****

**Matchmaking System**

**Software Requirement Specifications**

**<Version2.0>**

**Alejandro Chew**

**Maria Alejandra Rodriguez**

**Michelle Rojas**

**Table of Contents**

1. **Introduction**
   1. Purpose 1
   2. Scope of the System Specified 1
   3. Definition, Acronyms, and abbreviations 2
   4. References to Supporting Documents 2
   5. Overview of the SRS 3
2. **General Description 3**
   1. Product Perspective 3
   2. Product Functions 4
   3. User Characteristics 6
   4. General Constraints 6
   5. Assumptions and Dependencies 7
3. **Functional Requirements 8**
4. **Non-Functional Requirements 9**
5. **System Model 11**
6. **System Architecture 22**
7. **Appendices 23**
   1. **Data Dictionary 24**
      1. Actor Descriptions 24
      2. Use Case Descriptions 24
      3. Class Descriptions 24
   2. **Raw Use Case Point Analysis 27**
      1. Actors Summary Table 27
      2. Use cases Summary Tables 27
      3. Scenario Analysis Tables 28
8. **Other Appendices 37**
   1. Sequence Diagrams 38
   2. Database Tables 42
9. **Introduction**
   1. **Purpose**

The purpose of this Software Requirement Specification (SRS) document is to provide a detailed description of the functionalities of the MatchMaking System. The MatchMaking System is a web-based application. This SRS will handle topics of the system that will give a better understanding about the necessary requirements for the success of the project development in accordance with our proposed functional requirements and its non-functional requirements. In detail, it will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document's intended audience includes us (the developing team), FSU faculty, and parties interested in this particular system.

* 1. **Scope of the System Specified**

The idea of creating this web-based software came from the necessity of having a system that helped people to find/make/join a group for a group project. More specific, we found out that this issue was present during Software Engineering I course taught in Fall 2013 at FSU. In addition, classes in general are growing drastically fast making the task of finding or creating a group harder. Harder in the sense that you might not know the personality and skills of all the students in the course due to the size of it and later decide whether a particular student is a good choice or not to work as a team. This project will benefit currently enrolled FSU students that have group projects.

People in general that will interact with the system are the following: Students, Faculty, and the System Administrator.

The system will let students create or join to a group for a specific course currently taught at FSU. Students must register into the system by filling a small form provided at the website and will login using their FSUID email and password.

In addition, the system will show the status of each existing group whether it is full or empty. The information to be displayed of existing groups will contain a list the of the current group members. A student will be able to see each member’s profile, with their basic interest, or idea for the group project.

This system will let system administrator configure the system according to his needs. Maintenance and root control of the system are some tasks that the system administrator will be able to perform.

The goal of the system is to help students in large classroom find and match groups easier for group projects.

* 1. **Definitions, Acronyms, and Abbreviations**

**SRS** Software Requirement Specification.

**Courses** Class Offered at FSU

**Class** Same as Courses

**Term** Exactly what part of the year is a course offered at FSU

**Semester** Another way to say what part of the year the course is offered.

**System Administrator** Person in charge of the system in general, also called admin

**Profile** Brief Description of every student that registers in the system

**FSU** Florida State University.

**Faculty** Professor at Florida State University.

**OS** Operating System.

**Student** Currently Enrolled student at Florida State University that will use the system

**Group** A number of persons that will work together for a common purpose/goal.

**Web Browser** Any application/program that gives you access to the internet. (i.e. Internet explorer, Firefox Safari)

**Functional Requirements** Basic operations that the system should be able to do for the software development be as efficient and fast as possible.

**Non-Functional Requirements** Constraints imposed by the client to the software that are not important during the development phase of the software.

* 1. **References to Supporting Documents**

**Technical Reference:**

Bruegge, Bernd, and Allen H. Dutoit. Object-Oriented Software Engineering using UML, Patters and Java, Boston: Prentice Hall, 2009. Print.

Gehrke, Johannes. Database Management Systems. NewYork: Mc GrawHill, 2002.

* 1. **Overview of rest of SRS**

The following content of this SRS will describe several aspects of the MatchMaking System. The Overall Description section of this document gives an overview of the functionality of the product. In addition, it also describes the informal requirements of the project. This description is organized in several parts.

The first part gives a **General Description** of the System including:

* Product Perspective
* Product Functions
* User Characteristics
* General Constraints
* Assumption and Dependencies

The second part of the SRS document will provide information about the **functional requirements** of the software. This SRS will provide as accurate as possible the functional requirements to accomplish the basic goals of the customer.

The third part of the SRS document will provide information about the **non-functional requirements** of the software. As mentioned before, this SRS will provide as accurate as possible the non functional requirements and constraints present.

As the fourth part, we describe in detail the system model by giving a brief description of each use case diagrams and the basic procedure by using that specific use case diagram.

In the fifth part, we view the system architecture of the entire system by displaying the class diagram of the software.

At last, appendices to the SRS are located. These appendices will help the reader of the SRS to clarify any confusion he had throughout the entire document.

1. **General Description**
   1. **Product Perspective**

**Client Side:**

The system should run in a way that the user logs in into the system. Once logged in with his/her FSUID he may join or create any kind of group for a particular course. All these will be done throughout a web browser such as Mozilla Firefox, Internet Explorer, Google Chrome, or Safari. In addition, through this same interface, the user will be able to perform any action stated on the functional requirements.

**Server Side:**

This system is independent and totally self contained. It will run in a server that has a relational database (MYSQL), and a web server (apache). and The server service (Hardware and web server hosting will be provided by **www.godaddy.com**.

* 1. **Product Functions**

The main functions that can be found in this software are the following:

* + 1. **Register**
       1. Students must be able to register to the system by filling a form page and submitting it to the system.
       2. In order to register, student must be a valid FSU student that will be able to login (authenticate) with his blackboard account.
    2. **Login Into the System**
       1. The function to login into the web based application
       2. The user will login to the system using his FSUID email and password. (Blackboard account)
    3. **Create Profile**
       1. In charge of letting students create a profile to be displayed to the public. He can decide for which class it will appear.
       2. This function of the system will ask for information about a student personal experience skills and any other idea that he is willing to share to others to help find him find a group.
    4. **Modify Profile**
       1. Let the student modify any saved profile
       2. A student can delete or change the details of each of his personal profiles found in his account.
    5. **Create Group**
       1. Students must be able to create a group with certain specifications like course, max size, and a description of the goal of the group.
       2. Students must select a class to where they want the group to be created.
    6. **Find and Join Group**
       1. Students must be able to join any group that is not full without problems if administrator of the groups accepts him.
       2. The member that created the group must give authorization for the user requesting to join the group can do so.
       3. Students must select a class to where they want to search for a group and join.
       4. You must select a course to minimize the search and look for a group to join.
    7. **Request To Join Group**
       1. This function sends a request to the owner of the group in behalf of student that wants to join.
       2. This request will be sent through the FSU email.
    8. **View Groups**
       1. Students must be able to see all the returned from the find function.
       2. Students will be able to see the status and brief description of the group.
    9. **View Your Groups**
       1. Students must be able to see all the groups that he currently forms part.
       2. Students will also be able to see the status and brief description of the group.
       3. Student will be able to delete (get out) of any current group he forms part.
    10. **Delete group**
        1. A group owner can delete a group he created.
    11. **Add member to a group**
        1. Group owners will receive the request whether to let a person be part of the group or not.
        2. This function will execute the desire of the group owner either to accept or reject the request.
        3. Owner of the group can also request for someone to join the group.
    12. **Leave Group**
        1. Students will be able to leave the group anytime they want to do so.
        2. This function officially removes them from any group they are part of.
    13. **Delete Member of Group**
        1. Group Owner will be able to delete a member of the group any time he wants to.
    14. **Logout**
        1. Let students log out of the system in a safety fashion.
  1. **User Characteristics**

Eventual users of the system will be **Students, possibly few Faculty and System Administrators**

With these types of users, one can assume that they have at least a high school diploma as their highest degree. Therefore, constraints such as not knowing how to use a computer, read or write will not be an issue to take into consideration while developing the system.

The primary users of this web based application are the currently enrolled students of FSU. Students will be highly motivated to use this system because it will help them pick their groups.

* 1. **General Constraints**

**Client Side to consider:**

One of the general constraints for the system is the language for the user interface. The system should be programmed in English. The user must only interact with the system through a modern web browser.

The robustness of the user interface should be easy for users with low computer experience to understand.

The computer of the student must be able to support modern web browsers with the latest updates.

It is also assumed that the client has a good and stable internet connection.

**Server Side to consider:**

The system should run on any web server like apache. The system should be programmed in a fashion that will provide the system an easy maintenance and future upgrades features.

The machine (hardware) that will host the system must be capable of giving service for a small amount of computers (like 10) for testing purposes, but on the big scale, it should be able to give service to all the students currently enrolled at FSU.

**Additional Constraints:**

The system must support common input interfaces such as keyboard, mouse if available and touch screen.

The system should also be backward compatible with web browsers running in different platforms such as Windows XP OS and newer ones.

A good internet connection is required for the well functioning and performance of the system.

Another constraint is that any person who wants to register to the system must have a valid FSU email address. In other words, it must be a current student at FSU.

* 1. **Assumption, and Dependences**

The assumptions and dependencies to create this software are the following:

* + 1. Students that register and logs in into the system will have a computer and have access to internet.
    2. Students have a valid FSU email that is currently working.
    3. Students will have a basic understanding of how to use a computer and access URLs over the internet.
    4. Students who are logged in the system will be able to create, join, view, or exit a group.
    5. Students will not have to enter any personal information in the system (i.e. Last Name, First Name, Email address, classes) to login, they will only need their valid FSU email and the password set at the moment of registering for login.
    6. All group matchmaking or interaction between client and server will be done through a modern web browser and modern servers.
    7. Students know how to read and have a basic understanding of the language English.
    8. The system assumes that the computers will be able to run modern web browsers.
    9. The computer where the software is being used has a good performance and is working perfectly and has access to internet.
    10. There is no power outage in the computer running the system with the MatchMaking system and at the computer where the student is trying to create, join, view or exit a group.
    11. This system gives no help to disabled students such as hearing or touch aid.
    12. This system depends on a good internet connection to interact between the user and the system server.

1. **Functional Requirements**

**The functional requirements of the FSU Testing Reservation System are the following:**

1. Students must be able to login in the system using their FSUID email and password to login.
2. While the students are registering, they must fill a profile form that shows a brief description of them (courses being taken, gender …).
3. Students must be able to create a work group.
4. The student that creates a group must be able to assign the maximum number of team members and the desired project idea or goal.
5. A student may be part of just one group for each class.
6. Students can join a group with the permission of the group owner.
7. Group owners can decide whether to accept a person to his or her group or not.
8. Group owner must be able to add a person to the group after the request.
9. Students must be able to request people to join a group through email.
10. Students must be able to select a group and try to join if it is not full.
11. Students must be able to join a group that is not full.
12. Students must be able to exit a group whenever they want.
13. Group owners must be able to delete a group they created.
14. A confirmation email must be sent to all the group members if a group has been deleted.
15. Students must be able to logout from the system.
16. Students must be able to login.
17. System must be able to properly authenticate a user during login.
18. Students must be able to set the semester in which the groups they want to create will belong to.
19. Students must be able to select the class where they want the group to be created.
20. Students must be able to set the year of the semester as well as the term in which the group they want to create it belongs to.
21. Students must be able to select the course in which they want the group they want to create belong to.
22. Students must be able to select the semester in which they want to search for a group, then decide which course and finally view the groups available.
23. When deciding which course to select to search or create a group they must be able to see the details of the course such as the instructor teaching that course, course name and number.
24. Any student registered and logged in the system must be able to see the profile of the group members or from other people registered in the system for a particular class.
25. The system should be designed and coded in an object oriented way for future expansion of features.
26. Students can view the status of their current group at any particular time as long as they are logged in the system.
27. Students must be able to change their profile details.
28. Students must be able to delete their profile but mandatory have at least one.
29. Group owners must be able to modify the properties of the group they created.
30. **Non-functional Requirements**
31. Timeout due to inactivity. This is not required since our main goal is to let users create a reservation.
32. After a semester, the system resets itself and clears all data.
33. System must ask confirmation to kick a person off the group.
34. System must ask for confirmation in exiting voluntarily from a group.
35. System must ask confirmation to send a request to join a group.
36. The design of the website. This is not required as long as a reservation is possible to make by any user.
37. Colors applied on the website. This just contributes on how well the system looks; however, it does not contribute in any aspect of the main system requirements and goals.
38. Efficiency and Performance is not that important in the development phase since sooner or later a person will get a reservation for his test if the functional requirements of the system has been implemented.
39. Security is not a functional requirement because it does not contribute in delivering a match making system. Instead makes it harder to develop the software because security is a huge and delicate topic.
40. Maintainability is also not a functional requirement doe to the fact that the system compared to others is not as big as to require special requirements to keep the software easy to maintain.
41. The Robustness of the System should be simple for simplicity right now we don’t want to complicate things more than what they already are.
42. Logging out isn’t too important because usually people just close the browser and they are not worried out because the system itself detects that the connection has been lost and closes.
43. Compatibility is a non-functional requirement since this matchmaking system is intended to be used only at The Florida State University.
44. Documentation is not really a concern to us; (just the SRS) because it does not contribute to make the software running and prevent users start matching groups. Documentation is necessary after the formal release of the software for future updates and modifications another developer would like to implement.
45. There is no need of reliability at the beginning of software developing because it is not put into a real life situation where matchmaking a group is extremely crucial. Students are not in dangerous situation threatening with their performance and also because our main priority is to make all user be able to do the task said in the functional requirements of this document.
46. **System Model**

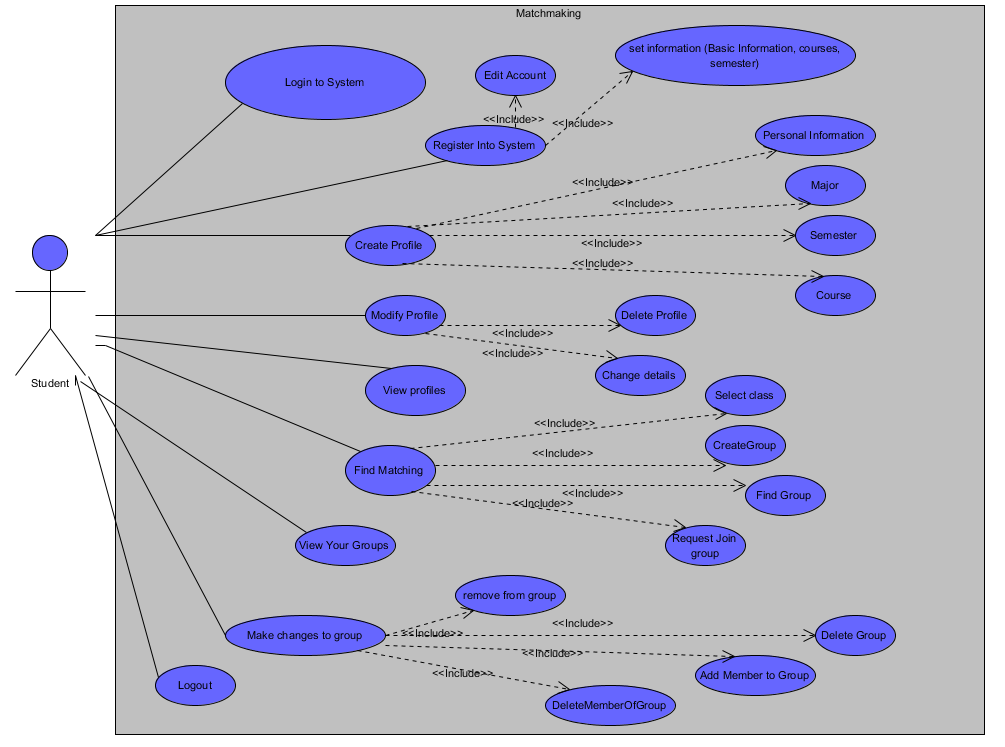
**Overall Description**

**System Environment**

**Functional Requirements Specifications**

This section outlines the use cases for each of the current valid actor of the system. The student is the main actor in the system.

1. **The Following is the general use case of a student and how the system interacts with it.**



**Brief Description:**

The user that is a student can find or create a group by first logging and registering in the system.

**Initial Step-By-Step description for the overall of the system:**

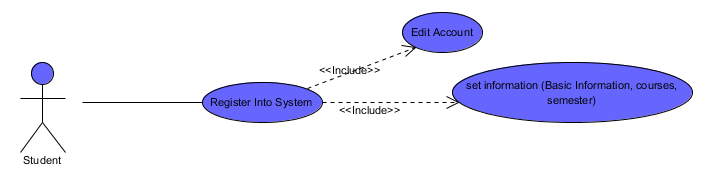
1. The student must login into the system in order to register. Students must fill a form (set information use case) and submit it through the website.
2. Once registered into the system, a student can login and change several account settings and information of his account.
3. Once having set his account to his desire, he must create a profile (create profile use case) to advance any further in the system features. A student may have several profiles and decide which one to display for different courses/classes
4. A user can modify a profile (modify profile use case) as long as one exists. If no profile exists, this option will be disabled.
5. A user can see his profile to know exactly how it will be displayed to the public.
6. Once all the previous steps are done, a user can start searching for a group (find matching use case).
7. A user can see in which groups is he enrolled or has created. Having no groups, the system will display a message that he does not belong to any group when he tries to see his groups.
8. Owners/creator of a group will be able to modify the group (make changes use case) properties as well as managing group members.
9. Iteratively the user can do all of the following procedure as many times he wants, finally after he is done, he can logout anytime he wants (leaving the website will work too).

**Note: The Main use cases (mentioned below), their name explain themselves about actually what the student can do. However, if there is any confusion about the functionality of these main use cases. You can find a brief description of the use cases on the Raw Use Case analysis under the Use case Summary Table.**

The main use cases are the following:

* Login to System
* Registering in System
* Create Profile
* Modify Profile
* View Profile
* Find/Search Matching
* View Your groups
* Make changes to Group
* Logout
* Configure System
* Maintenance and Updates
* Access Systems
* Assume Student Role

1. **The following is a more precise description for each general use case on the use case diagram above.**
2. Register Into System Use Cases

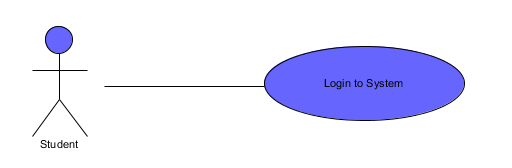


**Brief Description**

The student will register into the system by giving out the information requested by the system.

**Initial Step-By-Step description:**

1. Student will go to the URL of the system.
2. Student will login with his FSUID and password
3. The system will detect if he has registered into this system, else it will login automatically into the system.
4. He will then fill the information requested.
5. He can modify or edit information on his account.
6. Login to System Use Cases

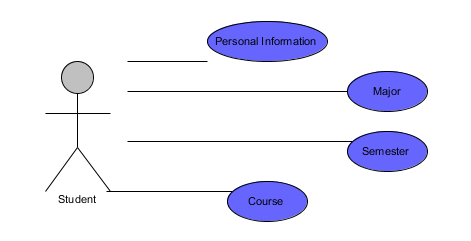


**Brief Description**

User will log in to the system using his FSU email and password.

**Initial Step-By-Step description:**

1. The username will be FSU email address and the password will be blackboard (email) password.
2. Create Profile Use Cases

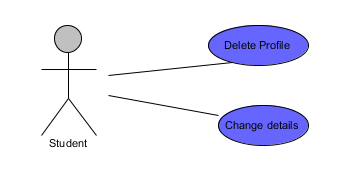


**Brief Description**

In this use case, the user will create a profile that will be visible to the parties of interest he decides.

**Initial Step-By-Step description:**

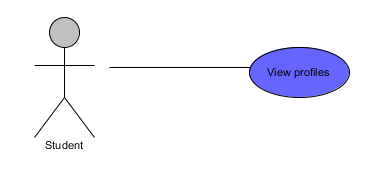
1. User will fill basic personal information his desire, a few to mention will be his name, DOB, skills, and interests.
2. The major use case lets the student decide and select his major.
3. A Student must pick a semester in which he wants the profile to be displayed as well as the course in which he wants the profile to be displayed.
4. A profile will be done once he click the save button.
5. Modify Profile Use Cases



**Brief Description**

The student will be able to modify the saved profiles he has saved in his account.

**Initial Step-By-Step description:**

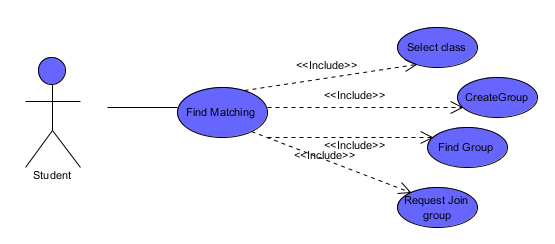
1. A user will be able to delete a profile by clicking a delete profile button near the profile he wants to delete.
2. The system will ask to confirm to delete a profile.
3. Students will be able to modify their profile anytime they want.
4. A profile will be done once he click the save button.
5. View Profile Use Case

**Brief Description**

The student will be able to see how the profiles he has saved in his account will be seen by others

**Initial Step-By-Step description:**

1. Students will pick a profile to display for preview.
2. Find Matching Use Cases

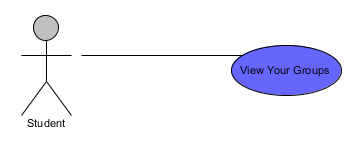


**Brief Description**

This is the core of the system. These system features will help students find a group for a group project. Students will be able to find groups of similar interests for a course.

**Initial Step-By-Step description:**

1. Students will select a class for which they need a group
2. Once they are in the class where they want to find a group and clicked on the search button, students can see group that currently exists.
3. If he decides that none of the current groups complies with the student criteria, the student has the option to create his own group.
4. When a student is searching for a group, or looking team members for the group he created, he can see the profile of each student that currently has no group.
5. Students may also see the profile of the team members of formed groups.
6. Finally, a student may create or receive a request. Either he sends a request to join a group he would like to be part of, or he receives a request from a group to be member of.
7. The system will be in charge to send you emails about these requests.
8. This runs iteratively until the user decides to stop.
9. User may only belong to a single group for each class.
10. View Group Use Case

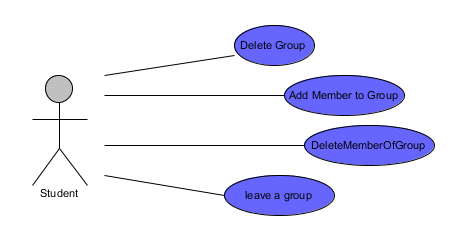


**Brief Description**

This feature lets you see all the groups in which you are currently part of. You may be in different groups because students usually take several courses in which group work is required.

**Initial Step-By-Step description:**

1. Just click a button that will say “View Groups” and a list will appear with all the groups in which you belong to.
2. Make Changes to a group Use Cases

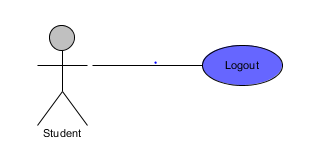


**Brief Description**

Students will be able to make any changes to the groups they currently belong to. Students that are owners of the group have additional control over the groups they own.

**Initial Step-By-Step description:**

1. Student will choose a group he belongs to and make any changes.
2. If the student is the owner, he can delete the group.
3. In addition, group owners will be able to reject a request to join the group, remove a member of the group, and make several changes to the properties of the group.
4. If a student is a member of the group and not the owner, he has the power to leave the group anytime he wants.
5. Logout Use Cases

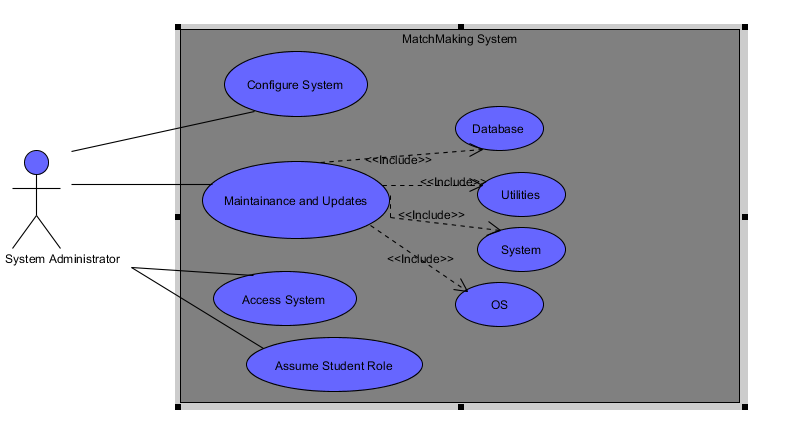


**Brief Description**

Students will logout of the system

**Initial Step-By-Step description:**

1. Students can logout safely from the system by clicking the logout button.
2. Any unsaved changes will be lost.
3. Students can logout too by simply closing the browser, although this is not recommended.

**The Following is the Use Case is for the System Administrator**

**Brief Description:**

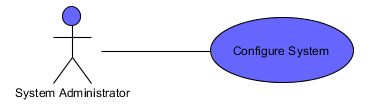
The System administrator is the one in charge of giving maintenance and troubleshooting any issues present while the system runs.

**Initial Step-By-Step description for the overall of the system:**

1. Automatically, the system administrator has root control of the system and can do any of the following in the next steps.
2. He can give maintenance of software, database, network, and configuration of the system.
3. Administrator can complete configure the system.
4. Finally when he is done, he may simply exit the system
5. He is not logged in the system; he is actually logged on the components that make possible the system to run.

**The following is a more precise description for each general use case on the use case diagram above.**

1. Configure System Use Case

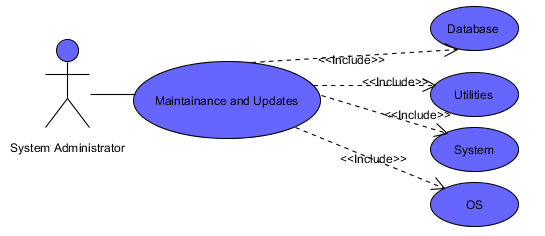


**Brief Description:**

The System administrator can configure the system properties anytime he wants to.

**Initial Step-By-Step description for the overall of the system:**

1. Admin will click in his menu the Configure system option for each of the system components that run the Matchmaking system and modify any aspect of it.
2. Maintenance and Updates Use Case

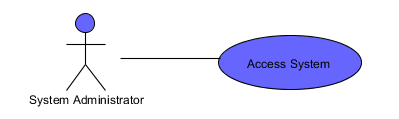


**Brief Description:**

The System administrator has root control over the components that host the system and has a lot of privileges. one of these privileges is to give maintenance to the system and keep it up to date.

**Initial Step-By-Step description for the overall of the system:**

1. Admin will click on the maintenance and updates button to enter that part of that component.
2. He will then choose what part of the system he would like to update.
3. The main components that keeps this system up and running that the System Administrator will give maintenance is Apache, MYSQL, and web pages.
4. Access System Use Case

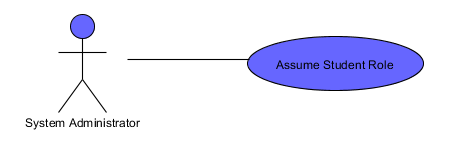


**Brief Description:**

Admin has access of all the properties and configurations of the system

**Initial Step-By-Step description for the overall of the system:**

1. Admin has access of the components that keeps up and running the Matchmaking System.
2. Assume Student Role Use Case



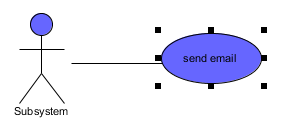
**Brief Description:**

Admin can assume the role of a student, therefore he can do anything as well that a student can do.

**Initial Step-By-Step description for the overall of the system:**

1. Admin simply logs in into the system and will be able to act as a student.
2. Immediately has same privileges that a student has.

**The Following is the Use Case is for the Subsystem**



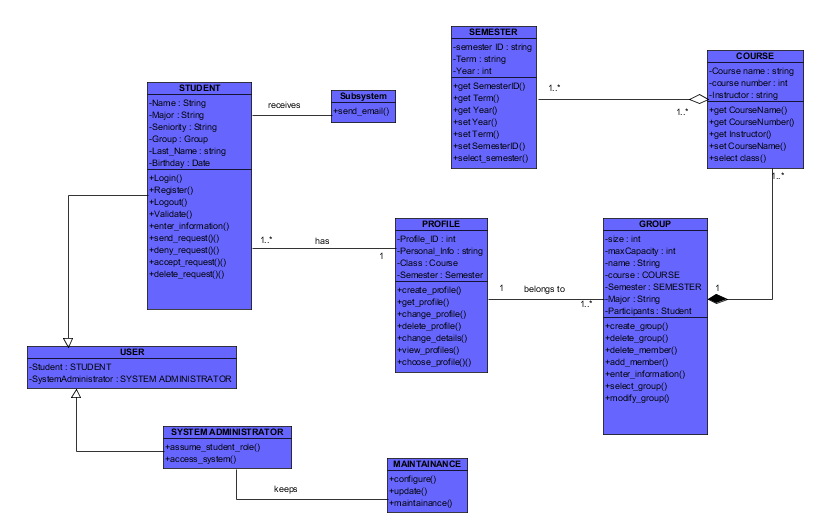
**Brief Description:**

Subsystem is in charge of sending notification, confirmation or any other message that is necessary to the appropriate party that is registered in the system.

**Initial Step-By-Step description for the overall of the system:**

1. Subsystem automatically sends emails when it is necessary to the students or appropriate party.
2. **System Model**

Below, a high level overview of the anticipated system architecture is prese3nted. It shows the fundamental classes that must be modeled with the system to satisfy with its requirements. This system architecture is presented with a class diagram in which it displays all the attributes, and major functions of each class. In addition, it also displays the relationship between classes and their multiplicity. The diagram was created with the following software: Visual Paradigm for UML.



1. **Appendices**
2. **Data Dictionary**
3. **Actor Description**
4. **Student:** The student is the main actor on the system. A student must be currently enrolled at FSU.
5. **System Administrator:** Person in charge of giving the appropriate maintenance and updates to the system. He is in charge of keeping the service provided by the system smoothly.
6. **Use Case Description for Student**

|  |  |  |
| --- | --- | --- |
| **#** | **Use Case** | **Description of the Use Case (Student)** |
| 1 | Register Into System | Let a student enrolled at FSU register into the system (Matchmaking System). |
| 2 | Set Information | Let the Register into System gather information from student. |
| 3 | Login to System | Login to system using FSU email and password |
| 4 | Create Profile | Let student create a personal profile that other people will see. |
| 5 | Personal Information | Student will add his personal information in this use case. |
| 6 | Major | Select the major in the profile. |
| 7 | Semester | Select semester where the profile will appear. |
| 8 | Course | Select the course in which people in that course can see your profile. |
| 9 | Modify Profile | Make Changes to the profile that has been saved in the account |
| 10 | Delete Profile | Function that will delete a saved profile. |
| 11 | Change details | Students will be able to change the details on a saved profile. |
| 12 | Find Matching | Students will find and create groups |
| 13 | Select Class | Student will select class where to look or make a group |
| 14 | Create Group | Student Creates a group |
| 15 | Find Group | Student can see and find a group of their desire |
| 16 | Request Join Group | Student can send a request to join in a group |
| 17 | Find profiles without group | Student can see the profile of all the students that are registered in the system on a particular class/course |
| 18 | View Your Groups | Student can see a list of groups he belongs to. |
| 19 | Make Changes to group | Group owner or member can do certain things on a group they belong to. |
| 20 | Leave a group | Student may leave a group |
| 21 | Delete Group | Group Owner can delete a group he created. |
| 22 | Add Member to Group | Group Owners take care of request to add a person into the group, or ask someone to join the group |
| 23 | Delete member of group | Group owner can delete a member of a group. |
| 24 | Logout | Exit the system safely from your browser. |
| 25 | View Profile | See other students profile. |

1. **Use Case Description for System Administrator**

|  |  |  |
| --- | --- | --- |
| **#** | **Use Case** | **Description of the Use Case (Student)** |
| 1 | Configure System | Admin can configure the system components properties |
| 2 | Maintenance and Updates | Admin can do Maintenance and Updates, this has specializations. |
| 3 | Database | Admin gives maintenance and updates to the database component of the system |
| 4 | Utilities | Admin gives maintenance and updates to the Utilities component of the system |
| 5 | System | Admin gives maintenance and updates to the System component of the system |
| 6 | OS | Admin gives maintenance and updates to the OS component of the system |
| 7 | Access System | Admin has control of all the system( Also known as root Control) |
| 8 | Assume Student Role | Admin must be able to act as a student, has all the privileges and function that a student can have. |

1. **Use case Description for Subsystem**

|  |  |  |
| --- | --- | --- |
| **#** | **Use Case** | **Description of the Use Case (Student)** |
| 1 | Send Email | Subsystem must automatically send email of confirmation and notifications when necessary. |

1. **Class Description**

**Student Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Name | Name of Student | String |
| Last Name | Last Name of Student | String |
| Major | Major of Student | String |
| Seniority | Year in College | String |
| Birthday | Birthday | Date |
| Group | Groups he belongs to | Group |
| Courses | Courses enrolled | Courses |

1. **Profile Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Profile\_ID | Number of profile | Int |
| Personal Info | Description of student | String |
| Email | Email of Student | String |
| Picture | Picture in Profile | Jpeg |

1. **Group Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Size | Size of group | int |
| Max Capacity | Total capacity of the group | Int |
| Name | Name of group | String |
| Course | The course where the group belongs | course |
| Section Number | Section of the course | int |
| Participants | Members of the group | Student |
| Group Description | Description of the group in general | String |

1. **Course Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Course Name | Name of course | String |
| Course Number | Umber of course | Int |
| Instructor | Instructor of course | String |

1. **Semester Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Semester id | Id of semester | String |
| Term | The term of the semester (fall, spring or summer) | String |
| Year | Year the semester belongs too | Int |

**Subsystem Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| None | None | None |

**User Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| Student | Act as role of a student | Student |
| User Administrator | Act as role of a System Administrator | System Administrator |

**Maintenance Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| None | None | None |

**System Administrator Data Dictionary Class**

|  |  |  |
| --- | --- | --- |
| **Name of Attribute** | **Description** | **Type** |
| None | None | None |

1. **Raw Use Case Point Analysis**
2. **Actor Summary Table**

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Student | Currently Enrolled student at Florida State University that use almost all the features of the system. This is the main actor of the system. He will be able to create, join, and delete a group. |
| System Administrator | Person in charge of keeping the system running smoothly. The system administrator will also troubleshoot all the issues given by the system. He also sets the basic properties and configurations of the system. |

1. **Use Cases Summary Tables**

|  |  |  |
| --- | --- | --- |
| # | Use Case Name | Description |
| 1 | Register into system | The student will register into the system by giving out the information requested by the system. |
| 2 | Login to System | User(Student) will login to the system with his respective FSU email and password. |
| 3 | Create Profile | Once a student is registered and logged into the system, he can create one or more profile with his personal description and basic information to be visible to others. |
| 4 | Modify profile | Student can modify a saved profile under his account. He may also delete a profile he has previously saved. |
| 5 | View Profile | Display a list with all the profiles done by the student that are saved on his account. |
| 6 | Find Matching | Student will find, create and join groups. This is the main feature of the system. They will also see profiles of other persons that will help them decide whether a group will be good or not. |
| 7 | View your groups | Display a list with all the groups that the student belongs to. |
| 8 | Make Changes to group | Group owners can send request for someone to join the group. They can accept request from person who wants to join the group. Group owners can delete a group member, and change basic properties of the group. |
| 9 | Logout | Student can exit safely from the system |
| 10 | Configure System | System administrator can change and modify basic properties of the system components. |
| 11 | Maintenance and Updates | Admin has the right and privileges to keep up to date all the components of the system, and give them maintenance to keep them running smoothly. |
| 12 | Access System | System Admin has the root control and access all parts that conforms the system. He can do whatever he wants with the system. |
| 13 | Assume Student Role | Administrators can assume the role of a student, therefore they have all the functionalities and privileges that a student have. |

1. **Scenario Analysis Tables**

|  |  |
| --- | --- |
| **Use Case Name** | **Register Into System** |
| **Participating Actors** | Student. |
| **Trigger** | The Student access the website |
| **Entry Conditions** | Student visits the website for the first time and hasn’t registered yet into the system and tries to login with his fsu email and password. |
| **Flow of Events** | 1. User enters very basic information such as full name and BOD. 2. User clicks submit buttons and registers into the system. |
| **Alternative Paths** | 1. The student is already registered and decides to login into system |
| **Exit Conditions** | Student is registered into the system and can now login successfully to it. |

|  |  |
| --- | --- |
| **Use Case Name** | **View Account** |
| **Participating Actors** | Student. |
| **Trigger** | The Student access the website and clicks on “Account” then “Edit Account” |
| **Entry Conditions** | Student is logged in the system. |
| **Flow of Events** | 1. Students modify any particular information of his account. 2. Student clicks submits. 3. Information is updated. |
| **Alternative Paths** | 1. The student decides to not modify anything and clicks cancel button. |
| **Exit Conditions** | Student either clicks submit or cancel on the web page and exit to his home menu. |

|  |  |
| --- | --- |
| **Use Case Name** | **Login to System** |
| **Participating Actors** | Student. |
| **Trigger** | The Student access the website and wants to get into the system |
| **Entry Conditions** | Student is already registered in the system. |
| **Flow of Events** | 1. Student inputs FSU email and password. 2. If valid, the student logs into the system, else the system will ask for his username and password again. |
| **Alternative Paths** | 1. Student doesn’t remember his FSU email or password must go to blackboard and request to recover for a lost password. |
| **Exit Conditions** | Student is logged in the system. |

|  |  |
| --- | --- |
| **Use Case Name** | **Create Profile** |
| **Participating Actors** | Student. |
| **Trigger** | The Student goes under “Profile” and then on “Create Profile” |
| **Entry Conditions** | Student is logged in the system in his account and home menu. |
| **Flow of Events** | 1. Student gives name to the profile. 2. Student selects if he wants his email to be displayed. 3. Student selects the course he wants the profile to be displayed. 4. Students fill all the descriptive information he wants for other users can know a little bit of him. 5. Student will save the profile. |
| **Alternative Paths** | 1. Clear all information and start filling the form again. 2. Cancel to create a new profile. |
| **Exit Conditions** | 1. Student has successfully created a profile, and is saved in his account. 2. Decides to not create a profile and goes back to his home menu button. |

|  |  |
| --- | --- |
| **Use Case Name** | **Modify Profile** |
| **Participating Actors** | Student. |
| **Trigger** | The Student goes under “Profile” and then “Edit/View Profiles”/ |
| **Entry Conditions** | Student has existing profiles under his account and is logged in the system. |
| **Flow of Events** | 1. Student clicks on the existing profile after clicking on “Edit/View profiles”. 2. Student will click on modify profile on the profile he wants to modify. 3. Student will change his existing profile. 4. He saves the profile after changes has been made by clicking submit button. |
| **Alternative Paths** | 1. Decides not to modify profile, and clicks cancel. |
| **Exit Conditions** | Student has successfully modified and made changes to his existing profile, or exit without doing any changes. |

|  |  |
| --- | --- |
| **Use Case Name** | **View Profiles** |
| **Participating Actors** | Student. |
| **Trigger** | The Student will go under “Profiles” and click “View Profile as public” |
| **Entry Conditions** | Student has existing profiles under his account and is logged in the system. |
| **Flow of Events** | 1. System will display all the profiles he has done exactly the same way how other users will see it in public. |
| **Alternative Paths** | 1. .None |
| **Exit Conditions** | Student has successfully seen a list of all the profiles he has created and the way other people will see it. |

|  |  |
| --- | --- |
| **Use Case Name** | **Find Matching** |
| **Participating Actors** | Student. |
| **Trigger** | The Student clicks on “Search” button |
| **Entry Conditions** | Student has existing profiles under his account and is logged in the system. |
| **Flow of Events** | 1. Students will select a class for which they need a group 2. Once they are in the class where they want to find a group, students have the option to search for a group that currently exists. 3. If he decides that none of the current groups complies with the student criteria, the student has the option to create his own group by going under “Groups” and click “Create Group”. 4. When a student is searching for a group, or looking team members for the group he created, he can see the profile of each student that currently has no group in one side of the page. 5. Students may also see the profile of the team members of formed groups. 6. Finally, a student may have to send a request. Either he sends a request to join a group he would like to be part of, or he receives a request from a group to be member of. 7. The system will be in charge to send you emails about these requests. 8. This runs iteratively until the user decides to stop. 9. User may only belong to a single group for each class. |
| **Alternative Paths** | 1. None |
| **Exit Conditions** | Student has successfully search for available groups for a particular class.  Student has knowledge to make a decision in joining, or creating a group. |

|  |  |
| --- | --- |
| **Use Case Name** | **View Groups** |
| **Participating Actors** | Student. |
| **Trigger** | The Student goes under “Groups” and then on “View Groups” . |
| **Entry Conditions** | Student has existing profiles under his account and is logged in the system. |
| **Flow of Events** | 1. Just click on “Groups” and then click on “View Groups”. 2. A list will appear with all the groups in which you belong to. |
| **Alternative Paths** | 1. None |
| **Exit Conditions** | Student has successfully seen the groups he has currently joined. |

|  |  |
| --- | --- |
| **Use Case Name** | **Make Changes to Group** |
| **Participating Actors** | Student. |
| **Trigger** | The Student goes under Groups and click on any of the following three choices: “Create Group”, “Edit Group” or “View Groups”. |
| **Entry Conditions** | Student has existing groups under his account and is logged in the system.  Student belongs to a group and/or is owner of a group. |
| **Flow of Events** | 1. Student will choose a group he belongs to and make any changes. 2. If the student is the owner, he can delete the group. 3. In addition, group owners will be able to reject a request to join the group, remove a member of the group, and make several changes to the properties of the group. 4. If a student is a member of the group and not the owner, he has the power to leave the group anytime he wants. |
| **Alternative Paths** | 1. User clicks in view groups to see all the groups he belongs to. 2. User click on create group and make a group. |
| **Exit Conditions** | Student has successfully has done any changes of his desire and on his power to the group he belongs/ owns.  Student has been able to create or view groups he created or belongs to respectively. |

|  |  |
| --- | --- |
| **Use Case Name** | **Logout** |
| **Participating Actors** | Student. |
| **Trigger** | Student decides to logout |
| **Entry Conditions** | Student is logged in the system |
| **Flow of Events** | 1. Student clicks on the logout button. |
| **Alternative Paths** | 1. Student simply closes the web browser. |
| **Exit Conditions** | Student has successfully logged out from the system. |

|  |  |
| --- | --- |
| **Use Case Name** | **Configure System** |
| **Participating Actors** | System Administrator |
| **Trigger** | The system is set to run for the first time to provide service.  A request from owner of the owner of the system decides that the current system needs some changes. |
| **Entry Conditions** | System is being set up for the first time or configuration changes are requested. |
| **Flow of Events** | 1. Admin will enter the menu of system settings for a specific component of the system and modify any aspect of it. |
| **Alternative Paths** | 1. None |
| **Exit Conditions** | System Administrator has successfully configured the system to his desire. |

|  |  |
| --- | --- |
| **Use Case Name** | **Maintenance and Updates** |
| **Participating Actors** | System Administrator |
| **Trigger** | The system presents signs of needs for maintenance.  New updates are available to the components that conforms the system. |
| **Entry Conditions** | System administrator has root control of each MatchMaking system component. |
| **Flow of Events** | 1. Admin will click on the maintenance and updates button of the component. 2. He will then choose what part of the system he would like to update. 3. The components he has to pick for maintenance and updates are: Database, Utilities, System, and OS. |
| **Alternative Paths** | 1. None |
| **Exit Conditions** | System Administrator has successfully given the necessary maintenance and updates to the system. |

|  |  |
| --- | --- |
| **Use Case Name** | **Access System** |
| **Participating Actors** | System Administrator |
| **Trigger** | System administrator enters into the system components, automatically he has root control. |
| **Entry Conditions** | System administrator has root control of the system. |
| **Flow of Events** | 1. Admin has access of the system once he has accessed the system through his web browser or other method to access a specific component of the system. |
| **Alternative Paths** | 1. None |
| **Exit Conditions** | System Administrator has root control and can access any part of the system  . |

|  |  |
| --- | --- |
| **Use Case Name** | **Assume Student Role** |
| **Participating Actors** | System Administrator |
| **Trigger** | System Administrator decides to be a student for testing purposes or to do something requested by his superior. |
| **Entry Conditions** | System administrator logs in to the MatchMaking system using his FSU email and password. |
| **Flow of Events** | 1. Once logged in, system treats him as a student. 2. Admin can start doing anything that a student can do. |
| **Alternative Paths** | 1. Logouts and he no longer will act as student. |
| **Exit Conditions** | System Administrator has successfully acts as a student and can do all the things that a student can do too. |

1. **Other Appendices**
2. **Sequence Diagrams**

**Student Sequence Diagram**

**The following pages have the Student and System Administrator Sequence Diagrams**

**After the Sequence Diagrams, we have the Database ER diagram.**

**.**

Member\_of

Belongs\_to

Is\_a

Is\_a

Product

Enrolled\_In

Searches

Students

Class

Group

Profile

Creates