UNIVERSITY ADMISSION SYSTEM (WEB APPLICATION)

Revision Number: 1.0

Last date of revision: 20/08/2022

Document Version Control

Date Issued	Version	Description	Author
20-08-2022	1	Initial DPR — V1.0	A Singh

Contents

Abstract

A huge crowd of students can be seen in the university during the Admission seasons. This leads to the wastage of time, energy, money and sometimes it also impacts the health of the students and their parents. This process of admission in the university is very complex and time consuming for the faculty to process this huge number of applications submit by the student.

The university admission system is designed to solve all the problem mention above by utilizing the online system to track and process the applications submitted by the student. The student will no longer required to visit any university for the admission. Student will be able to submit the application for any course from anywhere. Once the application is submitted by the student, faculty member can review the application and take the appropriate action.

1 Introduction

1.1 Why this DPR Document?

1. The document provides detailed information related to end-to-end project implementation including Architecture flow diagram, Use Case diagram and User IO flow diagram

1.2 Scope

The DPR documentation will provide information on below mentioned areas:

- Application architecture
- Use Case diagram
- User IO flow diagram
- Database
- Technology
- Deployed Server Name
- Navigation

1.3 Definitions

Term	Description
IDE	Integrated Development Environment
Database	Collection of all the information monitored by the system
Heroku	Heroku cloud service

2 General Description

2.1 Product Perspective

The University Admission System is developed in python using flask to streamline the process of admission in the university. This is done by optimizing the process of application submission by the student and review of the application by the faculty. It uses the flask login for user authentication and authorization. The application will use Cassandra database for storing all the data.

2.2 Problem statement

Create a desktop application by which students can apply for any ongoing course of a university. It should have below mentioned features.

- Faculty registration.
- Student Registration.
- Application submission.
- Management of course.
- Example:
  - Adding a new course.
  - Modifying an existing course.
  - Removing an existing course.
  - Scheduling various courses.
- Check application status regarding approval/disapproval, if the application is approved by the faculty, student can see their scheduled interview appointment.
- Based on interview performance faculty will select or reject the application.
- Functionality for faculty to review any application submitted by the student for a course.
- Students must receive a notification either on phone/ email from the University Admission System application regarding the progress of their applied courses.

2.3 Proposed Solution

The application will have two types of users Faculty and Students. Based on the user type the application will have two flow.

1. Faculty: Faculty type user will have access to the dashboard from where he/she can do the course management and application review and management.

2. Student: Student type user will have access to the dashboard where he/she can apply for the new course, see the list of already applied course and check the status of the applied course.

2.4 Further Improvements

The University Admission System can be enhanced further in future to include the below mention features:

1. Exam Schedule
2. Test Management.
3. Result Management.
4. Attendance Management.
5. Faculty Management.
6. Student Management.

2.5 Technical Requirements

Create a desktop application by which students can apply for any ongoing course of a university. It should have below mentioned features.

- Faculty registration.
- Student Registration.
- Application submission.
- Management of course.
- Example:
  - Adding a new course.
  - Modifying an existing course.
  - Removing an existing course.
  - Scheduling various courses.
- Check application status regarding approval/disapproval, if the application is approved by the faculty, student can see their scheduled interview appointment.
- Based on interview performance faculty will select or reject the application.
- Functionality for faculty to review any application submitted by the student for a course.
- Students must receive a notification either on phone/ email from the University Admission System application regarding the progress of their applied courses.

2.6 Tools used

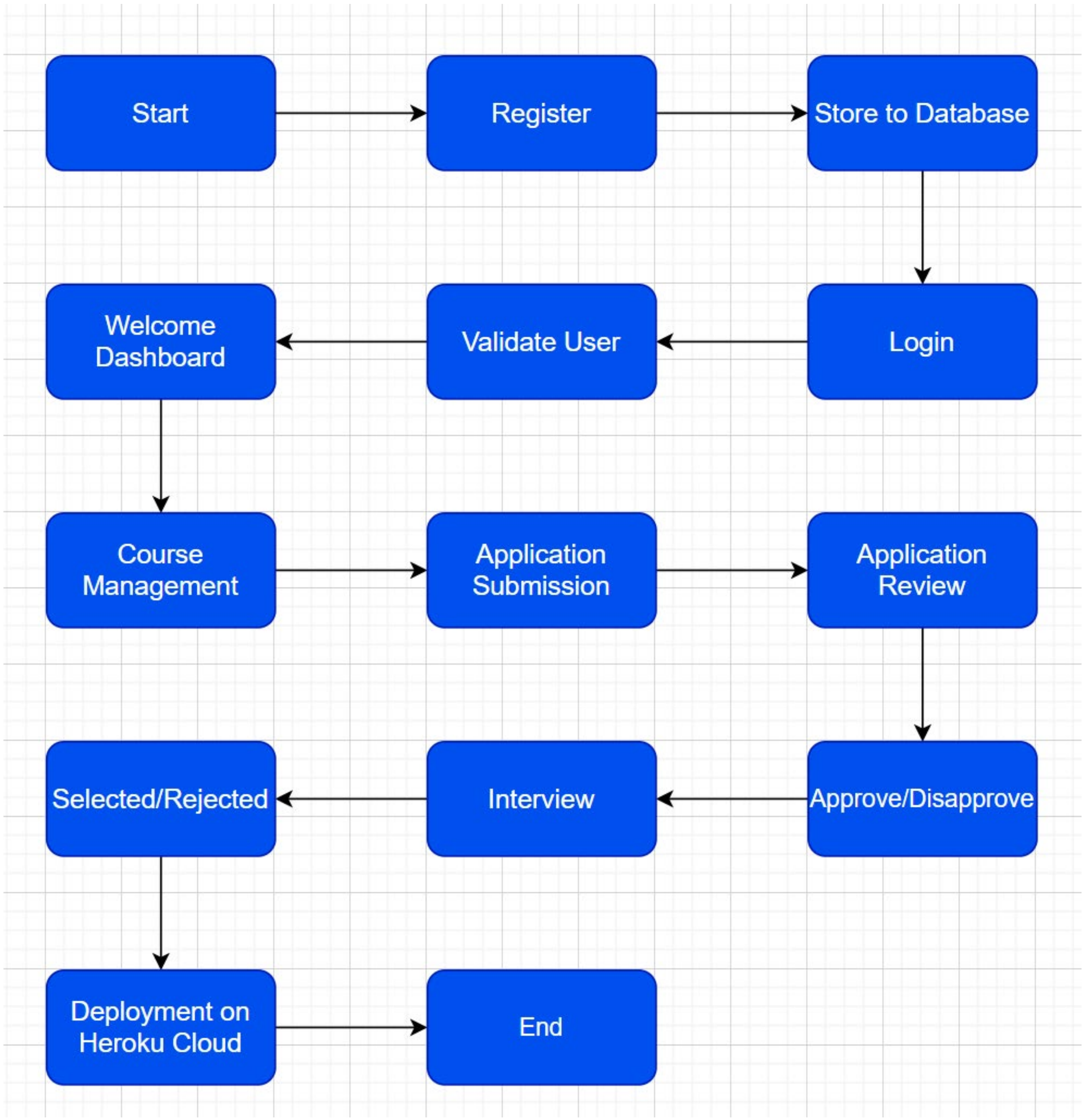
• Programming Language:	Python, HTML, CSS, JavaScript.
• Packages/Framework Name:	Flask, Flask-Login, Flask-Mail.
• Database:	Cassandra.
• IDE:	PyCharm.
• Repository:	GitHub.
• Front End:	HTML, CSS, JavaScript.
• Back End:	Flask.
• Version Control:	Git
• Cloud Service:	Heroku

2.7 Assumptions

The main objective of the project is to implement the use cases as mentioned in the Problem Statement under section 2.2.

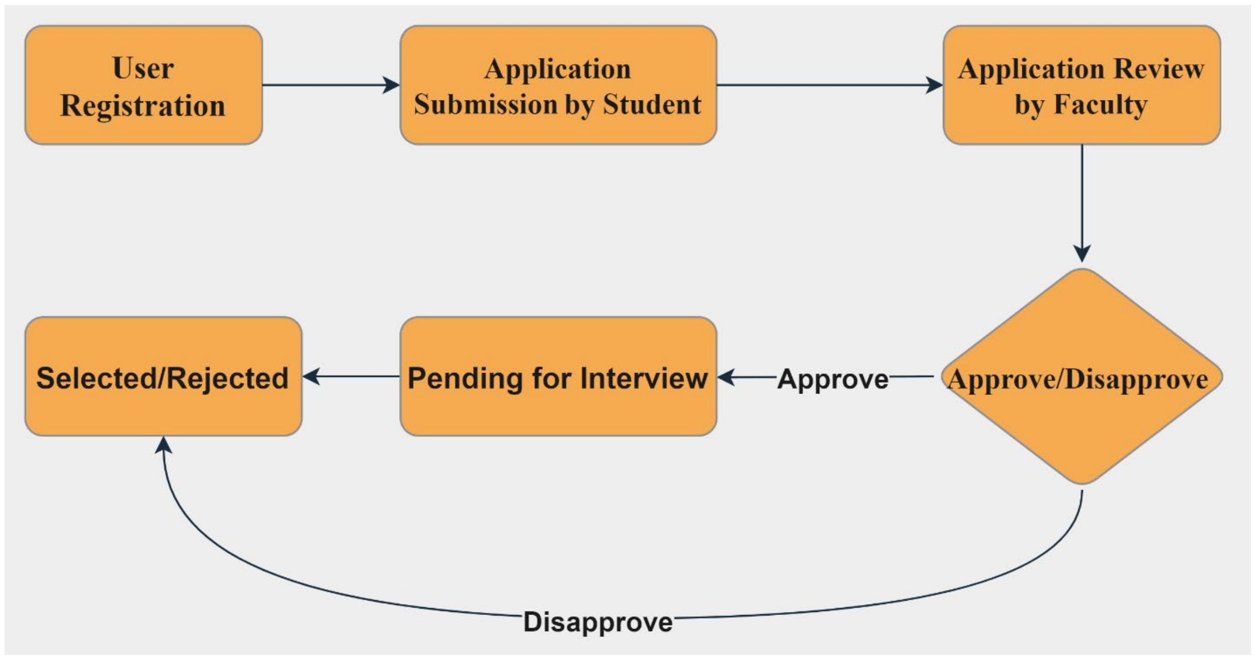
3 Design Details

3.1 Architecture

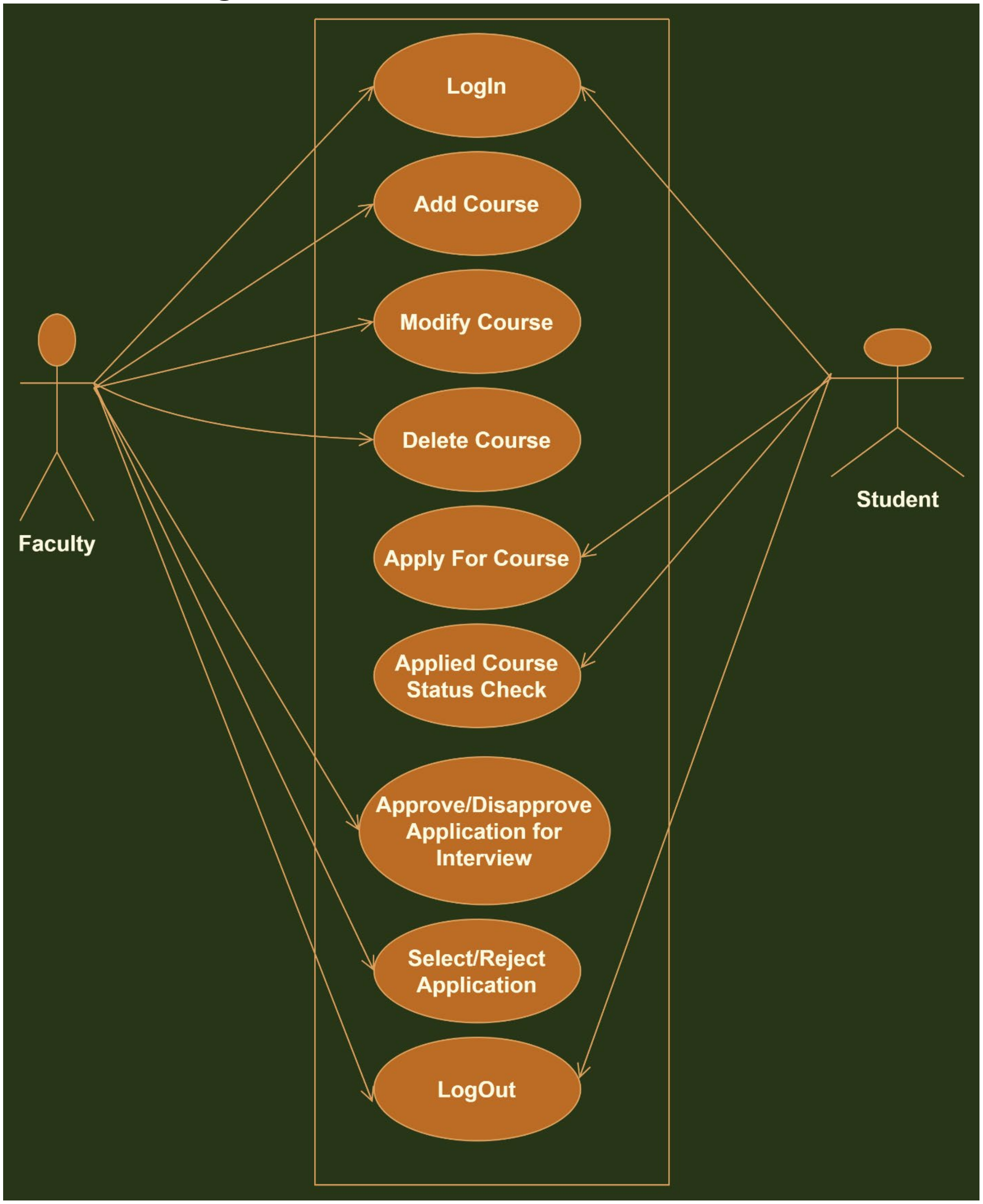


3.2 Process Flow

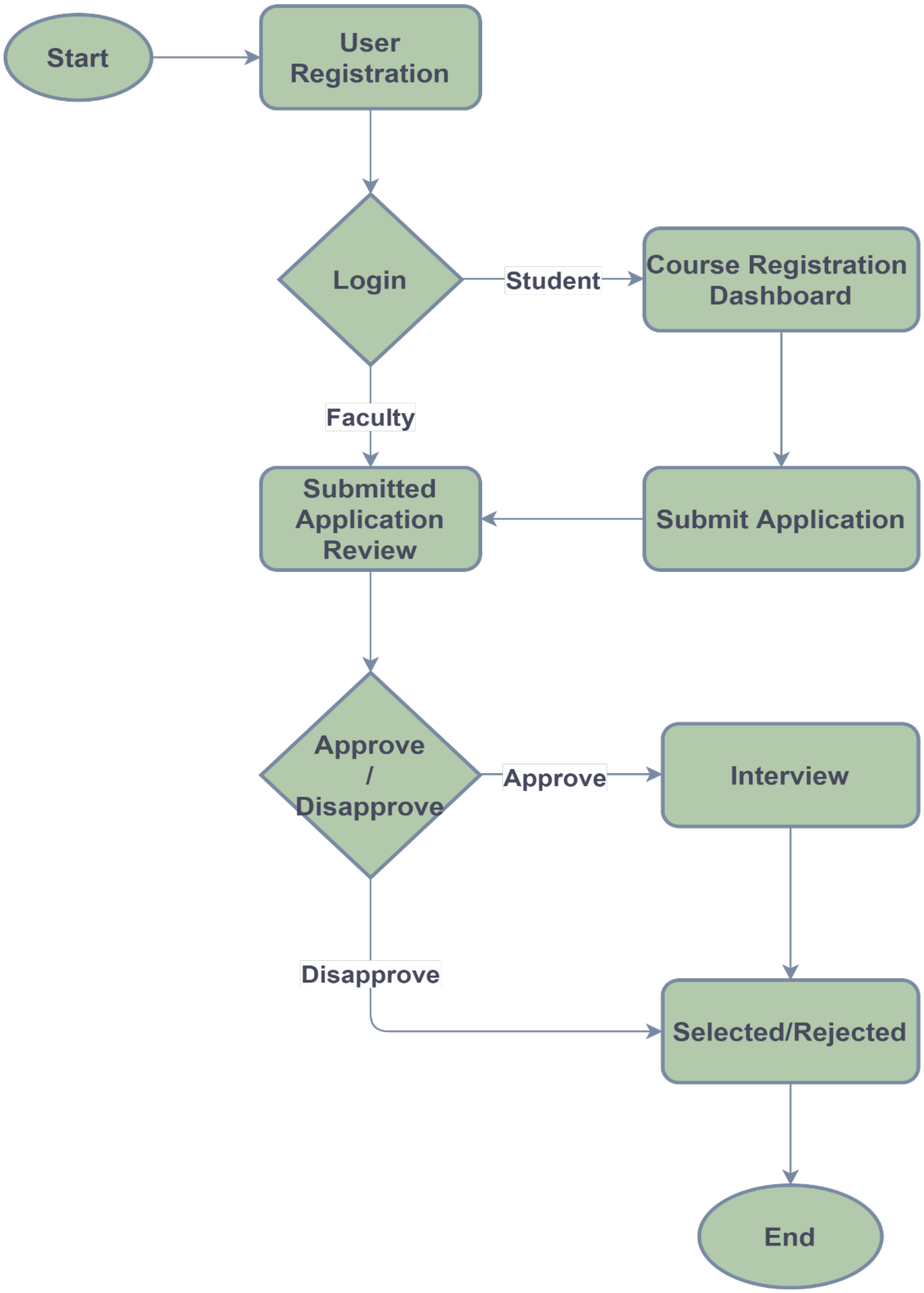
The process flow diagram is as shown below.



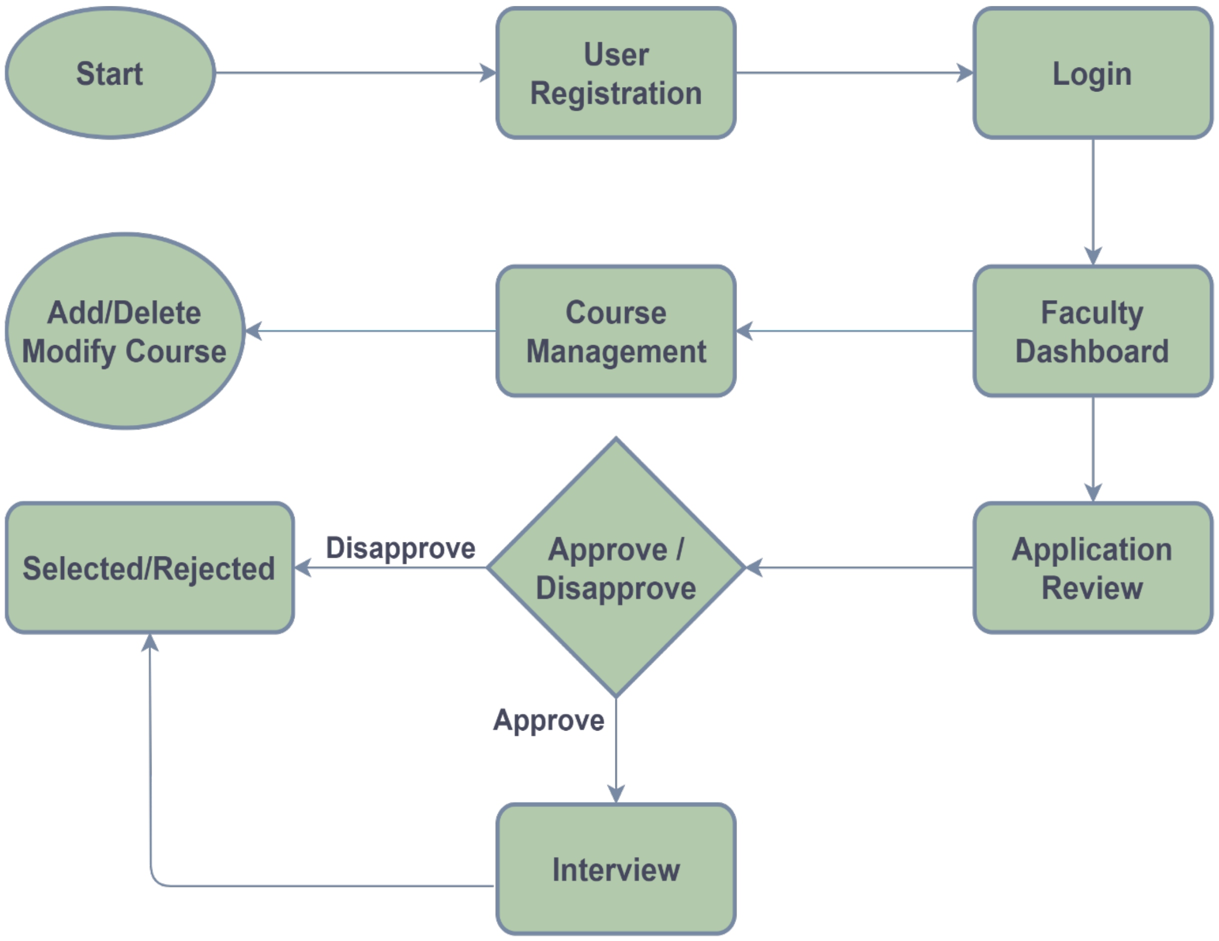
3.3 Use Case Diagram


LogIn
Add Course
Modify Course
Delete Course
Apply For Course
Applied Course Status Check
Approve/Disapprove Application for Interview
Select/Reject Application
LogOut
Faculty
Student

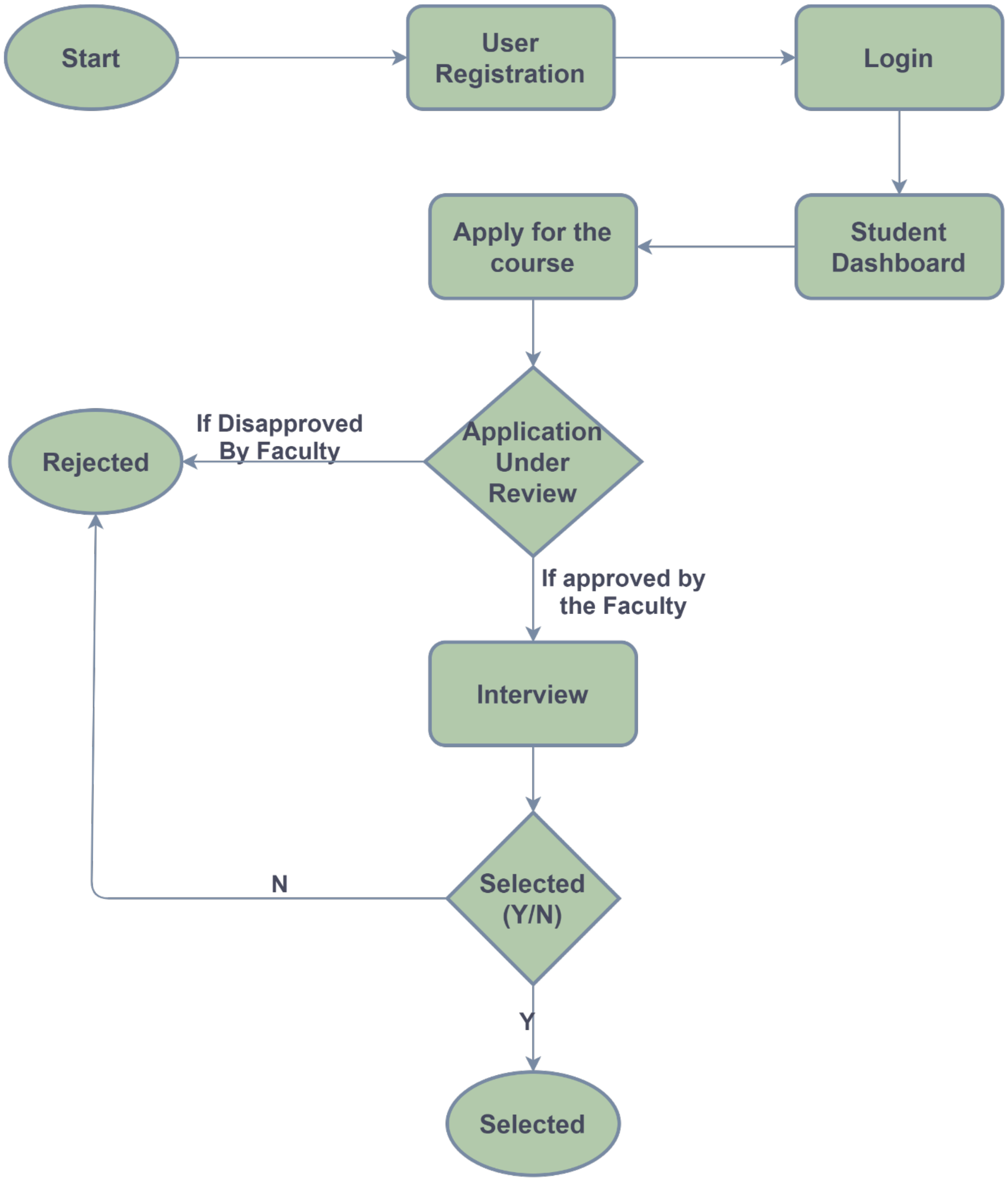
3.4 User IO Workflow



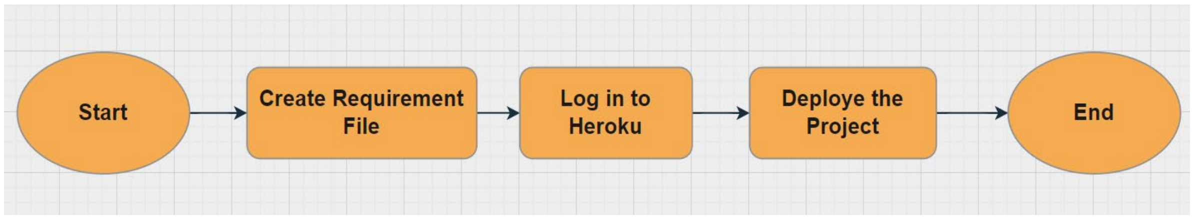
3.5 Faculty I/O Flow



3.6 Student IO Flow



3.7 Deployment Process



3.8 Event log

The system will log all the action that user will perform in the system, which can be used in the event of error. All the logging parameter is stored in the separate configuration file.

- The system will capture when the user logs in to the application and when logs out of the system.
- The system will capture the failed login attempts.
- The system will capture when the application is submitted by the student.
- The system will capture the date and time when the application is approve/disapproved by the faculty.
- The system will capture any exception that will be raised by system.
- The log will be captured in the log file.

3.9 Error Handling

During the process flow if any error is raised by the system user will get the explanation about what is wrong and at the same system will log the detailed error in the log file for debugging.

3.10 Performance

The system will automate the complete admission system in the university. This will reduce the process completion time at the same time student will be aware of the status his/her application is currently in by the continuous email notification for each step change.

3.11 Reusability

The code for the system is written in the modular faction and all the functionality is separated from each other so it can be reused without any problem and multiple team member can work separately.

3.12 Application Compatibility

The components for this project are developed using Python as an interface between them. Each component will have its own task to perform. Since the backend is exposed as an API to the front end so it is the job of the Python to ensure proper transfer of information between back end and front end. At the same time, it can be utilized by any system which can make an API call to the system.

3.13 Resource Utilization

When any task is initiated by the user, it will use all the processing power available until that function is finished.

3.14 Deployment

Microsoft Azure

Google Cloud

amazon web services™

HEROKU

4 Dashboards

System will capture all the information in the log file, which can be utilized by any dashboarding tools like power BI to monitor the health of the system.

KPIs (Key Performance Indicators)

- Log in and Log Out time of the user
- Time taken by the action to complete.
- Frequently used action.

5 Conclusion

The university admission system is designed to solve all the problem faced by student and faculty by utilizing the online system to track and process the applications submitted by the student. The student will no longer required to visit any university for the admission. Student will be able to submit the application for any course from anywhere. Once the application is submitted by the student, faculty member can review the application and take the appropriate action on each application. This will even make the process transparent.

6 References

1. Flask Mail: https://pythonhosted.org/Flask-Mail/
2. Flask Login: https://flask-login.readthedocs.io/
3. Flask: https://flask.palletsprojects.com/