**CHAPTER 1: INTRODUCTION**

**Abstract:**

In this project we aim to aid the university’s project committee in automating their work of assigning each group to a faculty member by designing a website that will take care of calculating the priority of each group and according to a set of rules assign theme to a faculty member that will work with them and inform the university of each group and their respective faculty member.

**Introduction:**

Every year the university’s project committee goes through a long and tedious process of assigning each group to a faculty member the process as is, wastes time and the students may have a misconception on how the process works and is very prone to human error, so we want to automate this process and make it as easy as possible, and ultimately limit unnecessary email usage for both the students and faculty members so they can utilize their time elsewhere and be more productive.

We will also provide the students with guidelines ,FAQ , and the capstone’s handbook, and provide each college with analyzed data to help them make decisions.

**Objectives:**

* avoid human interference
* limit user error
* provide dashboard & graphs for college administrators
* develop a ranking algorithm
* design an easy interface
* calculate faculty members loads to determine if they are available
* make project details and requirements clear
* handbook, guideline, and FAQ are easily accessible

**motivations:**

we find that usually some groups don't get any of the projects they have chosen in their top 10 selections because a faculty member submitted a proposal whilst he's not permitted to do so, another problem that we usually see some individuals might not find a group and both problems can bother administrators as well as unnecessary questions that can be easily answered.

**The literature:**

1- [https://www.rbu-admit.edu.sa/](https://www.rbu-admit.edu.sa/)

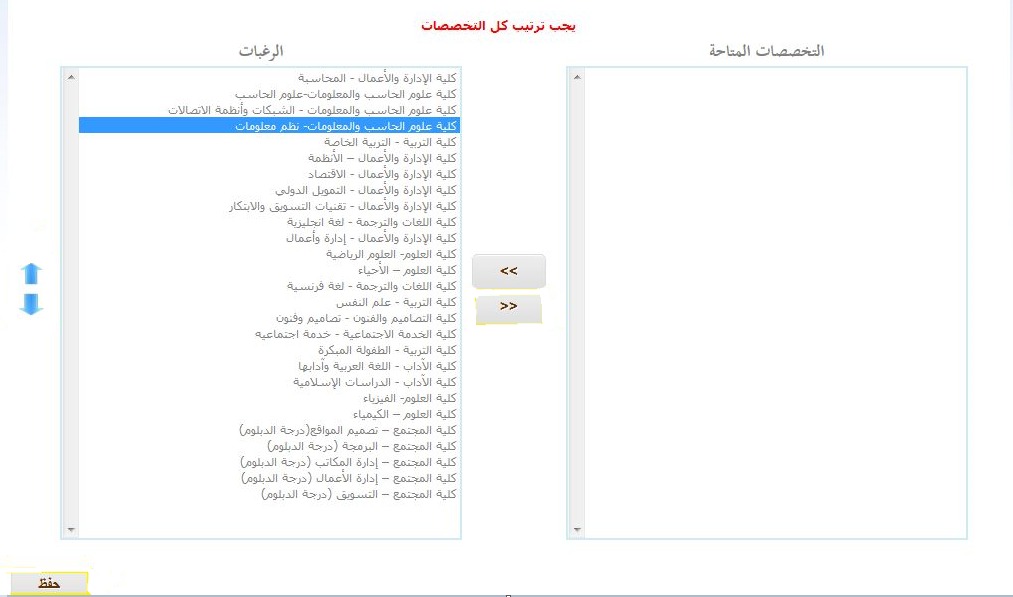
Graphical user interface, text

Description automatically generated

our project is very much similar to this website used to help

high school graduates apply to colleges in their region its very easy and intuitive and we aim to replicate that.

2- universities desirable majors ranking



This is another example very much similar but instead it is used to priorities majors after finishing the first common year, it calculates

your GPA plus some other parameters that will be used in a set formula that is very much different for every university to determine your major, we would like to take inspiration from both examples.

We would hope that our contribution can be used to make a better system in the future.

**POSITIONAL USERS:**

Students: who will use it to rank their desired project proposals and be informed of project details

Faculty members: who will post proposals and wait to be assigned a group.

Administrators: who will receive data to help them make better decisions.

**System development methodology:**

Waterfall:



The classical waterfall model is the basic **software development life cycle** model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used. But it is very important because all the other software development life cycle models are based on the classical waterfall model.   
The classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after the completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus the development process can be considered as a sequential flow in the wate­­rfall. Here the phases do not overlap with each other.

The main difficulties included handling change requests from customers during project development and the high cost and time required to incorporate these changes. To overcome these drawbacks of Waterfall model, in the mid-1990s the Agile Software Development model was proposed.

­­­

Agile:

Diagram

Description automatically generated

The Agile model was primarily designed to help a project to adapt to change requests quickly. So, the main aim of the Agile model is to facilitate quick project completion. To accomplish this task agility is required. Agility is achieved by fitting the process to the project, removing activities that may not be essential for a specific project. Also, anything that is waste of time and effort is avoided.

In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed. The Agile model adopts Iterative development. Each incremental part is developed over an iteration. Each iteration is intended to be small and easily manageable and can be completed within a couple of weeks only. At a time one iteration is planned, developed and deployed to the customers. Long-term plans are not made.

We choose the agile methodology since we feel that the requirements maybe a bit ambiguous and we might add or delete some features as the need arise.

**CHAPTER 2:**

**SYSTEM REQUIREMENTS**

**Functional requirements:**

**-Faculty member:**

|  |  |  |
| --- | --- | --- |
| # | Requirement | Description |
| 1 | Register | Faculty members should be able to register into the website by entering his KSU email, password |
| 2 | login | Faculty members should be able to login into the website by entering his KSU email, password |
| 3 | add proposal | Faculty members should be able to add proposals for the groups to choose from |
| 4 | view group | Faculty member should be able to view the groups assigned to him |
| 5 | contact group | Faculty member should be able to contact the groups assigned to him |
| 6 | delete proposal | Faculty members should be able to delete proposals |
| 7 | Set hours | The system asks faculty members to enter their work load for this semester |

**-Student:**

|  |  |  |
| --- | --- | --- |
| # | Requirement | Description |
| 1 | Register | The student should be able to register into the website n by entering his KSU email, password |
| 2 | login | The student should be able to login into the website n by entering his KSU email, password |
| 3 | rank proposals | Ranking proposal from 10 to 1 (priority list) |
| 4 | view group | Student should be able to view his group |
| 5 | contact faculty member | Group members should be able to contact faculty member assigned to the group |
| 6 | create group | Student should be able to create a group |
| 7 | send Invite | Send invite to other students to join the group |
| 8 | leave group | Student should be able to leave his group |
| 9 | Respond to invite | Student should be able to accept or decline invitation |

**-Admin:**

|  |  |  |
| --- | --- | --- |
| # | Requirement | Description |
| 1 | login | The admin should be able to login into the website n by entering his KSU email, password |
| 2 | set student count | How many students per group |
| 3 | set deadline | Deadline for registration period |
| 4 | Assign groups | Calculate and assign each group to a faculty member |
| 5 | Add student | Student who are permitted to register in the system to avoid students who didn’t sign for the course registration |

|  |  |  |
| --- | --- | --- |
| # | Requirements | Description |
| 1 | Security | * The system is protected by username and password with at least 8   Characters at least one capital char and special char.   * Email authentication will only allow university’s emails to be used with matching id. |
| 2 | Availability | The system will run 24/7 till registration period ends. |
| 3 | Usability | The system will provide a user friendly interface. |
| 4 | Performance | The system will be light and responsive. |
| 5 | Reliability | The system will run consistently with no errors. |
| 6 | Integrity | The system will use military grade encryption. |

**Non-Functional requirements:**

**CHAPTER 3:**

**SYSTEM ANALYSIS**

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |

|  |  |  |
| --- | --- | --- |
| Use Case Name |  | |
| Triggering Event |  | |
| Brief Description |  | |
| Actors |  | |
| Related Use Cases |  | |
| Stakeholders |  | |
| Preconditions |  | |
| Postconditions |  | |
| Flow of Activities | Actor | System |
|  |  |
| Exception Conditions |  | |