Software Requirements Specification

for

A Portal for Managing Students Capstone Projects

Version 1.0

Prepared by Wajeeh Anwar and Suhail Purkar

11/26/2018

Table of Contents

1.Introduction

1.1.Purpose

The purpose of this document is to give a detailed description of the requirements for the "Portal for Managing Students Capstone Projects" software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications. Document Conventions

1.2.Intended Audience and Reading Suggestions

This document is intended to be used by a(n):
1. Customer for its approval and a reference for developing the first version of the system.
2. Instructor user for guides on using and limitations of the system.
3. Student user for guides on using and limitations of the system.

1.3.Product Scope

The "Portal for Managing Students Capstone Projects" is a wed-based application which helps manage projects for my software engineering/ development courses. The application should be accessible via the course websites.

Instructors can:
1. Log in to administrator account.
2. Upload projects by course
3. Create and delete student user accounts by course
4. Pair match students to project groups

Students can:
1. Log in to student account
2. Review uploaded projects by course
3. submit their project preferences

Furthermore, the software needs to be run on a device via a web browser and have an Internet connection to fetch and display results. All system information is maintained in a NoSQL firebase database, which is located on Amazon's web-servers.

1.4.References

"Firebase Realtime Database | Firebase Realtime Database | Firebase." *Google*, Google, firebase.google.com/docs/database/.

2.Overall Description

2.1.Product Perspective

This system will consist of one web portal which will be used for managing the information about the projects. Since this is a data-centric product it will need somewhere to store the data. For that, an Amazon firebase database will be used. The web portal will communicate with the database to get, add and modify data. All of the database communication will go over the Internet.

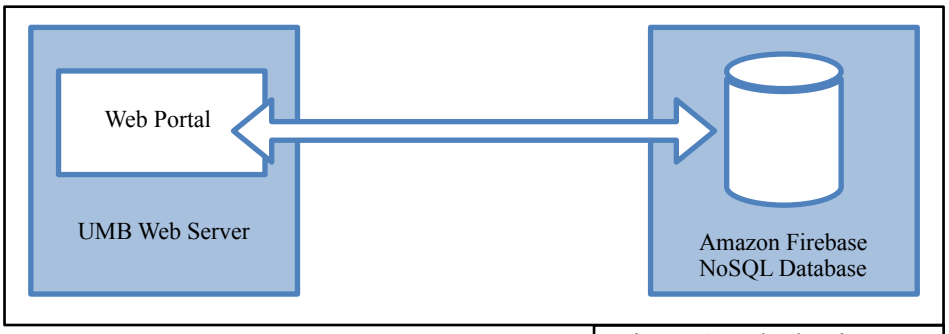


Figure 1 - Block Diagram

2.2.Product Functions

Students will be able to browse projects. The projects displayed will be based on the course assigned to the student. The projects will be viewed in a list view which will have one list item for each project matching the course and show a small part of the project information so the student can identify the project, along with an option to assign a preference rank to the project.

Instructors will be able to create/modify/delete projects and students. The projects will be viewed in a list view which will have one list item for each project matching the course and show a small part of the project information so the instructor can identify the project, and modify or delete it. Additionally the instructor will be able to create a new project and assign it a course. The instructor will be able to run the project-student pairing algorithm.

2.3.User Classes and Characteristics

There are two types of users that interact with the system: course instructors and course students. Both of these types of users have different use of the system so each of them has their own requirements.

1. The students are reviewing the available course projects and submitting their preferences.
2. The instructors are managing the overall system so there is no incorrect projects, courses or students within it. The instructors can manage the information for each project as well as each student.

2.4.Operating Environment

The environment that the web portal will operate in will be a UNIX web server communicating across many operating systems via a web browser. The environment that the database will operate in is an Amazon AWS cloud environment.

2.5.Design and Implementation Constraints

The web portal is constrained by the system interface Since there are multiple system we browsers, the interface will most likely not be the same for every one of them. The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function. The web portal will also be constrained by the capacity of the database. Since the database is external.

2.6.User Documentation

None.

2.7.Assumptions and Dependencies

One assumption about the product is that it will always be used on a computer, as opposed to something like a mobile phone.

Another assumption is that students will only be enrolled in one course. Although the system can handle multiple course entries by the instructor creating a separate student account for each course.

3.External Interface Requirements

3.1.User Interfaces

A user of the web portal should see the log-in page when he/she navigates via the link from their course page, see Figure 2. After logging in, if the user is a(n):

1. Student, they should arrive to a list view of the projects available, see Figure 3
    - Each project has a ‘view more’ option , see Figure 3a, which includes a project title, project description, project requirements, resources, client contact information.
    - Each project also gives the option for the student to rank its preference.

2. Instructor, they should arrive to an

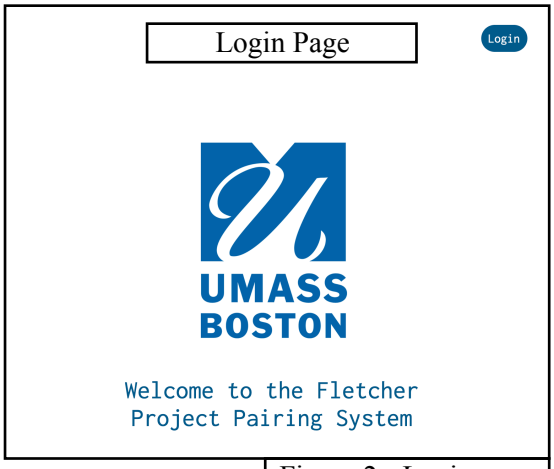


Figure 2 - Login page

administrative page, see Figure 4, where a link is present for:

- Creating student user logins. Creation of a student user includes assigning an email address, password, course and administrator privilege. * Student name will be derived from the email address. For example: an email address of frank.sinatra001@umb.edu will automatically set the student name to 'Frank Sinatra'. See Figure 4a.
- Creating projects. Creation of a project includes assigning a project title, project description, project requirements, resources, client contact information and course association. A project ID will automatically be assigned based on the number of existing projects. See Figure 4b.
- Executing the student-project pairing algorithm.
- Viewing the students assigned to each project. See Figure 4c
- Sending students emails with information about their assigned project and student team members.
- Resetting the semester by deleting all users and projects.

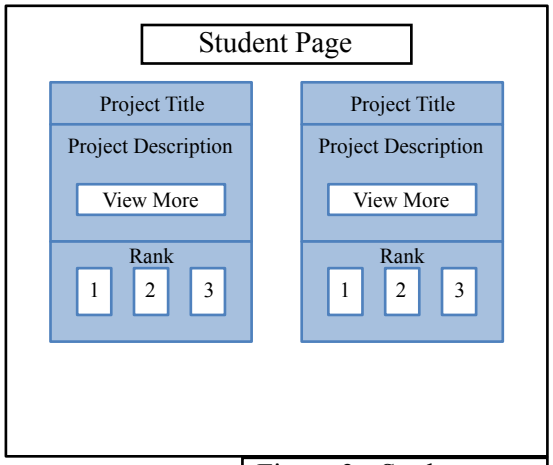


Figure 3 - Student page

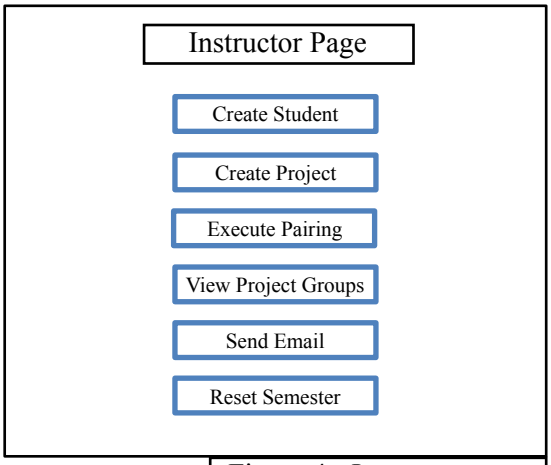


Figure 4 - Instructor page

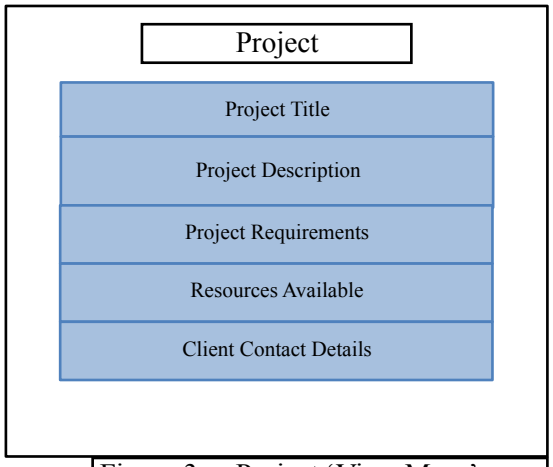


Figure 3a - Project 'View More' page

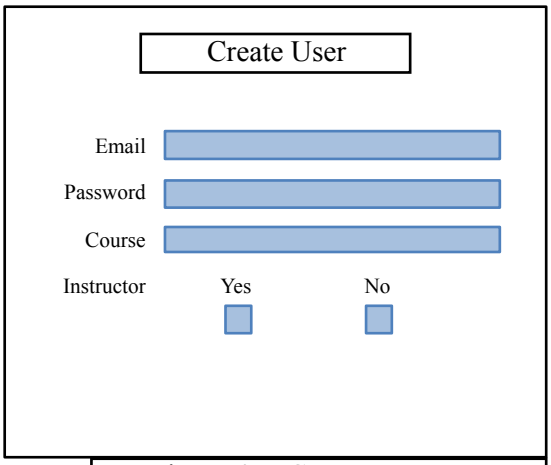


Figure 4a - Create User page

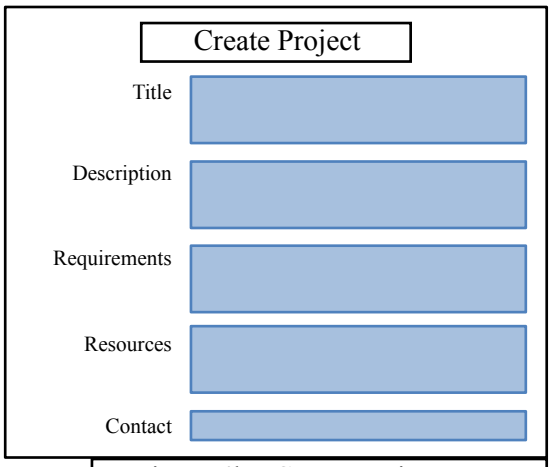


Figure 4b - Create Project page

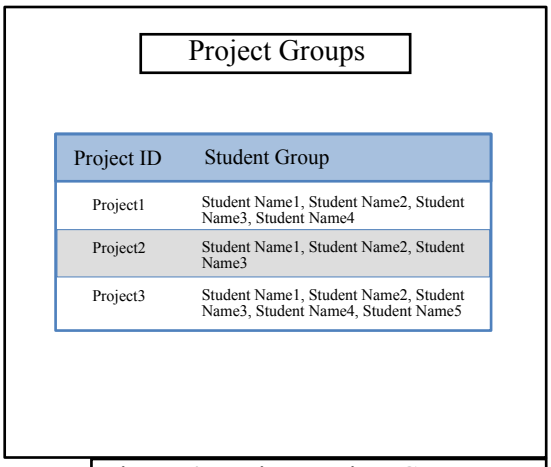


Figure 4c - View Project Group page

3.2.Hardware Interfaces

The web portal does not have any designated hardware, therefore it does not have any direct hardware interfaces.

3.3.Software Interfaces

The communication between the database and the web portal consists of operation concerning both reading and modifying the data.

3.4.Communications Interfaces

Although the communication between the different parts of the system is important since they depend on each other, the communications is handled underlying operating systems and browsers, as how the communication takes place is less important.

4.System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1.SR1: User log in

4.1.1 In order to use the system a student should be logged in to the web-portal. This is a high priority.

4.1.2 Stimulus/Response Sequences

User provides email address and password.

Result: System verifies log in with 3 outcomes:

1. System does not find the credentials provided by the user in the database.

   Action: System displays an error requesting new credentials.

2. System finds credentials with a true value for administration

   Action: System sets current user, loads and displays Instructor page.

3. System finds credentials with a false value for administration and course value.

   Acton: System sets current user, loads and displays Student page associated with course.

4.1.3 Functional Requirements

System must be able to communicate with database.

4.2.SR2: Student view projects

4.2.1 In order to rank project preferences, a student must first be able to view the details of each project available.

This is a high priority.

4.2.2 Stimulus/Response Sequences

System retrieves the course associated projects from the database and displays them

4.2.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Student'

REQ-1: SR1

4.3.SR3: Student select/rank project preferences

4.3.1 In order to be paired to a project, a student must first rank the available projects by their preference

This is a high priority.

4.3.2 Stimulus/Response Sequences

Student submits the preference ranks of each project.

Result: System verifies student rank selection with 3 outcomes:

1. Verification fails because ranks are not distinct and sequential.

Action: System displays an error and does nothing.

2. Verification passes but user has already previously submitted preferences.

   Action: System overwrites current user rank preferences.

3. Verification passes and user has not already submitted preferences.

   Acton: System sets current user rank preferences.

4.3.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Student'

REQ-1: SR1
REQ-2: SR2

4.4.SR4: Instructor create user

4.4.1 In order for students to view projects, a user account must be created for them.

This is a high priority.

4.4.2 Stimulus/Response Sequences

Instructor submits user email address, password, course number and admin status.

Result: System verifies information with 3 outcomes:

1. Email address fails pattern matching.

   Action: System displays an error and redisplays the 'create user' page.

2. Email address passes pattern matching and email address already exists.

   Action: System overwrites information associated with existing email address found.

3. Email address passes pattern matching and email address does not already exists.

   Acton: System adds information to user database.

4.4.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'

REQ-1: SR1

4.5.SR5: Instructor create project

4.5.1 In order for students to view projects, project must be created first

This is a high priority.

4.5.2 Stimulus/Response Sequences

Instructor submits project title, project description, project requirements, resources, client contact information and course association.

project title, project description, project requirements, resources, client contact information and course association. A project ID will automatically be assigned based on the number of existing projects.

Result: System adds information to project database

4.5.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'

REQ-1: SR1

4.6.SR6: Instructor modify project

4.6.1 In order for instructor to modify projects, the project must be created first

This is a medium priority.

4.6.2 Stimulus/Response Sequences

Instructor selects a project and clicks on the update button.

Result: System loads a create project page with fields already populated with the selected projects information from the database.

Action: Instructor edits the fields as needed and clicks on submit.

Result: The system overwrites the project associated information in the projects database

4.6.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'
Project must already exist

REQ-1: SR1
REQ-2: SR5

4.7.SR7: Instructor execute student-project pairing

4.7.1 In order for students to be assigned a project, the instructor must execute the student-project pairing algorithm.

This is a high priority.

4.7.2 Stimulus/Response Sequences

Instructor click on the 'execute pairing' button.

Result: System retrieves all user project preference data from the database and passes it to the pairing algorithm script which returns

an association list projects along with assigned students. The system then updates both user and project databases with this information.

4.7.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'
Projects, users and preferences must already exist.

REQ-1: SR1
REQ-2: SR2
REQ-3: SR3
REQ-4: SR4
REQ-5: SR5

4.8.SR8: Instructor view student-project pairing

4.8.1 In order to verify and email students about the their project pairing, the instructor must view the student-project pairing results.

This is a medium priority.

4.8.2 Stimulus/Response Sequences

Instructor click on the 'view project groups' button.

Result: System displays the 'project group' page.

4.8.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'
Student-project pairing algorithm must have already been executed.

REQ-1: SR1
REQ-2: SR7

4.9.SR9: Instructor email students

4.9.1 In order to inform students about the their project pairing, the instructor must email the student-project pairing results.

This is a low priority.

4.9.2 Stimulus/Response Sequences

Instructor click on the 'email students' button.

Result: System uses php to email all users of their assigned project and team members.

4.9.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'

Student-project pairing algorithm must have already been executed.

REQ-1: SR1
REQ-2: SR7

4.10.SR9: Instructor reset semester

4.10.1 In order to initialize the system for a new semester, the instructor must reset the system database.

This is a medium priority.

4.10.2 Stimulus/Response Sequences

Instructor click on the 'reset semester' button.

Result: System deletes all user and project data from the databases.

4.10.3 Functional Requirements

System must be able to communicate with database.
User must be logged in and identified as 'Instructor'

REQ-1: SR1

4.11.Performance Requirements

4.11.1 Access and retrieval from database must be within 2 seconds

4.11.2 Write to databases but be within 1 second for realtime updating

4.11.3 Execution of student-project pairing algorithm must be less than 1 minute in order to keep active session to database alive.

4.12.Safety Requirements

4.12.1 Database must be cloud based in order to reduce the risk of data loss

4.13.Security Requirements

4.13.1 All message passing between web-server and database must be encrypted in order to secure user and database login credentials.

4.14.Software Quality Attributes

4.14.1 Availability of web portal and database must be 99%.

4.15.Business Rules

4.15.1 Students must only access the system during their designated project selection period.

4.15.2 Instructors must reset semester once student-project pairing has been finalized.

5.Other Requirements

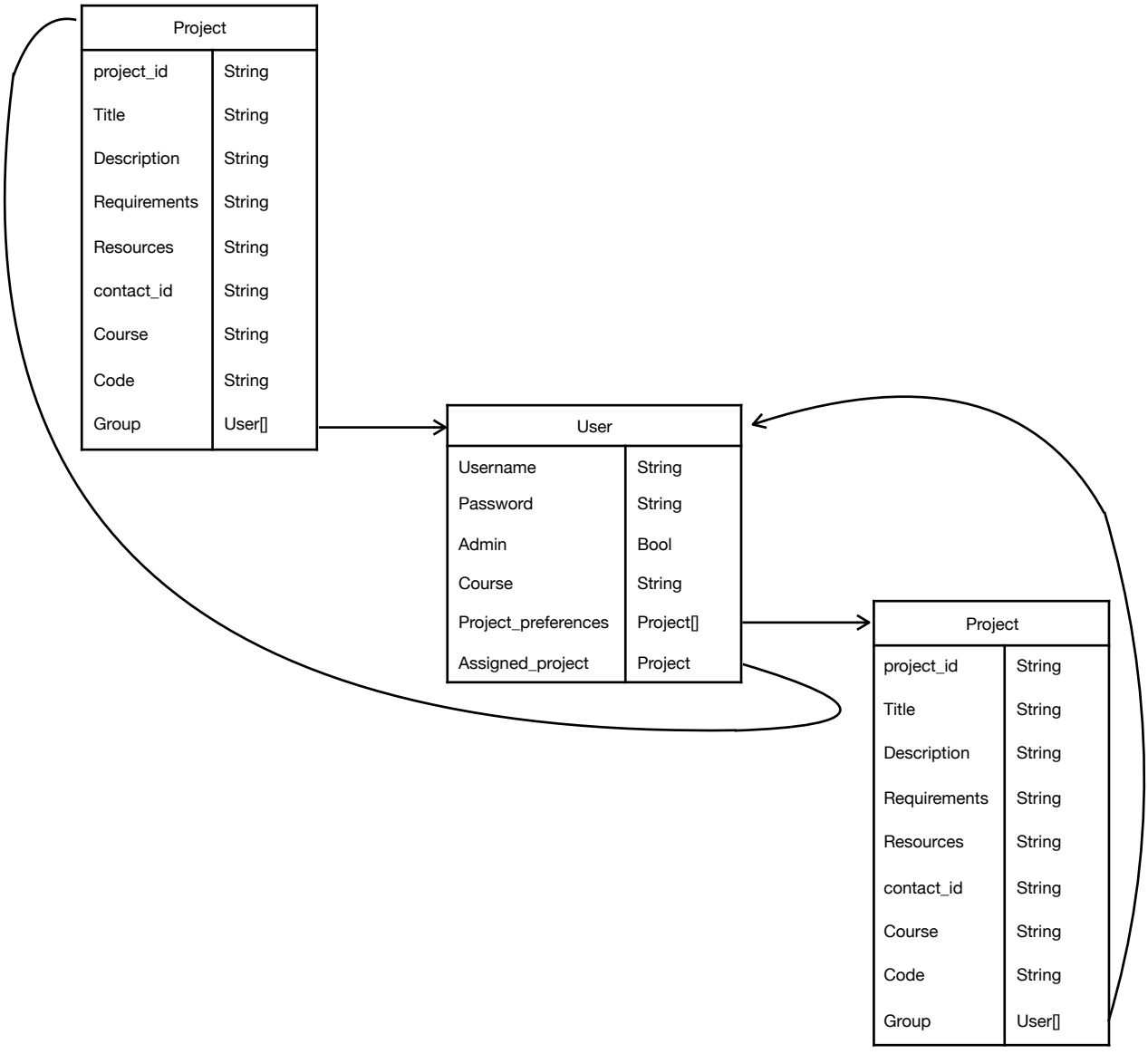
None.

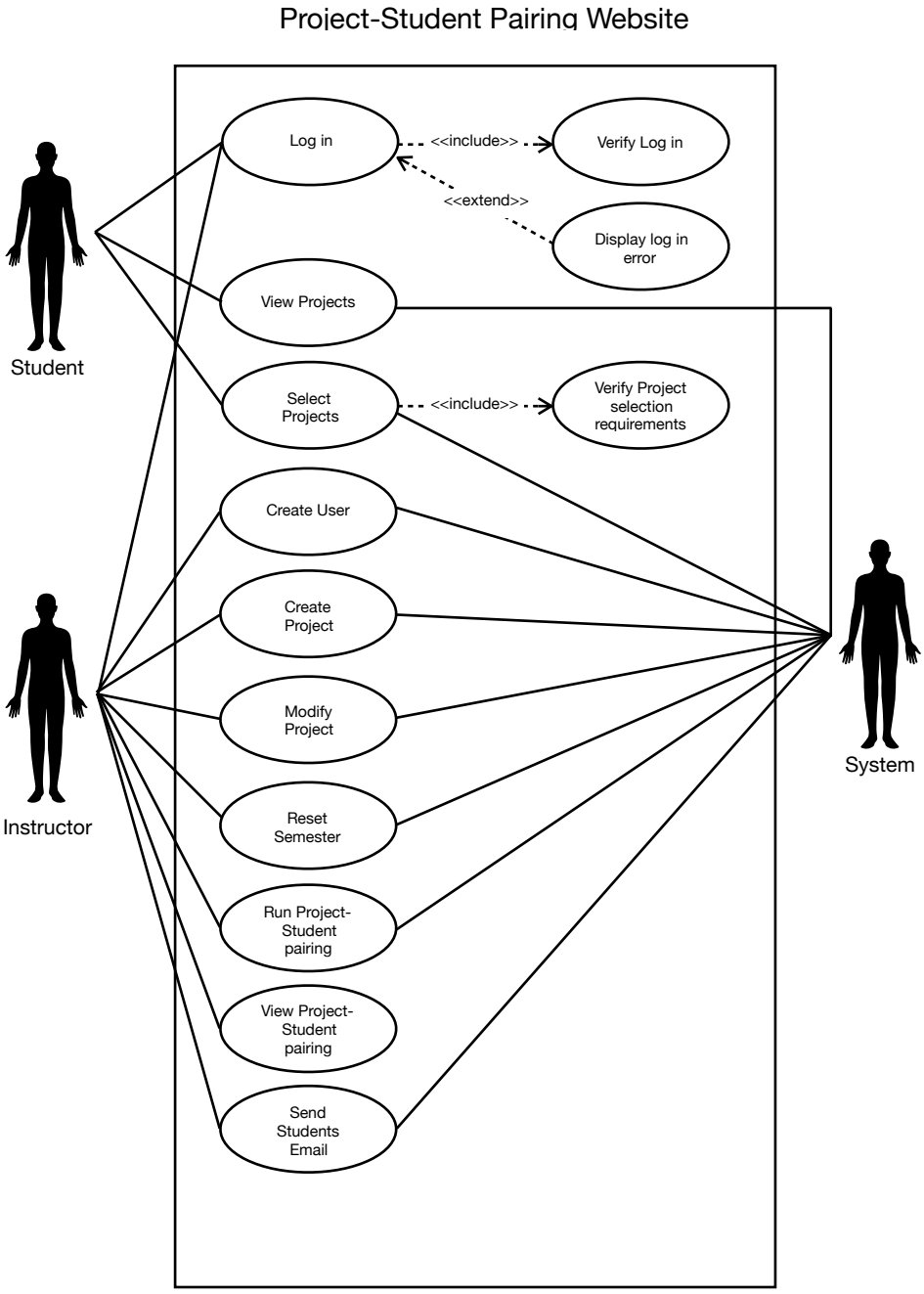
Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

NoSQL Firebase Database Scheme





Appendix C: To Be Determined List

None.