop down facility
Fields : 10,25,50,100
Display backend data table with attributes : UserID, Email, Username, Name, User type, Ban
Display Pages for the requested no. of entries

Method ban_user_account-permanent(request,item):

if (user is admin):
redirect the user to the dashboard
Endif
Else
Return
Call restrict_users(request)
search(User ID);
if(search == true)
Display the entry with all field in restrict-users database;
if((click on 'P' == true)
display(dialogue box "Ban User permanently")
if(click on 'ok' == true)
display(1 user banned permanently)
copy the entry to the un-restrict users database
delete the entry from -restrict users database

Else

Redirect to restrict_users

Else

Return NILL;

Method ban_user_account-temporary(request,item):

if (user is admin):

redirect the user to the dashboard

Endif

Else

Return

Call restrict_users(request)

search(User ID);

if(search == true)

Display the entry with all fields in restrict-users database;

if((click on 'T' == true)

display(dialogue box "Ban User temporarily for some days")

if(click on 'ok' == true)

display(1 user banned temporarily)

copy the entry to the un-restrict users database

delete the entry from -restrict users database

Else

Redirect to restrict_users

Else

Return NILL;

Method delete_staff_account(request,item):

if (user is admin):

redirect the user to the dashboard

Endif

Else

Return

Call restrict_users(request)

search(User ID);

if(search == true)

Display the entry with all fields in restrict-users database;

if((click on 'D' == true)

display(dialogue box "This staff account will be deleted")

if(click on 'ok' == true)

display(This staff account is deleted)

delete the entry from -restrict users database

Else

Redirect to restrict_users

Else

Return NILL;

Method unban_user_account-permanent(User ID):

if (user is admin):

redirect the user to the dashboard

Endif

Else

Return

Call un-restrict_users(request)

search(User ID);

if(search == true)

Display the entry with all fields in un-restrict-users database;

if((click on 'D' == true)

display(dialogue box "This user account will be unbanned")

if(click on 'ok' == true)

display(This user account is unbanned)

delete the entry from un-restrict users database

Else

Redirect to un-restrict_users

Else

Return NILL;

15.19 Manage Notifications

Method float_notifications(request):

if (user is admin or staff account with required permissions):

redirect the user to the dashboard

Else

Return

Display " User Accounts"

Display box "Show Entries" with a drop down facility

Fields : 10,25,50,100

Display backend data table with attributes : User ID, Email, Name, #

Display Pages for the requested no. of entries

if(click on 'To all Students')

Display Dialogue Box("Send notification to all students")

if("Enter text" != NULL and (click on 'send' == true))

Redirect to 'Float Notifications' page

```
        Endif
    Endif

    if(click on 'To all Companies')
        Display Dialogue Box("Send notification to all companies")
        if("Enter text" != NULL and (click on 'send' == true))
            Redirect to 'Float Notifications' page
        Endif
    Endif

    if(click on 'To all Staff')
        Display Dialogue Box("Send notification to all staff")
        if("Enter text" != NULL and (click on 'send' == true))
            Redirect to 'Float Notifications' page
        Endif
    Endif

    search('User ID')
    if(click on 'Send Notification' == true and (search == true))
        Display Dialogue Box("Send notification to this.userID")
        if("Enter text" != NULL and (click on 'send' == true))
            Redirect to 'Float Notifications' page
        Endif
    Endif
```

Method delete_notifications(request):

```
    if (user login == true):
        redirect the user to the dashboard
    Else
        Return

    if(click on 'profile_picture' == true)
        if(click on 'notification' == true)
            Display all notifications from database
            if(click on 'delete notification' == true)
                Delete notification from database
                Redirect to notification page
            Endif
        endif
    Endif
```

15.20 View Company Details / Fill Application Form

Method show_companies(request):

```
if (no error in the user profile)
        if (user is staff or super user)
                redirect the user to the home
        endif
        get user detail
        if(user have already got internship)
                display error that he can't register for internship
        endif
        get eligible company for round one
        return user to company page
endif
return error_detection(request,1)
```

Method show_companies_round_details(request, item):

```
if (no error in the user profile)
        if (user is staff or super user)
                redirect the user to the home
        endif
        get company round details
        return details
endif
return error_detection(request,1)
```

Method register_student_first_round_only(request, item):

```
if (no error in the user profile)
        if (user is staff or super user)
                redirect the user to the home
        endif
        get user detail
        if(user have already got internship)
                display error that he can't register for internship
        endif
        get eligible company for round one
        get announcement details
        get student data
        get company data
        if(there is no any announcement)
                display error
        endif
        if(student not satisfied min CGPA criteria)
                display the message
```

```
            endif
            if(user have already registered)
                display the message
            endif
            register user for round one
            get user detail
            get eligible company for round one
            return user to company page
        endif
        return error_detection(request,1)
```

15.21 Profile

Method staff_profile(request):

```
    if (requested user is staff and authenticated)
        if (request type is 'POST')
            get first name and last name of user
            save it as username
            redirect to the profile page
        endif
        else
            redirect to the dashboard
        endif
    endif
    else
        redirect to the dashboard
    endif
```

Method student_profile_3(request):

```
    if (requested user is authenticated)
        if (account is not a company account)
            get first name and last name of user
            save it as username
            get user address, gender, CGPA
            redirect to the profile page
        endif
        else
            redirect to the dashboard
        endif
    endif
    else
        redirect to the dashboard
    Endif
```

Method company_profile_2(request):

```
if (requested user is authenticated)
        if (account is a company account)
                get company name
                save it as company name
                get company address
                redirect to the profile page
        endif
        else
                redirect to the dashboard
        endif
endif
else
        redirect to the dashboard
endif
```

Method company_profile_3(request):

```
if (requested user is authenticated)
        if (account is a company account)
                create a form for change company photo
                if(form is valid)
                        update profile photo
                        redirect to profile
                endif
                else
                        redirect to profile
                endif
        endif
        else
                redirect to the dashboard
        endif
endif
else
        redirect to the dashboard
endif
```

Method student_profile_2(request):

```
if (requested user is authenticated)
        if (account is neither a company account nor a staff and superuser account)
                create a form for change user photo
                if(form is valid)
                        update profile photo
                        redirect to profile
                endif
                else
```

redirect to profile
endif
endif
else
redirect to the dashboard
endif
endif
else
redirect to the dashboard
endif

Method student_profile_1(request):

if (requested user is authenticated)
if (account is not a company account)
create a form for change user CV
if(form is valid)
update CV
redirect to profile
endif
else
redirect to profile
endif
endif
else
redirect to the dashboard
endif
endif
else
redirect to the dashboard
endif

Method student_company_number(request):

if (requested user is authenticated)
if (request type is 'POST')
get contact number
if(contact number of company or student already exist)
display number already exist
endif
get user detail
if(account is a company account)
if(OTP is not sent successfully)
redirect to dashboard
endif

```
                    get company detail
                else
                    if(OTP is not sent successfully)
                        redirect to dashboard
                    endif
                    get student detail
                endif
                save updated phone number
                redirect to profile page
            endif
            else
                redirect to the dashboard
            endif
        endif
        else
            redirect to the homepage
        endif
```

15.22 Delete My Account

Method delete_account(request):

```
if (no error in the user profile)
    get user detail
    if (user not found)
        give error message
    endif
    delete user account
    give message of successful account deletion
    send email to the user
    redirect to the home page
endif
return error
```

15.23 Technical Support:

Method technical_support(request):

```
Call the error detection method
If it does not return True then return profile page
If method!=”POST”:
    Return Support page with previous support data
Else:
    Fetch the message and support_id from form
    If support_id==0:
```

Create a new support thread

Else:

Reply to the given support id

Give error if comes

Return redirect('technical_support')

Method technical_support_assist(request):

Check for the permissions of requested logged in user

If permission record not found then create a new one and

Redirect user to dashboard

if permissions.can_manage_technical_support==False:

return error(request,"No permission")

If method!="POST":

Get all the unresolved tech responses.

Return assist_technical_support page with required data

Else:

Fetch the message and support_id from form

Create the reply support message for the support_id

Redirect to Respond_support Page for the support_id

Method respond_support(request,item):

Check for the permissions of requested logged in user

If permission record not found then create a new one and

Redirect user to dashboard

if permissions.can_manage_technical_support==False:

return error(request,"No permission")

try:

support=TechnicalSupportRequest.objects.get(id=int(item))

except:

return error(request,"Support Details Not Found")

if support.continued_support==True:

return error(request,"Support Details Not Found")

threads=get_all_threads(int(item))

Render the show_all_threads page with the required details