*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Final Deliverable

Vertically Integrated Projects 4.0:

**Team Members:**  Joseph Gonzalez, Shefali Nakum, Nandini Dandu, Ravi Kiran, and Franco Hernandez

**Product Owner(s)**: Mohsen Taheri

**Mentor(s)**:

**Instructor**: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) *2016 Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

***Abstract***

*This document presents the information necessary to gain a good understanding of the VIP 4.0 web app. This document will cover all the features and updates made to the VIP web app in the latest final version. Show diagrams with how these new features affect the system, and how they address several problems that existed on the site, as well as present a wish list, and backlog for future iterations of the site.*

*VIP 4.0 serves the expand on the successes of its predecessors by expanding the existing features of the site, as well as creating brand new features altogether in the name of*

*making a better environment in which both faculty and students can come together to work on innovative projects.*

*Among the updates to the site include, but are not limited to, a site wide update for skill items that make it easier for professors to specify which skills they are looking for from students, and for students to better match their skill sets with projects; a system of locks that allow admins to lock specific project and even entire semester's; a system to generate user reports at the click of a button on excel; and more! Please read below and enjoy*

**Table of Contents**

**Introduction** ……………………………………………………………………………………………………………………………….. 5

Current System ………………………………………………………………………………………………………………………... 5

Purpose of New System ……………………………………………………………………………………………………………... 6

**User Stories**

Implemented User Stories …………………………………………………………………………………………………………..7

Pending User Stories …………………………………………………………………………………………………………..….. 7

**Project Plan**

Hardware and Software Resources ………………………………………………………………………………………….…8

Sprints Plan …………………………………………………………………………………………………………………………. 13

*Sprint 1*  …………………………………………………………………………………………………………………………... 8

*Sprint 2*  …………………………………………………………………………………………………………………………... 8

*Sprint 3*  …………………………………………………………………………………………………………………………... 17

*Sprint 4*  …………………………………………………………………………………………………………………………... 32

*Sprint 5*  …………………………………………………………………………………………………………………………... 44

*Sprint 6*  …………………………………………………………………………………………………………………………... 47

*Sprint 7*  …………………………………………………………………………………………………………………………... 47

**System Design**

Architectural Patterns ………………………………………………………………………………………………………….. 48

System and Subsystem Decomposition ……………………………………………………………………………………….… 48

Deployment Diagram ………………………………………………………………………………………………………….…... 49

Design Patterns ……………………………………………………………………………………………………………….….... 50

**System Validation**  …………………………………………………………………………………………………………………….51

**Glossary**  ………………………………………………………………………………………………………………………………….67

**Appendix**  ………………………………………………………………………………………………………………………………….68

Appendix A - UML Diagrams ……………………………………………………………………………………………………. 68

*Static UML Diagrams*  ……………………………………………………………………………………………………….68

*Dynamic UML Diagrams*  …………………………………………………………………………………………………..68

Appendix B - User Interface Design ……………………………………………………………………………………….…... 90

Appendix C - Sprint Review Reports ……………………………………………………………………………………...…… 91

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents …………………………………………………………………….…………… 93

**References** ……………………………………………………………………………………………………………………...………...142

# 

# Introduction

VIP 3.0 brought in various major improvements to the site, but with all things, there is room for improvement. In VIP 4.0, several updates and improvements were added to the site to expand the overall usability of the site. This includes updating and fixing existing features, as well as the creation of brand new features to the site. This includes, but is not limited to new ways to manage projects, formatting for the site, data organization, a new archiving system, and new report generating features.

## 

## Current System

Before we got started, VIP 3.0, while a strong and dynamic site, lacked several features required by the product owner. There was no means to manage a project long term, the site required a more refined formatting method, admins needed more tools to control the site, and many of the forms throughout the site were missing needed data. There were also problems with the server, as the live and dev servers shared a database, which would cause problems during testing updates. The database itself was also having issues as there existed redundancies in the database, and necessary fields were not being utilized by the system.

## Purpose of New System

The new system sought to update and refine many of the existing features, such as updating the formatting of project descriptions, getting the dev server on a separate site, and add several new features to expand project management functionality. The final result is a more refined VIP site, that allows admins more control over the content on the site, allows students and professors to find projects that better suit their skills and needs, and allows for long term management of projects over multiple semesters.

# User Stories

The following section provides the detailed user stories that were implemented in this iteration of the …. project. These user stories served as the basis for the implementation of the project’s features. This section also shows the user stories that are to be considered for future development.

## Implemented User Stories

* Adding formatting to description in Project description
* Add Mingle section to project proposal form
* Add Skill Items in student application form
* VIP4 Version control
* Update Email Services
* Add Faculty Registration Page to the site
* Add text box whispers to social icons
* Search name in admin panel
* Divide login page
* Add Lock Project features
* Add Lock Semester features
* Add excel report generator in admin panel
* Add search bar in Projects page
* Update project filters
* Add Semester field to project database
* Update Terms collection in database
* Move Dev Server to separate site on separate database
* Make fields required in all forms
* Add guideline videos to login page

## Pending User Stories

* Add tabs to projects page to separate project by semester

# Project Plan

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

## Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

* HTML5 and CSS3 for web design
* AngularJS for server and directives
* MongoDB for the database
* NodeJS for collections and models
* Github for Version Control
* An Ubuntu-based Linux server run by FIU
* Putty and WinSCP for file sharing and SSH client

## 

## 

## Sprints Plan

### Sprint 1

Sprint 1 served to get the team up to speed on the system, in that regard, no features were designed in this sprint, and instead the goal was for every developer to get up to speed on the site

### Sprint 2

User Story #1152

# Mingle Filed and Logo

# User Story - Testing and Creating Mingle Filed Entry and Social Icon in VIP Projects

* As a User, I would like to be able to have a link on the project page that sends me directly to the Mingle site

## Use Case

**Use Case ID:** (VIP-1152)

**Details:** Adding a Mingle URL entry in the “Project Proposal Form” and a Mingle Social Icon on the “Project Profile Page”.

**Actor:** Users

**Pre-Condition:**

* User has navigated the VIP site
* One User must be a “Faculty/Staff or Pi/CoPI” level user
* Other Users must be a member of the Project which Project Profile Page they are accessing.

**Description:**

1. Use Case begins when a Faculty/Staff or PI/CoPI navigates to the “Project Proposal Form” (<http://vip.fiu.edu/#/project-proposal>)
2. User fills out the form, and inputs a link in the option labeled “Mingle Link”
3. User submits Project for Approval
4. PI/CoPI approve the new project
5. The Product Owner/Project Members navigate to the “Project Page” (<http://vip.fiu.edu/#/vip-projects>)
6. The Product Owner/Project Members navigate to the “Project Details Page” (<http://vip.fiu.edu/#/vip-projects-detailed/><uniqueProjectHashHere>)
7. Under the section labeled “Discover More” will be any links that were created in step 2
8. Users click on the Social Icon Link “M” that corresponds to Mingle

**Post-Conditions:**

1. Users are redirected to their “Mingle Login Page” (<https://fiu-scis-seniorproject.mingle.thoughtworks.com>)
2. From here Users can login and access their project’s Mingle page

**Alternative Course of Action**

N/A

**Exceptions:**

1. User is not a Project Member or Product owner for the Project they are trying to access
2. Project Owner did not set the “Mingle Link” in the “Project Proposal Form”

**Related Use Cases:**

N/A

**Decision Support:**

**Frequency:** High: Users need to access Mingle often.

**Criticality:** Low: No immediate effect on the rest of the sites performance

**Risk:** Low.

**Constraints:**

1. **Usability:** 
   1. No training time required.
   2. User guide should provide more in-depth information.
2. **Reliability:** 
   1. Mean time to Failure – 1% failures for every twenty four hours of operation is acceptable.
   2. Availability – Dependent on Users
3. **Performance:** 
   1. Link takes less than 5 seconds to redirect users to Mingle Page
4. **Supportability:**
   1. All modern web browsers should be supported

User Story #1146

Make login permanently on the user's system after first login

# User Story – Make login permanently on the user's system after first login

* As a user, I would like to be able to be permanently be logged after the first login.

## Use Case

Use Case ID: (VIP-1146)

Details: Creating a remember\_me checkbox and a cookie that remember user id and password for a determined period of time.

Actor: Users.

Pre-conditions:

* User has navigated to the VIP website.
* User must be staff member or Pi/CoPI user.
* User must check checkbox and login for first time

Description:

1. Use case begins when the user navigates to the VIP website login page
2. The user enter user id and password
3. The user check the remember me checkbox
4. The user click on the signin button

Post-conditions:

1. Users should be able to stay logged for a week

Alternative Courses of Action:

N/A

Exceptions:

N/A

Related Uses Case:

N/A

**Decision Support:**

**Frequency:** Low: Users only need to check the remember\_me checkbox once every two weeks

**Criticality:** Low: No immediate effect on the rest of the sites

**Risk:** Low.

**Constraints:**

1. **Usability:** 
   1. No training time required.
   2. User guide should provide more in-depth information.
2. **Reliability:** 
   1. Mean time to Failure – 1% failures for every twenty four hours of operation is acceptable.
   2. Availability – Dependent on Users
3. **Performance:**
   1. Request should be sent and saved within 5 secs.
4. **Supportability:**
   1. All modern web browsers should be supported.

User Story #1142

# Add format to "PROJECT PROPOSAL FORM" 's team description

User Story - Add format to "PROJECT PROPOSAL FORM" 's team description

● As a user, I would like to view the description of project in the same format as the professor has entered so that it would be easy to understand.

## Use Case

**Use Case ID:** (VIP-1142)

**Details:** Add format to "PROJECT PROPOSAL FORM" 's team description

**Actor:** Faculty

**Pre-conditions:**

1. Faculty has navigated to the VIP website.

2. Faculty is browsing the VIP website as any user(student, Faculty, or admin) after logging in.

3. Faculty should see different projects in the website and also should view individual project by clicking on a project.

**Description:**

1. Use case begins when the User opens the vip website

2. The User may or may not sign in to the website

3. User should see all the projects in the Projects page with descriptions in good format.

4. If User clicks on any project then the project detail page will be with project description in the exact same format that has been entered during proposal of a project.

5. If Faculty wants to propose a project then he can select propose a project form and should find toolbar above the text area provided to enter description.

6. This format should be kept same throughout the website wherever it displays it.

**Post-conditions:**

1. Users should find description in the same format that has been entered during proposal of a project

2. Users should see the projects in the home page with good format

3.. Faculty can make use of tool bar above the project description box while proposing a project

User Story #849

Using VIP Team Filter

# User Story – Using VIP Team Filter

* As a developer, I would like to have filter options that would sort the projects in ascending and descending order.

**Use Case**

**Use Case ID:** (VIP-849)

**Details:** Using VIP team Filter Proposed Filter

**Actor:** Developer

**Pre-conditions:**

1. Developer has navigated to the VIP website.
2. Developer is browsing the VIP website as a admin User and trying to sort the projects based on the ascending to descending order
3. Developer should see the projects in their respective order.
4. After that the developed would select the project that they like and have it join it.

**Description:**

1. Use case begins when the Developer navigates to the “VIP website” (http://vip.fiu.edu/#/login).
2. The user can sign or else normally can browse the website for the projects to apply for.
3. When the user selects the filter by option, he should get the drop down with the options and should have the options (A-Z) or (Z-A).
4. The user would select according to his/her preference.
5. They need to fill a form and submit to get enrolled in that project.

**Post-conditions:**

1. Users should be able to filter the projects when they select the filter by option.
2. They should be able to select them and register in them as well.

User Story #1144

**User STORY:**

**Add Skill item in student application form and faculty proposal form.**

### Description:

* As a user, I would like to have a separate text box for skill item so that student can enter the skills that they have and professors can enter the skills that project requires.

### Acceptance Criteria:

1. While applying for project, developer has to access the website as a student.
2. While proposing for a project, developer has to access the website as a faculty.

**Use Case:**

### Participating Actors: Student and Faculty/Staffs. Entry Condition:

### Developer will navigate to the VIP website.

### Developer will log in to the VIP website.

### Events Flow:

**Part A**

* If Developer logs in as a Student and wish to apply for a project, then he will go to “Apply for a Project” form.
* Developer will notice a new field called “Skill Item” where he can enter the skills he is good at.
* When Developer enters the skills in the application form, this skill will reflect in the Developer’s Profile information with the same label called “Skill Item”.
* This Skill Item can be viewed in the project detailed information by clicking on the student name.
* This Skill Item can also be viewed by the admin in review user and can approve based on the Skill Item.

**Part B**

* If Developer logs in as a Professor, and wish to propose a project, then he will go to “Propose a Project” tab.
* Developer will then fill the form and notice a new field called “Skill Item” where he can enter the desired skills he is looking in the students.
* This information can be viewed in VIP Project detailed information.

### Sprint 3

*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

User Story #1160

# Add Text Box Whispers to Social Icon Links

# User Story - Add Text Box Whispers to Social Icon Links

* As a User I’d like to be able to hover the social icons on the site, and see a text whisper of what that Icon is a link to

## Use Case

**Use Case ID:** (VIP-1160)

**Details:** Adding text box whispers to social icon links

**Actor:** Students, Faculty/Staff, PI/CoPI

**Pre-Condition:**

* Project Proposer filled out any of the Youtube, Github, Google, or Mingle links in “Project Proposal Form”
* User is a member of the Project which “Project Details Page” they have navigated to

**Description:**

1. User has navigated to the “Project Details Page” (<http://vip.fiu.edu/#/vip-projects-detailed/><UniqueProjectHash>)
2. User navigates to the “Discover More” section of the “Project Details Page”
3. User places mouse over any of the Social Icon Links in the “Discover More” Section
4. Text Box Whisper appears with the name of the Social Icon Link next to cursor

**Alternative Course of Action**

N/A

**Exceptions:**

N/A

**Related Use Cases:**

N/A

**Decision Support:**

**Frequency:** High: Once created links will always be present on the Project Details Page

**Criticality:** Low: This feature has no impact on the rest of the sites performance

**Risk:** Low.

**Constraints:**

1. **Usability:** 
   1. Requires no experience
2. **Reliability:** 
   1. Availability – Dependant on Project Proposer
3. **Performance:** 
   1. Text box whispers appear within a second of being cursed over
4. **Supportability:**
   1. All modern web browsers should be supported.

User Story #1153

# VIP4 Version Control

# User Story - VIP4 Version Control

* As a developer, I would like to make sure all members of the team are using the right code for Implementation

## Use Case

**Use Case ID:** (VIP-1153)

**Details:** Adding and organizing an Integration Branch to the shared VIP4 Github repository

**Actor:** Developers

**Pre-Condition:**

* Developer is a member of the VIP4 Project
* Developer has access to the shared Github Repository
* An Integration Branch has been created on Github

**Description:**

1. Use case begins when a developer’s feature has been considered “done” by the project owner.
2. The developer then proceeds to upload any code changes they’ve made to their personal branch on the VIP4 github (<https://github.com/FIU-SCIS-Senior-Projects/Vertically-Integrated-Projects-VIP-Ver-4.0>)
3. Developers then proceed to add their changes to the features changelog in the shared Google Drive (<https://drive.google.com/drive/u/0/folders/0BwRxNPGpQTNydGdmQkFSX1o0T1E>)
4. The developer in charge of the Integration Branch applies each developer's changes to the original code
5. The integration developer then tests all updates on their localhost to ensure no issues occurred between the features.
6. These changes are then uploaded to the Integration Branch on Github

**Post-Conditions:**

1. All developers now have access to an updated version of the sites code
2. This code must be re-updated after each sprint

**Alternative Course of Action**

N/A

**Exceptions:**

1. If a developer's features were not approved by the product owner, they will not be added to the integration branch

**Related Use Cases:**

* Maintaining a personal Changelog

**Decision Support:**

**Frequency:** Medium: Updates occur every 2 weeks.

**Criticality:** Medium: Updates needed to make sure future features aren’t missing any portions of the site.

**Risk:** Medium. Mistakes could cause update losses between developers code.

**Constraints:**

1. **Usability:** 
   1. Requires experience in Github
   2. One Developer must take on extra regular work
2. **Reliability:** 
   1. Github has an upload contraint of 25MB or 100 files at a time, that can be a hinderance on updating the site. Can be bypassed by using Git from the command line.
   2. Availability – Every other week.
3. **Performance:** 
   1. Downloading the contents can take several minutes, and uploading files can take some time as well.
4. **Supportability:**
   1. All modern web browsers should be supported

User Story #1172

# Dividing the webpage into multiple sections and adding guideline videos

**User Story**

·   
As an admin , I would like to divide the web page into sections, one for students and the other one for the faculty. Along with that I would like to add guideline videos, that would be helpful for the students and the faculty.

**Details:**  
Divide the login webpage into two sections and adding respective guideline videos to them.

Actor: Faculty, Student

**Pre-Condition:**

We have to consider the faculty and student. Both of them are on the login page.

**Description:**

1. The use case will begin when the student would be on the login page. The students just have to click on the Google Icon as they the FIU email is already linked to Gmail. They do not need to create an account. If the students do not know something they would watch the user guideline video which is next to the google icon.

2. For the faculty they would have to enter their username and password. Similarly the faculty have the guideline videos as how to go through the website.

**Post-Conditions:**

1. Students/ Faculty must be able to access the VIP website without any issues.

User Story #1140

# Add Faculties registration page to the website

# 

# 

# User Story - Add Faculties registration page to the website

● As a Website Admin(Pi/CoPi), I would like to view the list of faculties registration requests.

## Use Case

**Use Case ID:** (VIP-1140)

**Details:** Add Faculties registration page to the website

**Actor:** Admin(Pi/CoPi)

**Pre-conditions:**

1. Admin(Pi/CoPi) has navigated to the VIP website.

2. Admin(Pi/CoPi) is able to login.

**Description:**

1. Admin(Pi/CoPi) has to click on Prospective Students and click on the “REVIEW FACULTY REGISTRATIONS” tab.

2. Admin(Pi/CoPi) should find the faculty registration requests which is similar to the student’s application requests page

**Post-conditions:**

1. Admin(Pi/CoPi) should be able to Approve, Reject, Undo or Delete the faculty registration request.

**Alternative Courses of Action:**

N/A

**Exceptions:**

N/A

**Related Uses Case:**

N/A

User Story #1172

# Add Guideline Videos

Instructor: Masoud Sadjadi

# User Story - Add Guideline Videos

* As a user, I would like to watch videos available in the login page so that it would be easy to use the VIP website.

## Use Case

Use Case ID: (VIP-1172)

Details: Make guidelines videos appear on the login page and arrange them according to the current login design.

Actor: Faculty, student

Pre-Condition:

* Faculty and student has navigated to the login page.

Description:

1. Use case begins when a faculty member click on the User Guide for Faculty video
2. Use case begins when a student click on the User Guide for Student video

Post-Conditions:

1. Users should be able to navigate and use the VIP website without constraints.

Alternative Course of Action

N/A

Exceptions:

N/A

Related Use Cases:

N/A

Decision Support:

Frequency: Low: User might only use videos once in order to use the VIP website

Criticality: Low: This feature has no impact on the rest of the sites performance

Risk: Low.

Constraints:

1. Usability:
   1. Requires no experience.
2. Reliability:
   1. Availability – Videos are always available on login page.
3. Performance:
   1. Videos will play after 5 sec of been clicked.
4. Supportability:
   1. All modern web browsers should be supported.

Modification History:

Owner: Franco Hernandez

Initiation date: 10/19/2016

Date last modified: 10/23/2016

User Story #1158

# Search name in admin panel

User Story - Search name in admin panel

● As a user, I would like to search the students in the list of Users available in the admin panel so that it would be feasible for quick navigation.

## Use Case

**Use Case ID:** (VIP-1158)

**Details:** Search name in admin panel

**Actor:** Pi/CoPi

**Pre-conditions:**

1. Pi/CoPi has navigated to the VIP website.

2. Pi/CoPi is able to login and click on admin panel.

3. Pi/CoPi is navigated to the admin panel page

**Description:**

1. Pi/CoPi should be able to use the search box in order to search for the students in the list.

2. If Pi/CoPi enters any student’s name in the search box then only their name should be visible in the list

**Post-conditions:**

1. Pi/CoPi should be able to click on the name of the student whom he just searched for the detail information

**Alternative Courses of Action:**

N/A

**Exceptions:**

N/A

**Related Uses Case:**

N/A

User Story #1167

# Updating Emailing Services

# User Story - Updating Email Services

* As a developer I need to make sure that the VIP emailing services are updated for this semester

## Use Case

**Use Case ID:** (VIP-1167)

**Details:** Updating the existing email handler code for the current semester

**Actor:** Developers

**Pre-Condition:**

* Updating developer is a current member of the VIP Project

**Description:**

1. Use case begins when a new semester starts with a new VIP team
2. A member of the team, henceforth referred to as a developer must access the current code, and modify any mailing service lists.
3. Once updated the new list is passed onto the Integration Use case, to be eventually updated on the Dev, and then Live servers of the site.

**Post-Conditions:**

1. Last semester’s developers will no longer receive emails from the VIP site
2. This semester's developer will now begin to receive emails from the VIP site.

**Alternative Course of Action**

N/A

**Exceptions:**

N/A

**Related Use Cases:**

N/A

**Decision Support:**

**Frequency:** Low: This occurs once a semester.

**Criticality:** Low: This feature has no impact on the rest of the sites performance

**Risk:** Low.

**Constraints:**

1. **Usability:** 
   1. Requires some javascript experience
2. **Reliability:** 
   1. Availability – Once a semester.
3. **Performance:** 
   1. Emails are sent within 5 minutes of code initiation
4. **Supportability:**
   1. All modern web browsers should be supported.

**Modification History:**

**Owner:** Joseph Gonzalez

**Initiation date:** 10/4/2016

**Date last modified:** 10/4/2016

### 

### 

### Sprint 4

User Story #1143

# Adding Excel report in Admin Panel.

**User Story**

As an admin, I would like to have an excel icon in the admin panel. When the admin wants list of students, faculty information he can simply click on the excel icon to generate it.

## USECASE:

## Use Case ID: (VIP-1143)

**Actor: Pi/coPi**

**Pre-Condition:**

We need to consider that the admin is in the admin panel.

Admin selects the user type and other fields in the filters and clicks on “LookUp”.

A list of students or the respected chosen details are generated

**Description:**

The use case will begin when the admin clicks on the excel report icon.

**Post-Conditions:**

Admin must be able to generate the excel report when clicked on the icon.

The report will contain all the information that the admin wants.

Alternative Course of Action:

N/A

Exceptions:

N/A

Related Use Cases:

N/A

User Story #1192

# Enable Pi to upload the documents

# User Story - Enable Pi to upload the documents

* As a PI user I would like to be able to have access to create and edit reports and their correspondent files from the dropdown menu on the VIP header panel. Also I would like to see the results of these changes on the same dropdown menu.

## Use Case

**Use Case ID:** (VIP-1192)

Details: Add a page to the VIP website where PI can name, edit and delete reports, report files and files links for the Google Drive. Add a link to have access to this page on the VIP header panel on the report section.

**Actor:** PI

**Pre-Condition:**

* PI has to reach the VIP site.
* PI has to login.
* PI has to click on the Report section of the VIP header panel.

**Description:**

1. Use case begins when PI click the Edit/Create link of the report section on the VIP header panel.
2. PI access the Create/Edit form and create or edit data related to the reports, report files and file links.

**Post-Conditions:**

1. Latest addition, edit or deletion of reports names, report files and file links will appear on the dropdown menu of the report section below Create/Edit link.

Alternative Course of Action

N/A

Exceptions:

N/A

Related Use Cases:

N/A

**Decision Support:**

Frequency: Low: IP User might use this feature from 1 to 3 times per semester.

Criticality: Low: This feature has no impact on the rest of the sites performance

Risk: Low.

**Constraints:**

1. Usability:
   1. Requires no experience.
2. Reliability:
   1. Availability – Create/Edit link, Create/Edit form, latest report names and their files are always available.
3. Performance:
   1. New additions and changes will be visible on the report dropdown menu after

5 sec of been submitted.

1. Supportability:
   1. All modern web browsers should be supported.

Modification History:

Owner: Franco Hernandez

Initiation date: 11/9/2016

Date last modified: 11/11/2016

User Story #1175

# Modify mandatory fields to all 3 forms to the users & add honors college

User Story – Modify mandatory fields to all 3 forms to the users & add honors college

As a user, I would like to see the required fields before filling the forms and also see the “Honors College” in the list of colleges in the form so that I would not forget to give the required information and select the correct college respectively.

.

## Use Case

**Use Case ID:** (VIP-1175)

**Actor:** Student, Staff/Faculty

**Pre-conditions:**

1. “Apply for a Project” Form should open without any errors.
2. “Propose a Project” Form should open without any errors.
3. “Faculty Registration” Form should open without any errors.

**Description:**

1. User should find “Honor College” in the drop down menu of college in all the three forms.
2. User should fill all the required fields that are marked with Star \* before submitting either of the three forms.

**Post-conditions:**

1. User should be able to submit the forms once the required field has been filled properly.

**Alternative Courses of Action:**

N/A

**Exceptions:**

N/A

User Story #1159

# Maintaining and Updating the Dev and Live Server

# User Story - Maintaining/Updating the Live and Dev Servers

* As a developer I need to be able to test the site on the dev server, and then push to the live server once the code is ready.

## Use Case

**Use Case ID:** (VIP-1159)

**Details:** Maintaining/Updating the Live and Dev Servers

**Actor:** Developers

**Pre-Condition:**

* Developer has been given access rights to the vip server
* Developer has logged into the live server via Putty and WinSCP

**Description:**

1. Use Case begins when developer logs into the VIP dev server(vip-dev.cs.fiu.edu) via Putty and WinSCP
2. From WinSCP, developer navigates to the directory where they wish to run the dev server (/var/www/VIP4)
3. Developer must modify the following files in their code to run on the dev server
   1. Server.js
   2. Api/config/config
   3. api/config/auth.js

In all 3 of these files, the Developer must modify the lines referring to localhost or

vip.fiu.edu and replace them with the dev servers porting information (explained in

detail below)

1. Developer then make a zip file of the modified code
2. Developer then drags and drops the .zip file containing the dev\_code from their system to the vip server
3. From putty, the developer now navigates to the same directory, and unpackages the zip file.
4. The developer navigates into Code file they’ve unpackaged and runs “forever start server.js”
5. Now the dev server should be running on vip-dev.cis.fiu.edu:8001
6. From here the developer now tests the site to ensure no bugs and errors. Once all is well, they must undo the changes made in step 3, and repeat steps 4-6.
7. Once ready to go live, disable the existing live server by running “forever stopall”
8. Finally, repeat step 7, and verify the server is live at vip.fiu.edu

**Post-Conditions:**

1. The Live server should now be running on the latest version of the code

**Alternative Course of Action**

* If updating the live server, follow same directions, but the live SSH into vip.fiu.edu for live server

**Exceptions:**

N/A

**Related Use Cases:**

Updating the Dev Server

**Decision Support:**

**Frequency:** Low: This occurs only a few times a semester.

**Criticality:** High: This task affects the live running version of the site, and involves testing for critical functionality of the site

**Risk:** High.

**Constraints:**

1. **Usability:** 
   1. Requires experience with linux server commands
2. **Reliability:** 
   1. Availability – A few times a semester.
3. **Performance:** 
   1. Site must be running stable on the live server
4. **Supportability:**
   1. All modern web browsers should be supported.

**Modification History**

User Story #1101

# Creating new Lock & Unlock Mechanism for Projects

# User Story - Creating Lock & Unlocking Mechanism to Projects

* As a User I would like to be able to lock projects so that they can no longer be joined

## Use Case

**Use Case ID:** (VIP-1101)

**Details:** Create Lock and Unlocking Mechanisms to Projects

**Actor:** Users

**Pre-Condition:**

* A Project has been created and approved on vip.fiu.edu

**Description:**

1. Use Case begins when a CoPI navigates to the admin panel
2. From the admin panel, CoPI navigates to the “Lock/Unlock Projects” dropdown menu
3. From the top dropdown menu they can select a project
4. From the bottom dropdown menu they can select “Active” or “Inactive”
5. Once both items are selected, they can press the green “Change” button to change the status of the project

**Post-Conditions:**

1. If the status was set to “Inactive” users will no longer be able to join the project. When pressing either the join button in the project’s page, or when trying to join the project via the student application
2. If the project was set to “Active” users will now be able to join the project.

**Alternative Course of Action**

N/A

**Exceptions:**

N/A

**Related Use Cases:**

**Decision Support:**

**Frequency:** Medium: This occurs at the request of the product owner

**Criticality:** Medium: These locks prevent projects from being joinable by new members

**Risk:** Low.

**Constraints:**

1. **Usability:** 
   1. No training required
2. **Reliability:** 
   1. Availability – A few times a semester.
3. **Performance:** 
   1. Locks access and update the database within a reasonable time frame
4. **Supportability:**
   1. All modern web browsers should be supported.

**Modification History:**

**Owner:** Joseph Gonzalez

**Initiation date:** 10/17/2016

**Date last modified:** 10/24/2016

*Florida International University*

*School of Computing and Information Sciences*

CEN 5011- Advanced Software Engineering

Feature Document

User Story #1186

# Search name in admin panel

**Team Member:**

Ravi Kiran Agarthi

**Product Owner(s)**:

Masoud Sadjadi

**Mentor(s)**:

Mohsen Taheri

**Instructor**: Masoud Sadjadi

# 

# 

# User Story - Save semester for both project proposal and application for a project

● As a user, I would like to save on which semester does Professor proposes the project and on which semester student applies for a project so that I could check information based on the semester.

## Use Case

**Use Case ID:** (VIP-1186)

**Details:** Save semester details

**Actor:** Staff/Faculty and Student

**Pre-conditions:**

1. Staff/Faculty and Student has navigated to the VIP website.

2. Staff/Faculty and Student is able to login and click on Project Proposal form and Apply for a Project form respectively.

**Description:**

1. Both Staff/Faculty and Student should be able to find and use semester drop down menu in the respective forms.

2. Both Staff/Faculty and Student should be able to submit the form without any errors

**Post-conditions:**

1. Both Staff/Faculty and Student semester details that they have filled on the forms should be saved in the database

**Alternative Courses of Action:**

N/A

**Exceptions:**

N/A

**Related Uses Case:**

N/A

### Sprint 5

*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

Feature Document

User Story #1193

# Extend Lock and Unlock Functionality for semester locks

# 

**Team Member:**

Joseph Gonzalez

**Product Owner(s)**:

Masoud Sadjadi

**Mentor(s)**:

Mohsen Taheri

**Instructor**: Masoud Sadjadi

# User Story - Extending Lock Services to semester locks

* As a user, I would like to lock an entire semester, so that all the projects in that semester are also locked, and so projects cannot be proposed for that semester

## Use Case

**Use Case ID:** (VIP-1193)

**Details:** Extending Lock Services to a Semester Lock

**Actor:** Users

**Pre-Condition:**

* Projects have been created and assigned a semester
* Terms have been created in the terms collection

**Description:**

1. Use Case Begins when a Pi/CoPI navigates to the admin panel (vip.fiu.edu/#/admin-panel)
2. Pi/CoPI navigates to the “Lock/Unlock Semester” Panel
3. In the drop down menu labeled “Term” Select a semester
4. In the second drop down menu labeled “change Semester's status to” the desired status
5. Once both menus have options filled out, press the button labeled “Change”
6. A pop up message will appear indicating a change has been made

**Post-Conditions:**

1. The term’s status has been updated
2. Any projects from that term are also locked

**Alternative Course of Action**

N/A

**Exceptions:**

N/A

**Related Use Cases:**

Lock/Unlock a Project

**Decision Support:**

**Frequency:** Low: This occurs only a few times a semester.

**Criticality:** High: This affects which projects are available to users and which semesters projects can be proposed for

**Risk:** High.

**Constraints:**

1. **Usability:** 
   1. Requires no coding experience
2. **Reliability:** 
   1. Availability – At Users will
3. **Performance:** 
   1. Completes modifications to the database within 5 seconds, which is deemed acceptable
4. **Supportability:**
   1. All modern web browsers should be supported.

**Modification History:**

**Owner:** Joseph Gonzalez

**Initiation date:** 10/15/2016

**Date last modified:** 11/1/2016

### Sprint 6

For Sprint 6, we worked on various minor quality of life upgrades, in that regard, while plentiful, none warranted a separate use case

### Sprint 7

**.**

**.**

**.**

# 

# 

# 

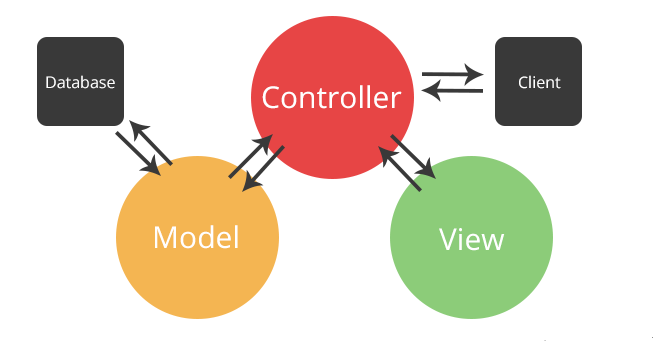
# 

# System Design

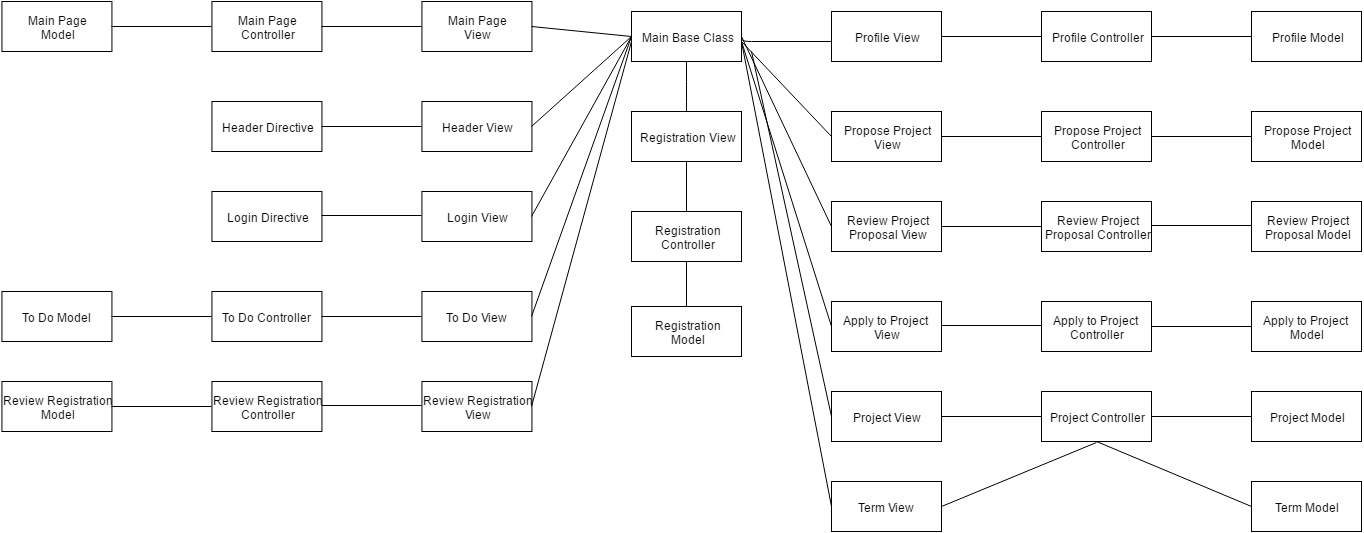
This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

## Architectural Patterns

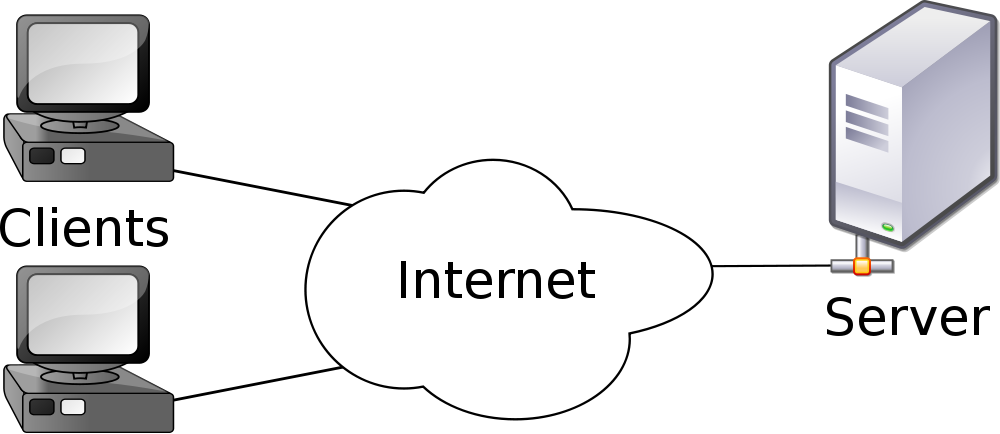
The site runs on a Model-Control-View or MVC Architecture. As shown in the picture below



## System and Subsystem Decomposition



## Deployment Diagram

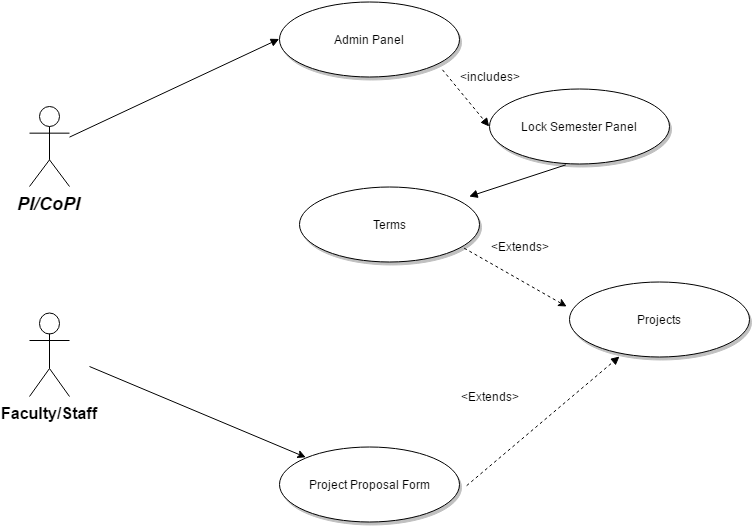


## 

## 

## 

## Design Patterns



# 

# 

# 

# 

# 

# 

# System Validation

**Unit & Integration Tests:**

**Test Case 1 (Sunny Day)**

**ID: VIP-SD-1152-01**

**Purpose**

* Ensure the “Mingle Link” section of the “Project Proposal Form” saves the URL correctly

**Precondition**

* User fills out the Mingle Link and required fields in the Project Proposal form.

**Input**

* User Clicks on the “Submit” button at the bottom of the page.

**Expected Result**

* Mingle Link is included in the Project Details entry

**Actual Result**

* Same as expected.

**Test Case 2 (Sunny Day)**

**ID: VIP-SD-1152-02**

**Purpose**

* Ensure Mingle Social Link Icon appears when a Mingle Link is tied to a Project.

**Precondition**

* User is the Product Owner or Project Member of the Project.
* Mingle Link section of the Project Proposal Form was filled out

**Input**

* User Navigates to the “Project Details Page”.

**Expected Result**

* The Mingle Social Icon appears under the ‘Discover More” section of the “Project Details Page”

**Actual Result**

* Same as expected.

**Test Case 3 (Sunny Day)**

**ID: VIP-SD-1152-03**

**Purpose**

* Ensure the Mingle Social Icon redirects users to their Mingle Login Page

**Precondition**

* User is the Product Owner, or Project Member of the Project
* Mingle Link section of the “Project Proposal Form” was filled out.

**Input**

* User Navigates to the “Project Details Page”
* User clicks the “M” Icon for Mingle under the “Discover More” section of the page

**Expected Result**

* User is redirected to the “Mingle Login Page”

**Actual Result**

* Same as expected.

**Test Case 1 (Sunny Day)**

**ID: VIP-SD-1146-01**

**Purpose**

* Ensure the checkbox are able to be checked and unchecked

Precondition

* User fills out the login information

Input

* User Clicks on the on the checkbox.

Expected Result

* User stay login for a period of 2 weeks

Actual Result

* Same as expected.

**Unit & Integration Tests:**

## ● Test Case 1(Sunny Day) ○ Purpose: Test that the format of project description stays the same as professor entered in the proposal form in the Projects page and also in the project detail information.

## ○ Precondition:Faculty has reached the VIP site and reached project Proposal form.

## ○ Input: Faculty has to enter description in the form.

## ○ Expected Result: The format of the description should stay the same in the Projects page and stays the same even after clicking on a particular project. ○ Actual Result: The format of the description stays the same in the Projects page and stays the same even after clicking on a particular project. ○ Status: SUCCESS

## ● Test Case 2(Sunny Day) ○ Purpose: Test that the toolbar in the project proposal form is working ○ Precondition:Faculty has reached the VIP site and reached project Proposal form.

## ○ Input: Faculty has to enter description in the form.

## ○ Expected Result: The description format can be changed with the help of toolbar options and can be saved ○ Actual Result: The description format is getting changed with the help of toolbar options and getting saved smoothly with no errors ○ Status: SUCCESS

## ● Test Case 1(Rainy Day) ○ Purpose: Test that the format of project description stays the same as professor entered in the proposal form in the Projects page and also in the project detail information.

## ○ Precondition: Faculty has reached the VIP site and reached project Proposal form.

## ○ Input: Faculty didn’t enter the description and submitted the form

## ○ Expected Result: Popup message showing to enter the required fields. ○ Actual Result: sPopup message showing to enter the required fields. ○ Status: SUCCESS

**nit & Integration Tests:**

**Test Case 1 (Sunny Day)**

**Purpose:**

* When student/teacher browse through the website and wish to know which are the recent projects and oldest projects.
* As admin or a teacher, I would to like to know the oldest projects so that I might make modifications or else delete them. So sorting the projects by proposed date is helpful.

**Precondition**

* Developer clicks on the filter option and selects the option proposed date.

**Input**

* Developer selects the either proposed date ascending that would be oldest to new or descending new to old

**Expected Result**

* User gets the projects sorted in the format of either old to new or new to old.

**Actual Result**

* Same as expected.

**Test Case 2 (Sunny Day)**

**ID: VIP-SD-717-02**

**Purpose**

* We do add multiple projects on the same day. In such scenarios the time has to be considered for sorting the projects.

**Precondition**

* Developer would have to make sure that the sorting is happening respectively.

**Input**

* Developer clicks on the same field again..

**Expected Result**

* Developer should be getting the projects sorted similar to the initial test case.

**Actual Result**

* Same as expected.

**Unit & Integration Tests**

**Test Case 1 (Sunny Day)**

**Purpose**

* When student enters skill item in the application form for a project then it should reflect in his profile.

**Precondition**

* Developer must log in to the website as a student and can apply for a project, encountered no errors while doing so.

**Input**

* Developer enters skill item and clicks on the “Submit” button at the bottom of the page.

**Expected Result**

* Data added to the user profile and can be viewed by the professor.

**Actual Result**

* Same as expected.

**Test Case 2 (Sunny Day)**

**Purpose**

* While proposing any project, professor should be able to mention required skill item for his project.

**Precondition**

* Developer has to log in to the website as professor and encountered no errors while doing so.

**Input**

* Developer enters required skill item and clicks on the “Submit” button at the bottom of the page.

**Expected Result**

Data will go to the Database.

**Actual Result**

* Same as expected.

**Test Case 3 (Rainy Day)**

**Purpose**

* Ensure project application must contain valid data for all required fields (skill item)

**Precondition**

* Student has reached the VIP site and reached apply for project application form/ Propose a project form.

**Input**

* Student leaves a required field blank or inputs invalid data

**Expected Result**

* Submit button becomes disabled

**Actual Result**

* Submit button becomes disabled

**Unit & Integration Tests:**

* **Test Case 1(Sunny Day)**

**Purpose**: To check if admin is able to generate the excel report

## Precondition: Admin is in the admin panel

## Input: Admin selects the respective fields or uses the search box

## and clicks on the excel icon

## Expected Result: A report must be generated automatically with the

## name current and Report.xlsx all the fields and details of the

## requested user type

## Actual Result: The admin must be able to get an excel report with

## current date and all

## the details

## Status: SUCCESS

## 

## Test Case 2(Rainy Day) Purpose: To check if the admin is able to generate an excel sheet.

## Precondition: Admin is in the admin panel.

## Input:Admin selects the respective fields and clicks on the Excel Icon.

## Expected Result: Admin would not be able to generate the report or

## the details.

## Actual Result: An excel report with details will not be generated

## when admin clicks on excel icon

## Status: SUCCESS

Test Case 1 (Sunny Day)

ID: VIP-SD-1192-01

Purpose:

* Test that guideline videos appear on the login page.

Precondition

* User navigates to the login page.

Input

* N/A

Expected Result

* Videos must be visual on login page.

Actual Result

* Same as expected.

Test Case 2 (Sunny Day)

ID: VIP-SD-1172-02

Purpose

* Test videos are functional.

Precondition

* Videos are available on login page.

Input

* User click on video.

Expected Result

* Videos plays without interruptions.

Actual Result

* Same as expected.

**Unit & Integration Tests:**

**● Test Case 1(Sunny Day)**

## ○ Purpose: Test that the three form “Faculty registration form”, “Apply for a project” and “Proposing a project” is showing the required fields with a star mark.

## ○ Precondition: User has reached the VIP site and reached the respective form.

## ○ Input: Details are filled by the users.

## ○ Expected Result: User can be able to submit the forms without errors.

○ Actual Result: User is submitting the forms without errors  
○ Status: SUCCESS

**● Test Case 2(Sunny Day)**

## ○ Purpose: Test whether user is able to submit the form by selecting Honors college without selecting any school.

## ○ Precondition: User has reached the respective form.

## ○ Input: User is selecting college as “Honor College” without any school

## ○ Expected Result: User can be able to submit the forms without errors.

○ Actual Result: User is submitting the forms without errors  
○ Status: SUCCESS

**Test Case 1 (Sunny Day)**

**ID: VIP-SD-1159-01**

**Purpose**

* Test that the dev server is running at the same time as the live server

**Precondition**

* Live server is currently running

**Input**

* The Dev Server is turned on and running

**Expected Result**

* The features run as intended

**Actual Result**

* Same as expected.

**Test Case 2 (Rainy Day)**

**ID: VIP-RD-1159-01**

**Purpose**

* Confirm the Dev code on the dev server is running

**Precondition**

* Dev server code has been updated

**Input**

* The Dev Server is turned on and running, and the new features are being tested

**Expected Result**

* The server crashes due to an error

**Actual Result**

* The server is able to run the new code without crashes or errors

**Test Case 3 (Sunny Day)**

**ID: VIP-SD-1159-03**

**Purpose**

* Test that the live server is now running with code updates

**Precondition**

* Live server code has been updated

**Input**

* The Live server is turned on with the new code

**Expected Result**

* The site runs as intended

**Actual Result**

* Same as expected.

**Unit & Integration Tests:**

**Test Case 1 (Sunny Day)**

**ID: VIP-SD-1101-01**

**Purpose**

* Test that the lock is now working

**Precondition**

* A project has been selected in the admin panel

**Input**

* The projects status is set to inactive

**Expected Result**

* The Database now reflects the changes made in the admin panel

**Actual Result**

* Same as expected.

**Test Case 2 (Sunny Day)**

**ID: VIP-SD-1159-02**

**Purpose**

* Confirm an inactive project cannot be joined

**Precondition**

* Project’s status has been set to inactive

**Input**

* A user attempts to join a project via the project details page or project application form

**Expected Result**

* The site blocks them from joining the project and alerts them of the reason

**Actual Result**

* Same as expected

**Unit & Integration Tests:**

## ● Test Case 1(Sunny Day) ○ Purpose: Test that when Professor proposes a project, the semester details are getting saved in database

## ○ Precondition: Staff/Faculty has reached the VIP site and reached the Project Proposal Form.

## ○ Input: Staff/Faculty selects a semester from drop down menu and submits the form after entering all the required fields.

## ○ Expected Result: The semester details should be saved in the database. ○ Actual Result: The semester details are getting saved in the database. ○ Status: SUCCESS

## 

## ● Test Case 2(Sunny Day) ○ Purpose: Test that when Student Applies for a project, the semester details are getting saved in database

## ○ Precondition: Student has reached the VIP site and reached the Apply to a Project Form.

## ○ Input: Student selects a semester from drop down menu and submits the form after entering all the required fields.

## ○ Expected Result: The semester details should be saved in the database. ○ Actual Result: The semester details are getting saved in the database. ○ Status: SUCCESS

**Unit & Integration Tests:**

**Test Case 1 (Sunny Day)**

**ID: VIP-SD-1193-01**

**Purpose**

* Test that the lock is changing the status of the term collection

**Precondition**

* The Lock/Unlock Semester panel has been filled out

**Input**

* The status of the term has been changed

**Expected Result**

* The status of the term in the terms collection now matches the option selected

**Actual Result**

* Same as expected.

**Test Case 2 (Sunny Day)**

**ID: VIP-SD-1193-02**

**Purpose**

* Confirm that a Project cannot be proposed in a locked semester

**Precondition**

* A term has been locked
* The same term is selected in the “Semester” section of the “Project Proposal Form”

**Input**

* The user attempts to submit the project under the locked term

**Expected Result**

* An error message indicates that the semester is locked and the project cannot be proposed for that project

**Actual Result**

* Same as expected

**Test Case 3 (Sunny Day)**

**ID: VIP-SD-1193-03**

**Purpose**

* Test that projects from a now locked semester are also locked

**Precondition**

* The projects term has been locked

**Input**

* A user attempts to join a project from a locked term

**Expected Result**

* They are given an error message preventing them from joining the project

**Actual Result**

* Same as expected.

# 

# 

# 

# 

# Glossary

Guest **-** Lowest level user on the site can only view the homepage, and projects

Student - Entry level user, can join projects

Faculty/Staff - Mid Level user, can create projects, and accept students into projects

Pi/CoPi - Top level user, full access to all of sites features, serves as administrator to the site.

Projects - Created by Faculty/Staff and Pi/CoPi team undertakings that are proposed on the site

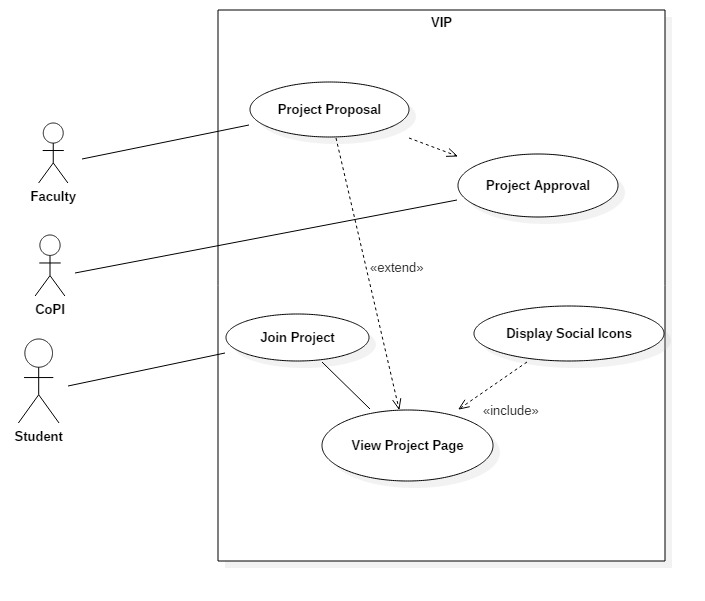
Terms - Semesters that are accessible on the site, used to archive projects

Skill item - necessary knowledge set on the system that can be stated in a project details to match students knowledge base

# Appendix

## Appendix A - UML Diagrams

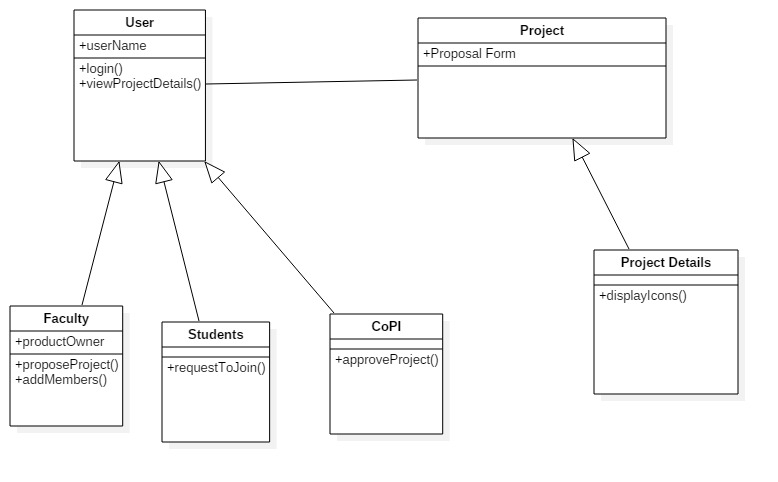
## Use Case Diagram

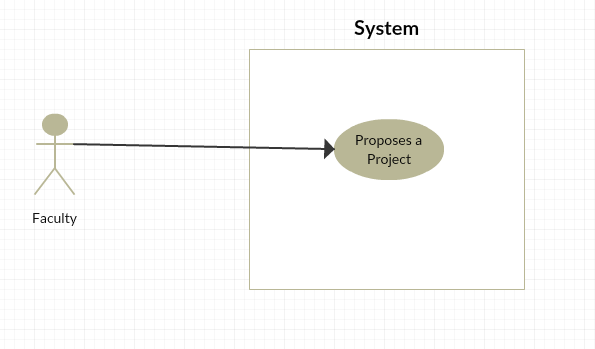


**Sequence Diagram**



**Class Diagram**

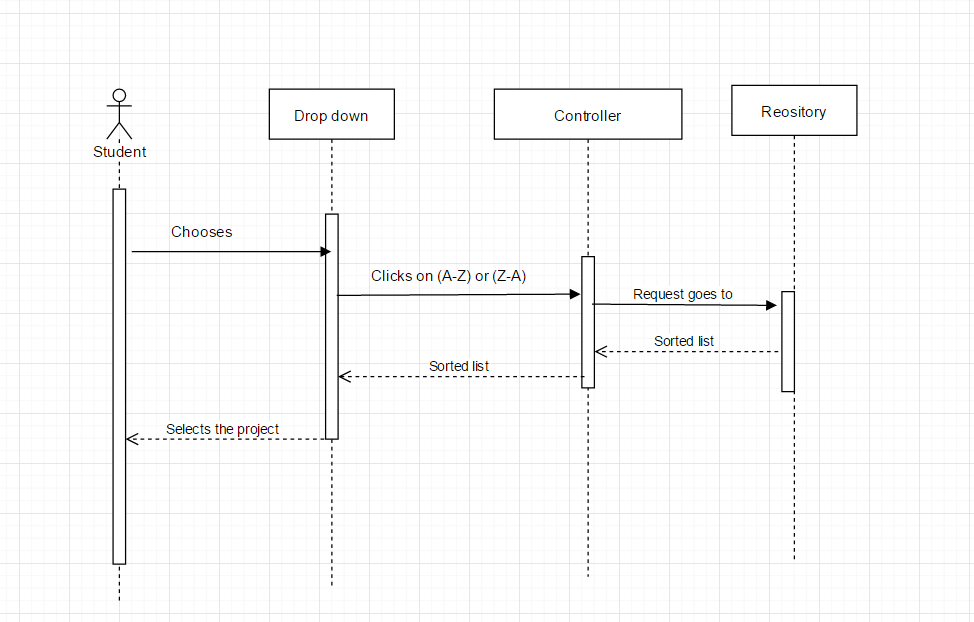




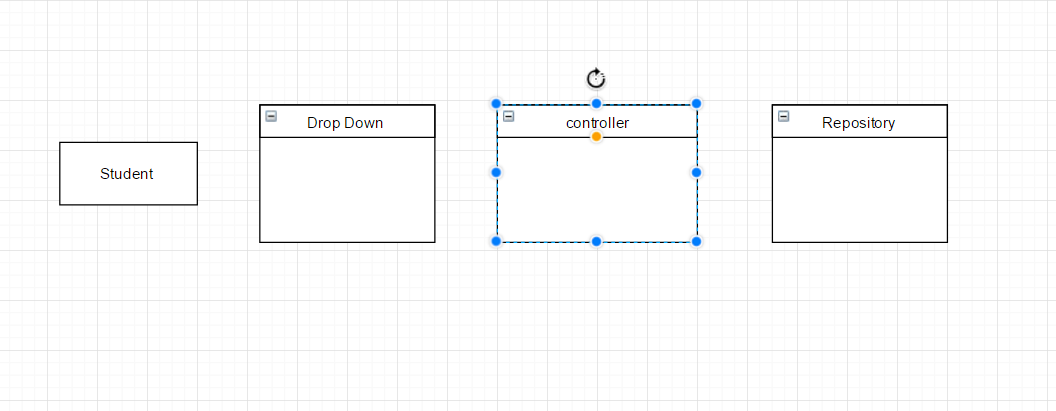
**se Case Diagram**



**Sequence Diagram**

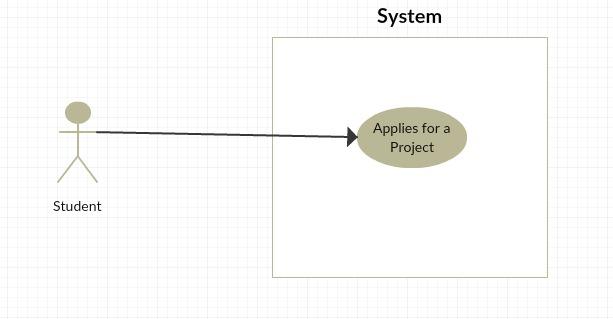


**Class Diagram**

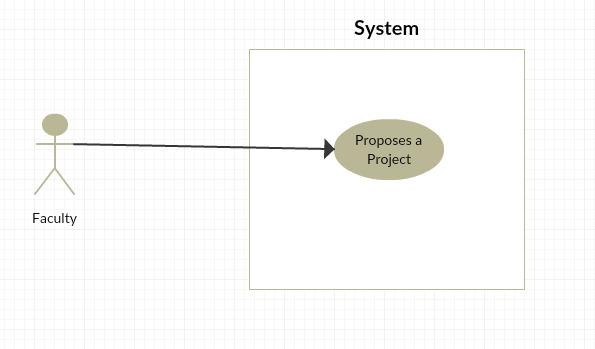


**Use Case Diagram**

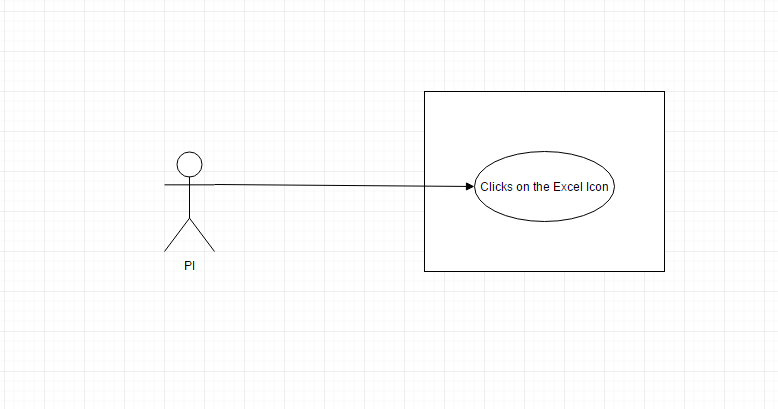
* When student applies for a project.



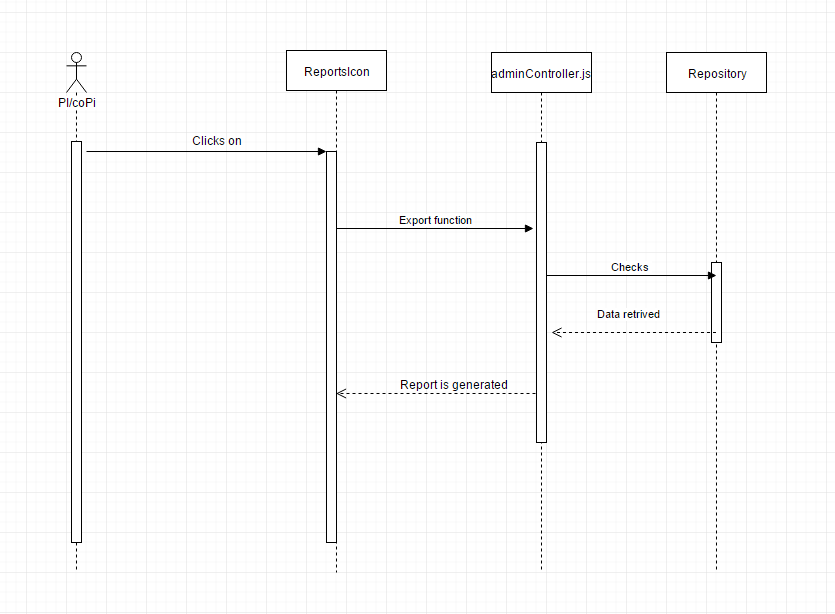
* When professor proposes a project



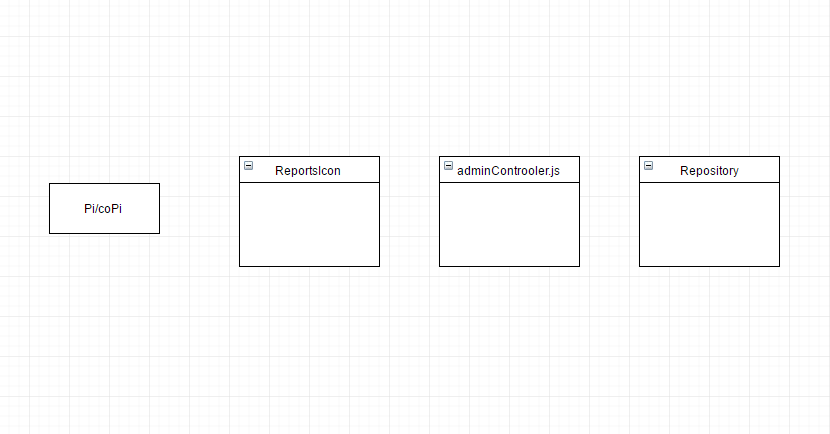
**Use case Diagrams:**



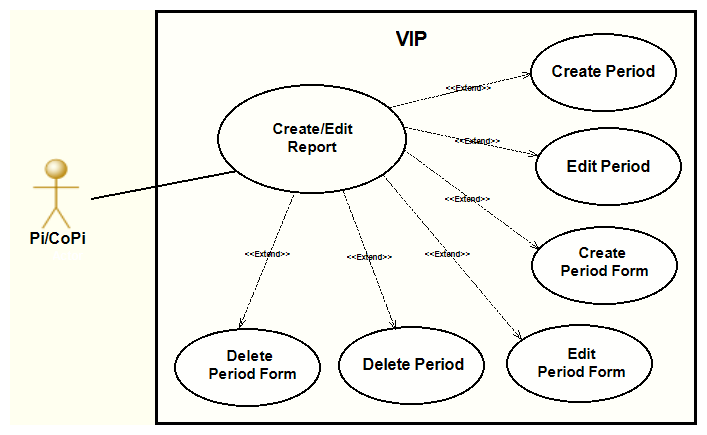
**Sequence Diagram:**



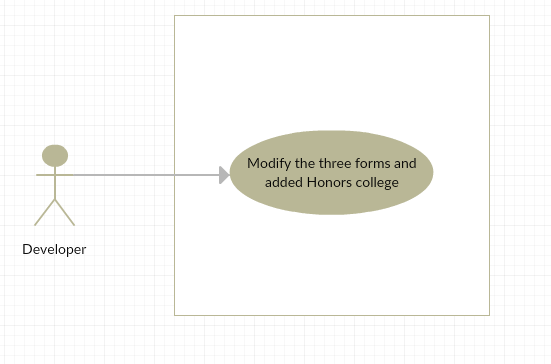
**Class Diagram:**



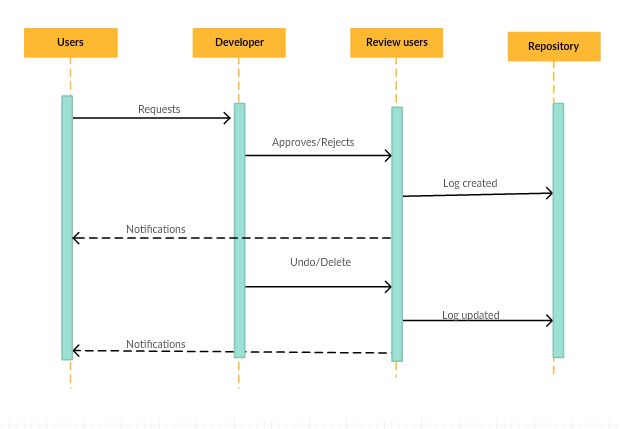
Use Case Diagram



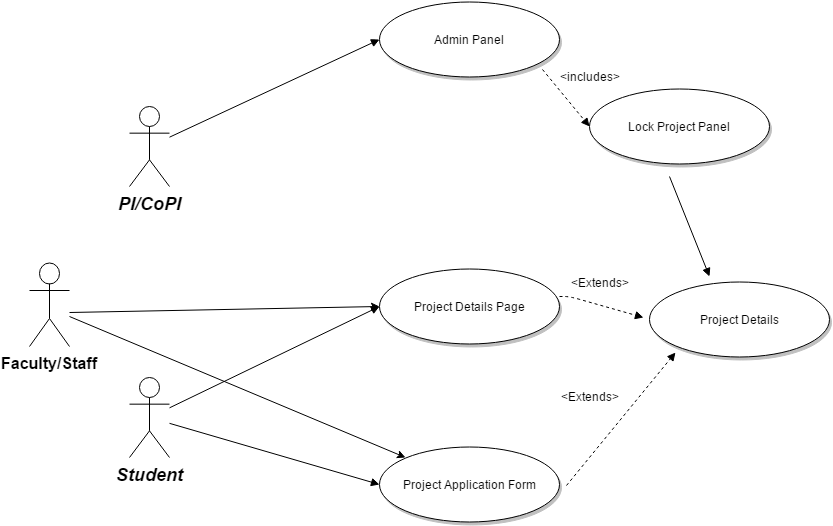
**Use Case Diagram**



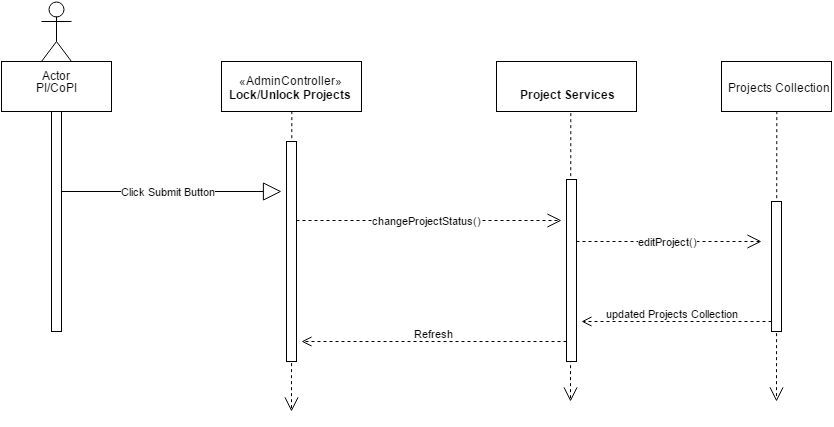
**Sequence Diagram**



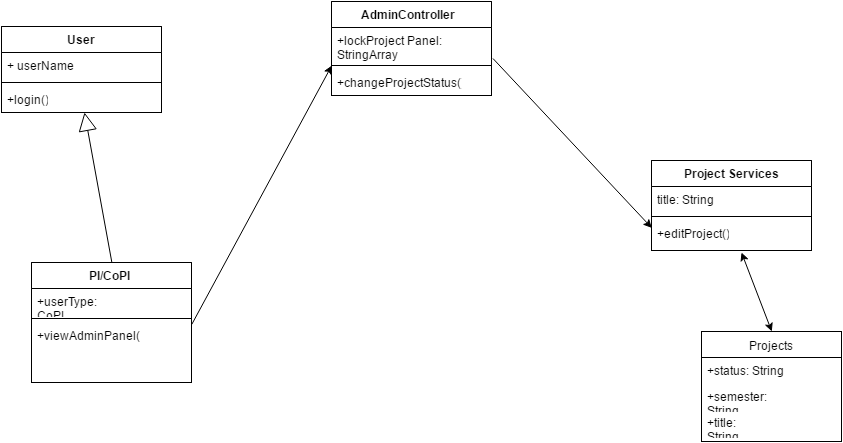
**Use Case Diagram**



**Sequence Diagram**

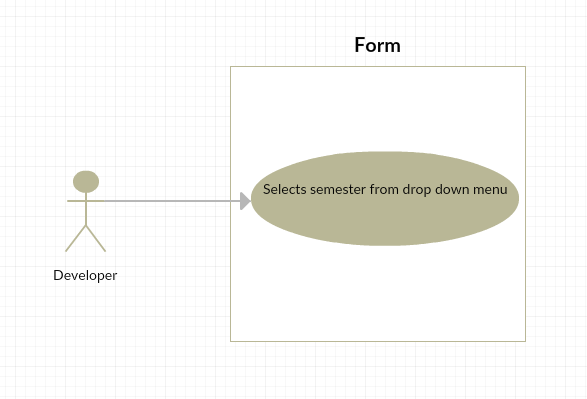


**Class Diagram**

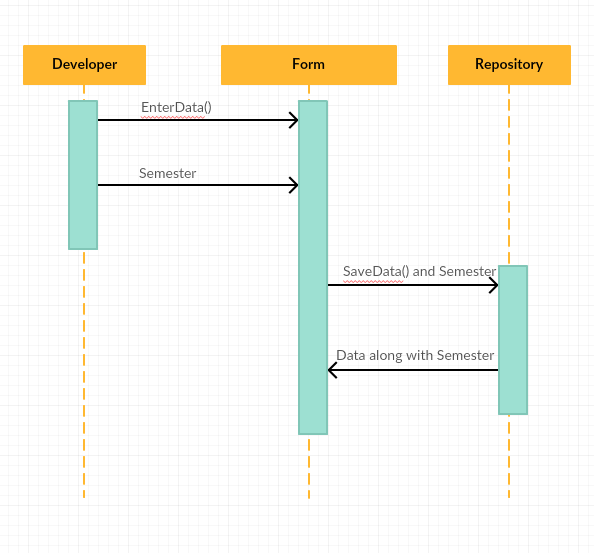


**Related Uses Case:**

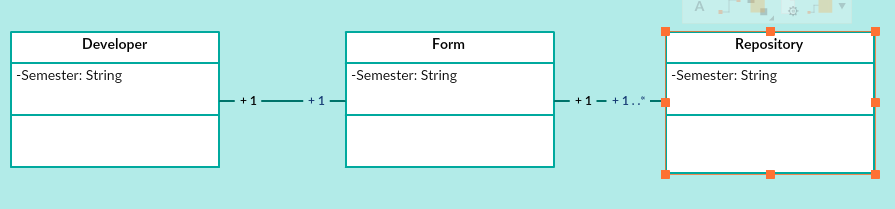
N/A



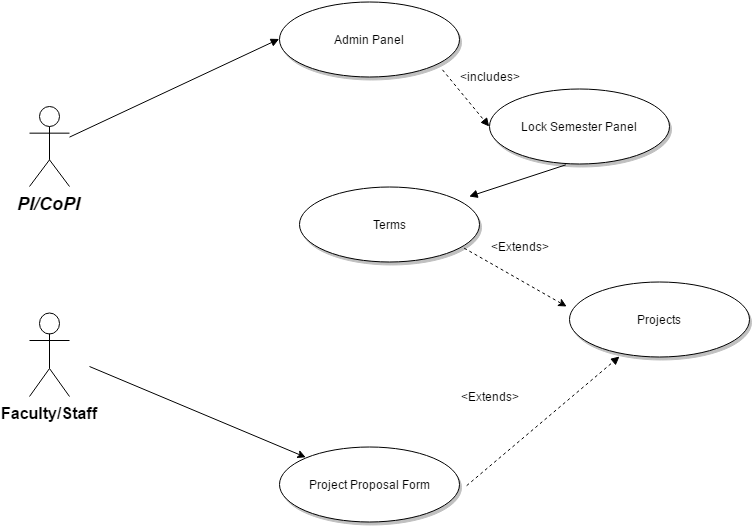
**Sequence Diagram**



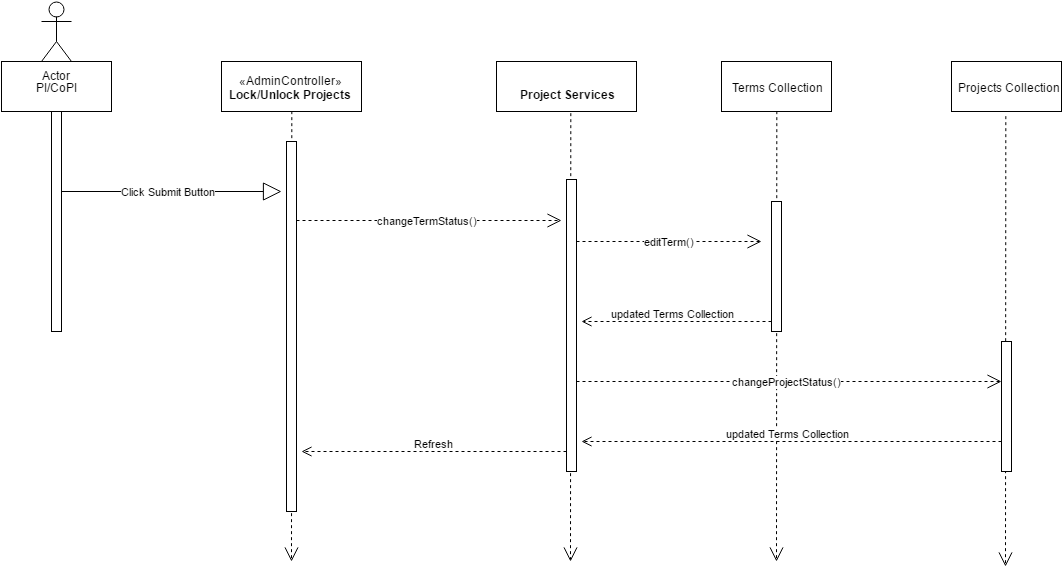
**Class Diagram**



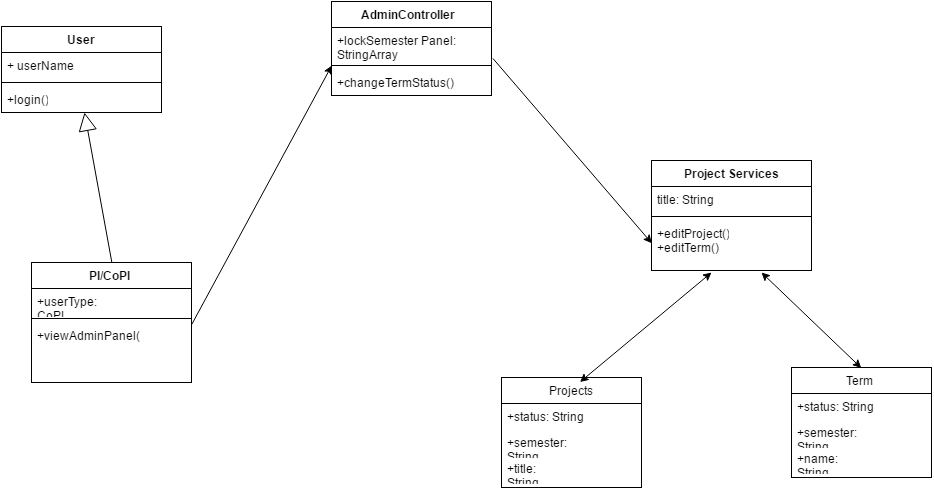
**Use Case Diagram**



**Sequence Diagram**



**Class Diagram**



## Appendix B - User Interface Design



## 

## 

## 

## 

## 

## 

## Appendix C - Sprint Review Reports

**Sprint 2 Review Meeting Minutes**

Attendees: Joseph Gonzalez, Ravi Kiran, Shefali Nakum, Nandini Varma,

Start Time: 6:30

End Time: 7:23

After a show and tell presentation, the implementation of the following user stories were accepted by the Product owners:

* Mingle Filed and Logo #1152
* Add format to Project Proposal Form #1142
* Add Skill item in student application form and faculty proposal form #1144

# 

The Following ones ere rejected and moved to the product backlog to be assigned to a future sprint at a future planning meeting.

* Making sure All Users are Using the Right Code for Implementing #1153
* Use VIP Team Filters #849

**Sprint 3 Review Meeting Minutes**

Attendees: Joseph Gonzalez, Ravi Kiran, Shefali Nakum, Nandini Varma, Franco Hernandez

Start Time: 10pm

End Time: 10:57pm

After a show and tell presentation, the implementation of the following user stories were accepted by the Product owners:

* VIP Version Control #1153
* Updating Email Services #1167
* Add Faculties Registration Page to the Website #1140
* Add Text Box Whispers to Social Icon Links #1160
* Search Name in Admin Panel #1158
* Use VIP Teams Filter #849

The Following ones were rejected and moved to the product backlog to be assigned to a future sprint at a future planning meeting.

* Taking Care of Dev Server #1159
* Add Excel Report in Admin Panel #1143
* Make Login Permanent on the System #1146

**Sprint 4 Review Meeting Minutes**

Attendees: Joseph Gonzalez, Ravi Kiran, Shefali Nakum, Nandini Varma, Franco Hernandez

Start Time: 10pm

End Time: 10:57pm

After a show and tell presentation, Product owner accepted all of the assigned user stories except Make login permanent #1146

**Sprint 5 Review Meeting Minutes**

Attendees: Joseph Gonzalez, Ravi Kiran, Shefali Nakum, Nandini Varma, Franco Hernandez

Start Time: 7pm

End Time: 7:15pm

After a show and tell presentation, the implementation of the following user stories were accepted by the Product owners: All of them

**Sprint 6 Review Meeting Minutes**

Attendees: Joseph Gonzalez, Ravi Kiran, Shefali Nakum, Nandini Varma, Franco Hernandez

Start Time: 6pm

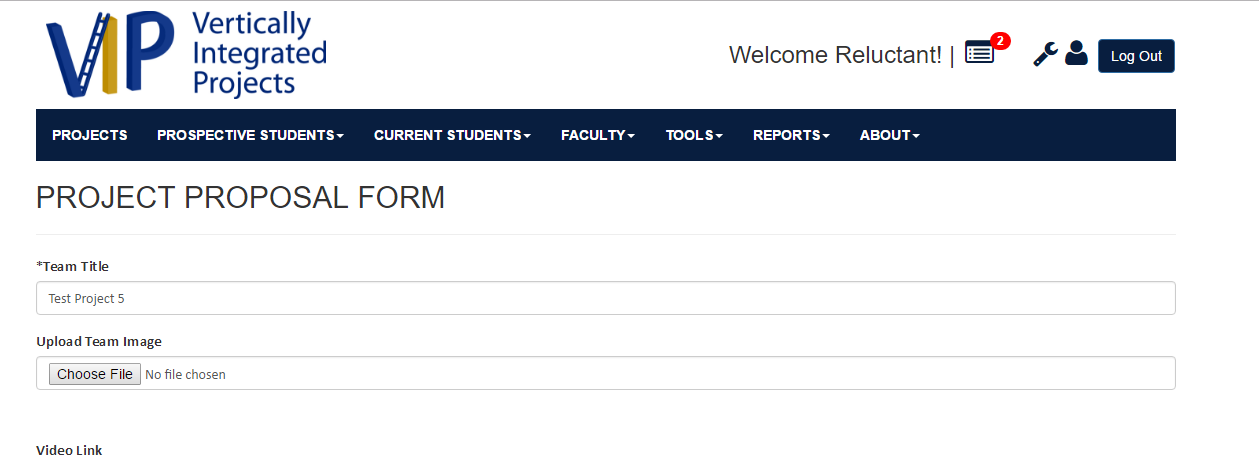
End Time: 6:32pm

After a show and tell presentation, the implementation of the following user stories were accepted by the Product owners: All of them

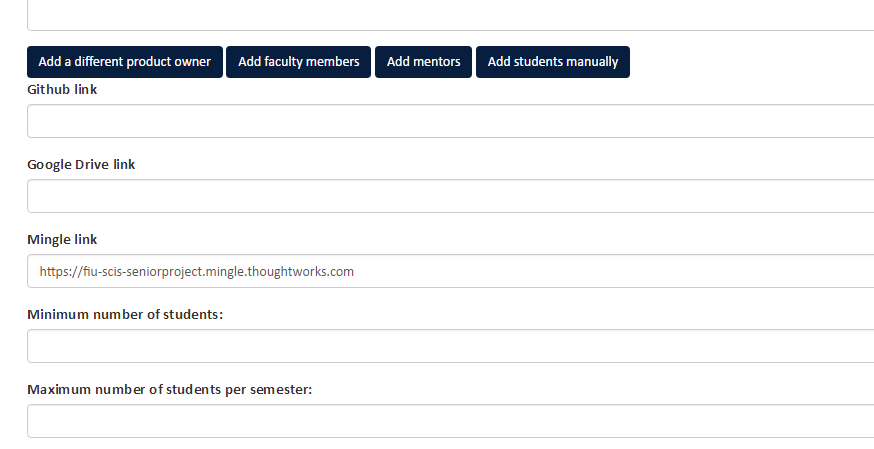
## Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

**User Guide:**

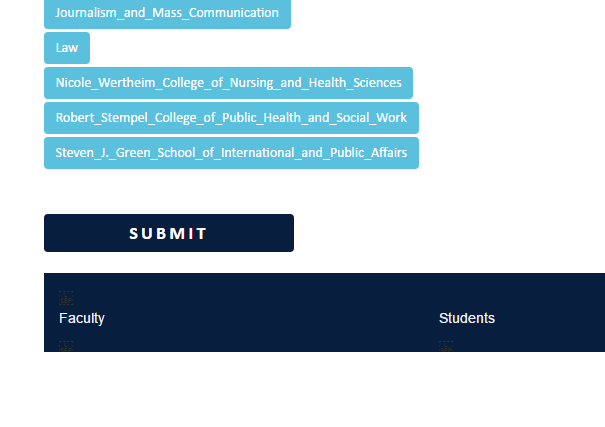
Step 1. User Navigates to Project Proposal Form



Step 2. User Scrolls down to the “Mingle Link” text box in the form



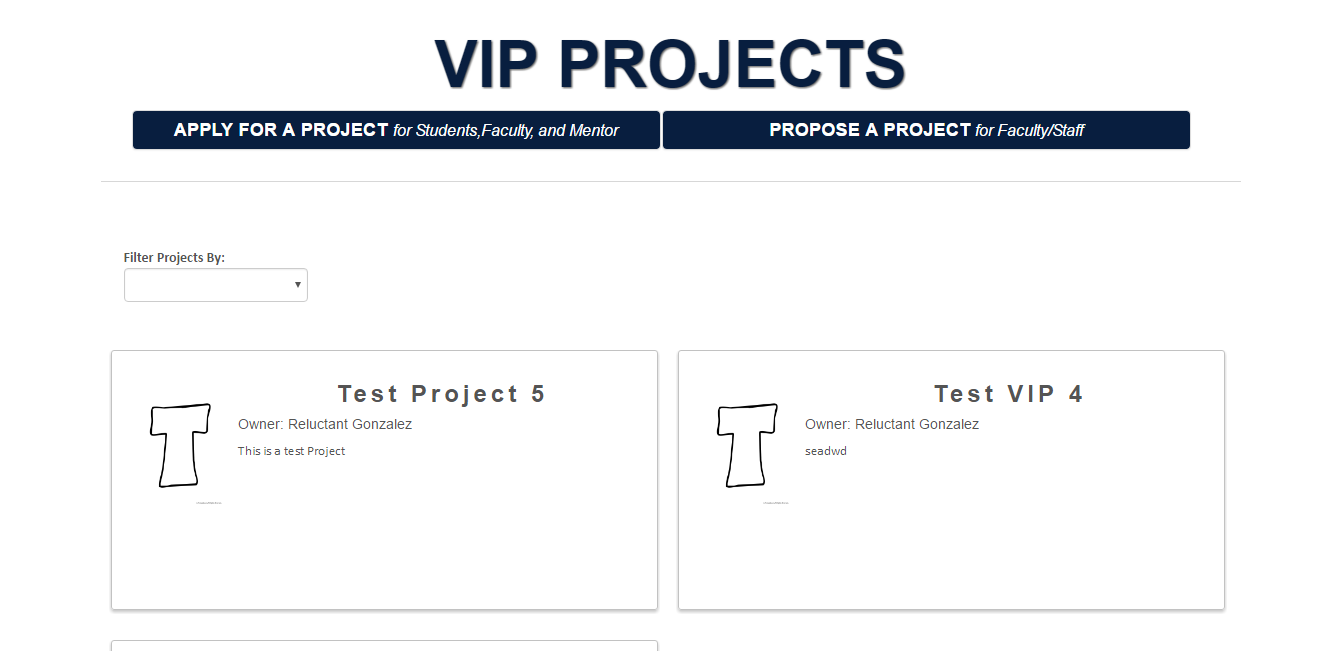
Step 3. User Submits the Project



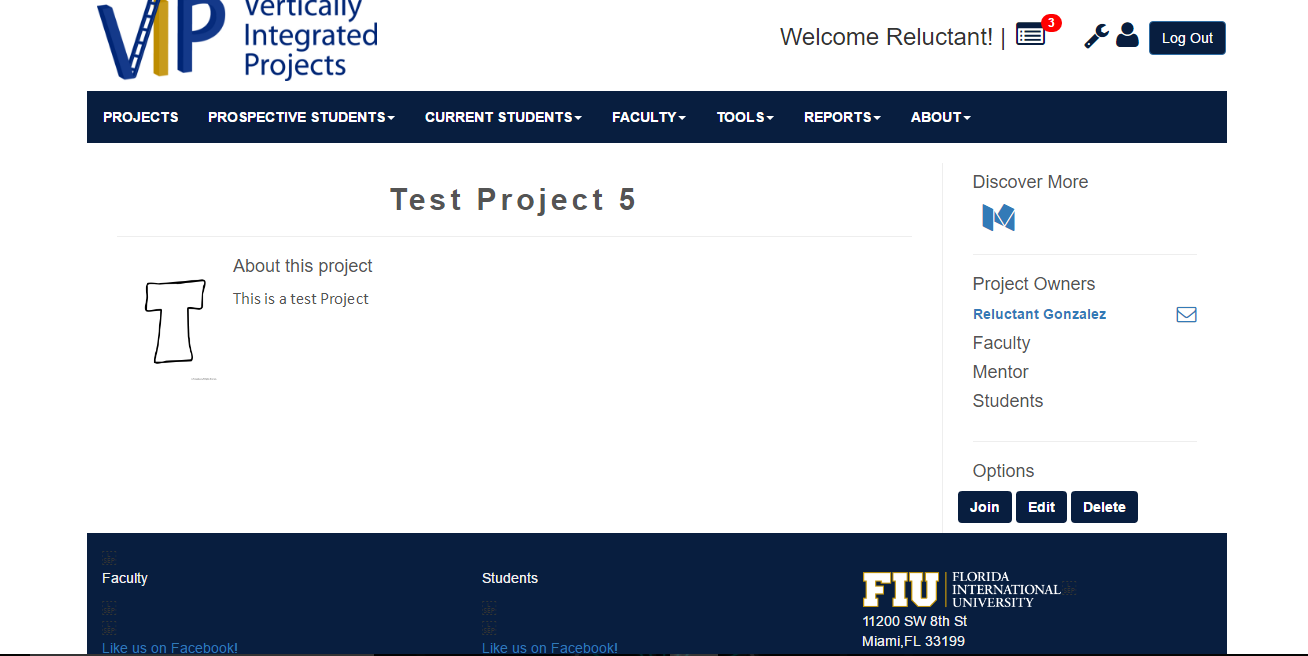
Step 4. The Project is now submitted, once a PI/CoPI has approved the project, the user must navigate to the Projects Page by clicking the “Projects” Button on the header of the site



Step 5. User Navigates to their Project



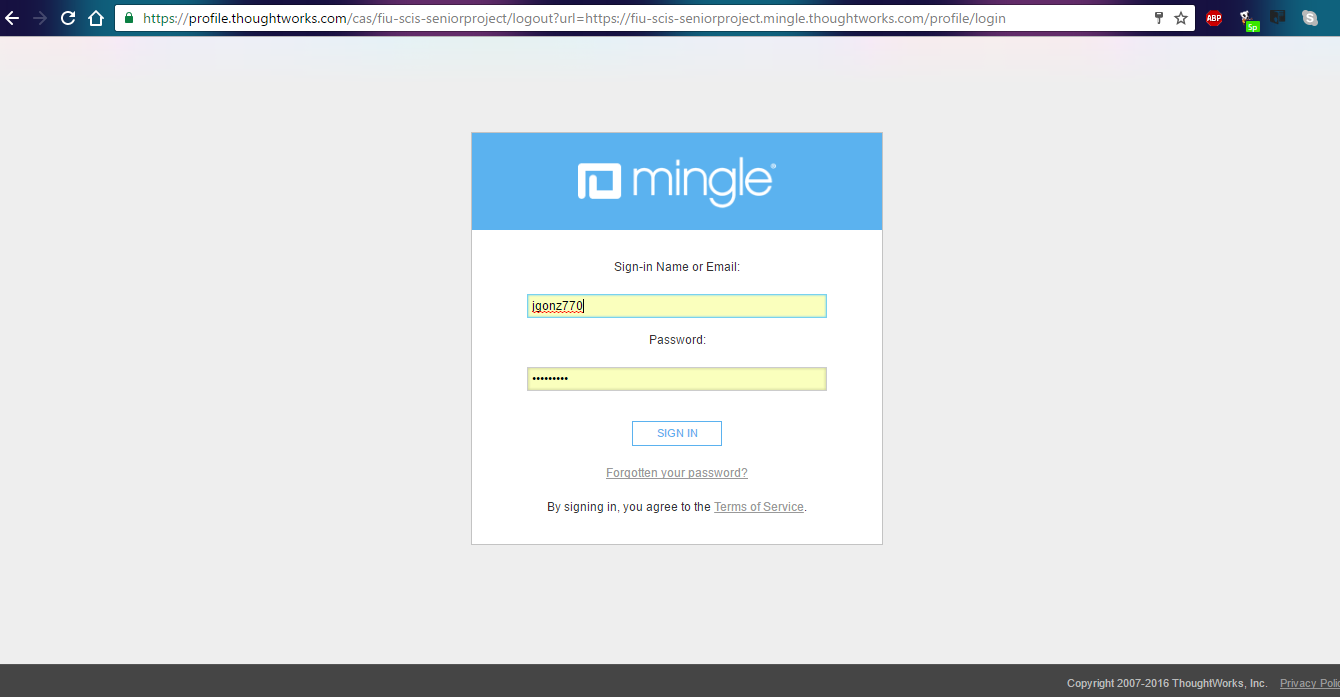
Step 6. User Navigates to their Project Details Page



Step 7. User Navigates to the “M” social icon under the Discover More section of the webpage



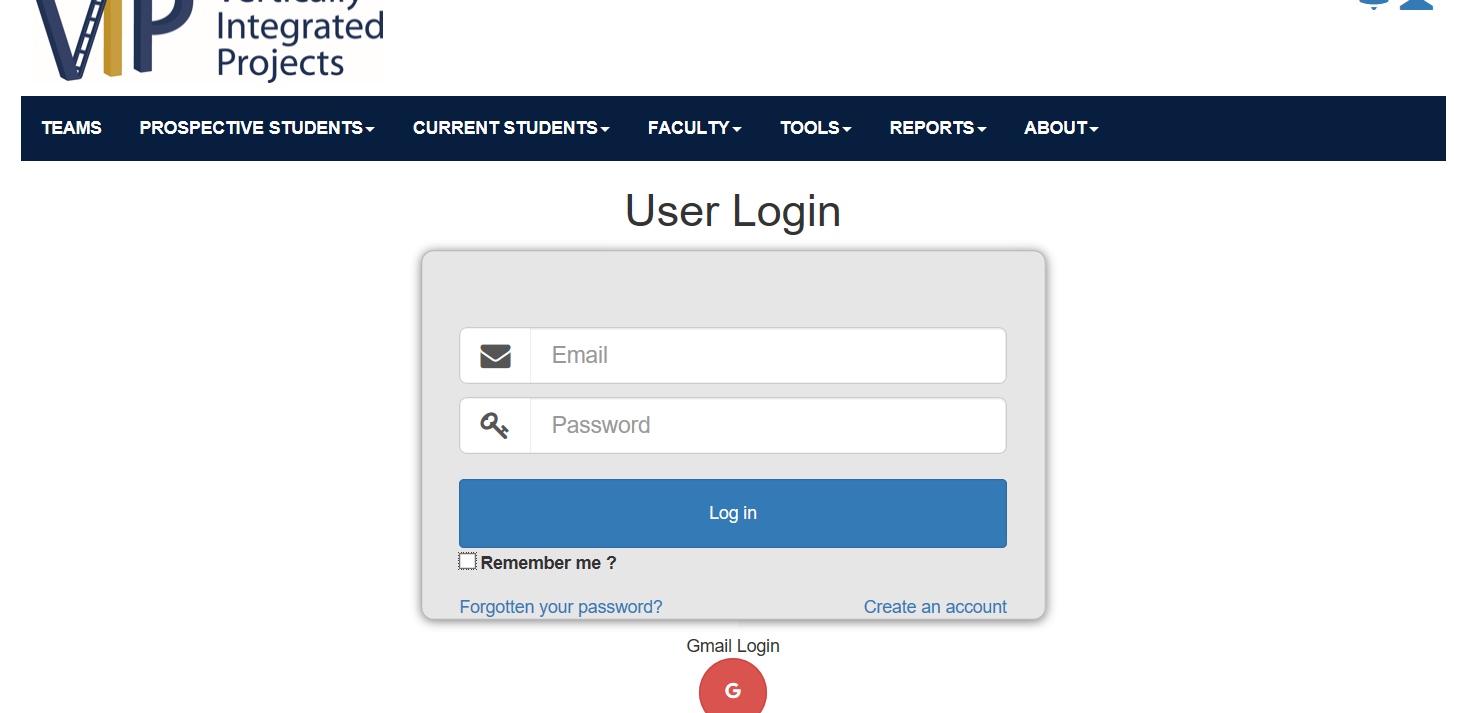
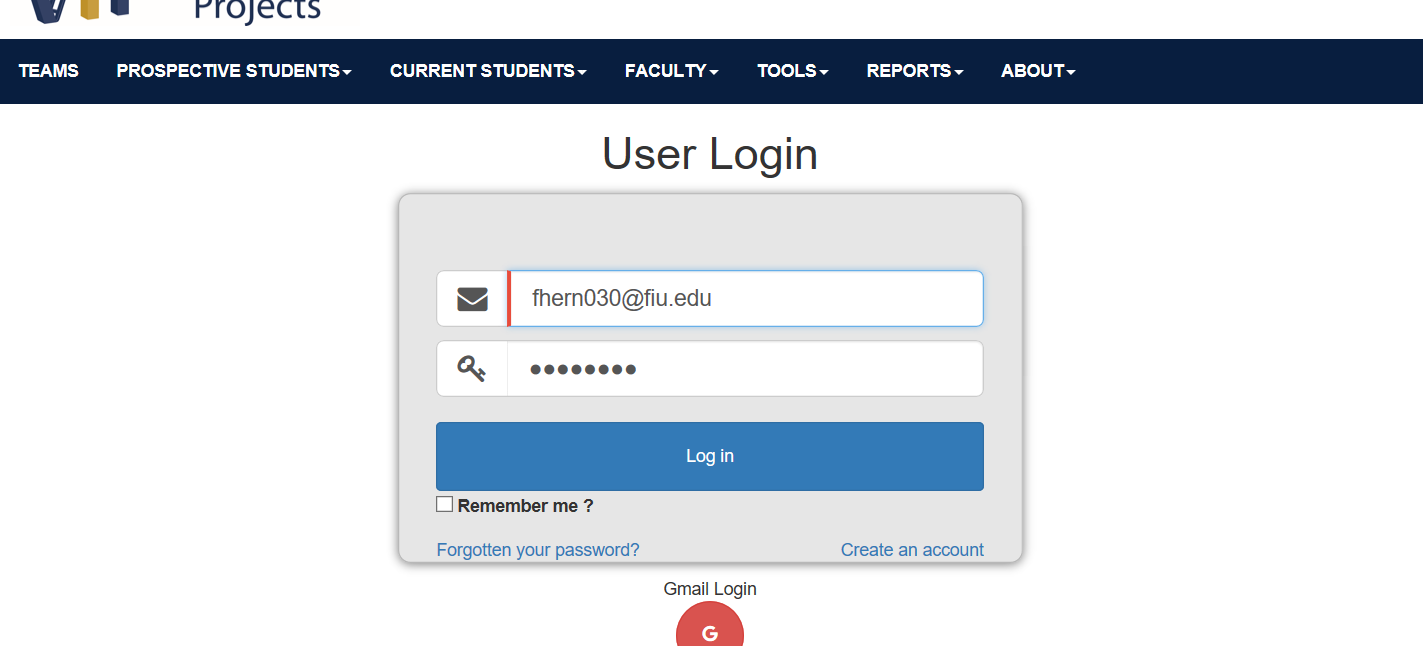
Step 8. The user clicks the “M” social link, and is redirected to the Mingle Login Page

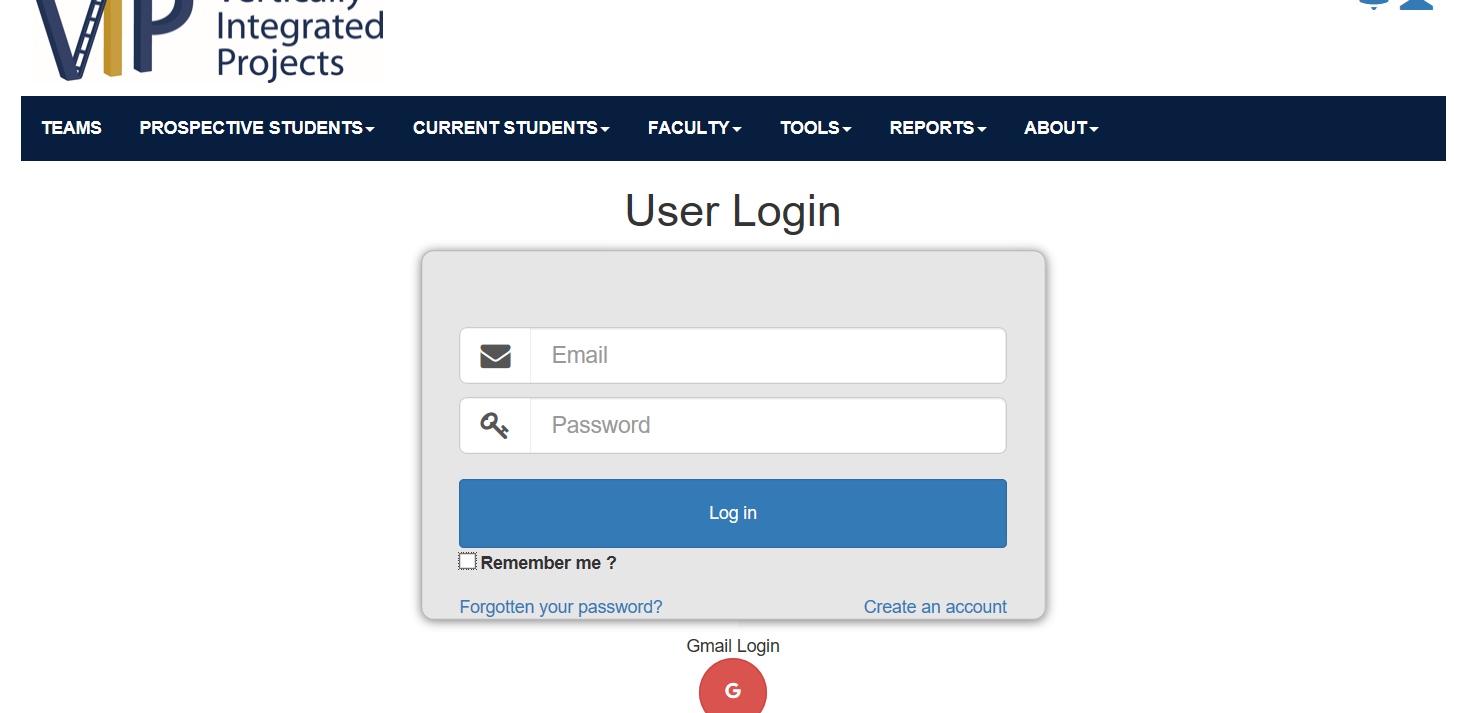


From here they login, and access their Mingle as needed.

**User Guide:**

Step 1. User Navigates to User Login Form

Step 2. User Enters User Login Information



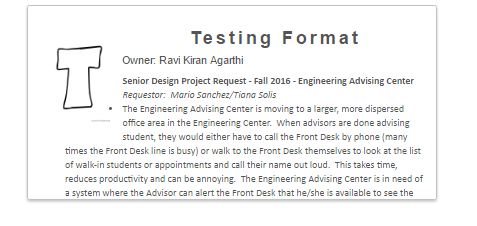
Step 3. User Check Remember\_Me CheckBox

## Visual User Guide

Project Proposal Form



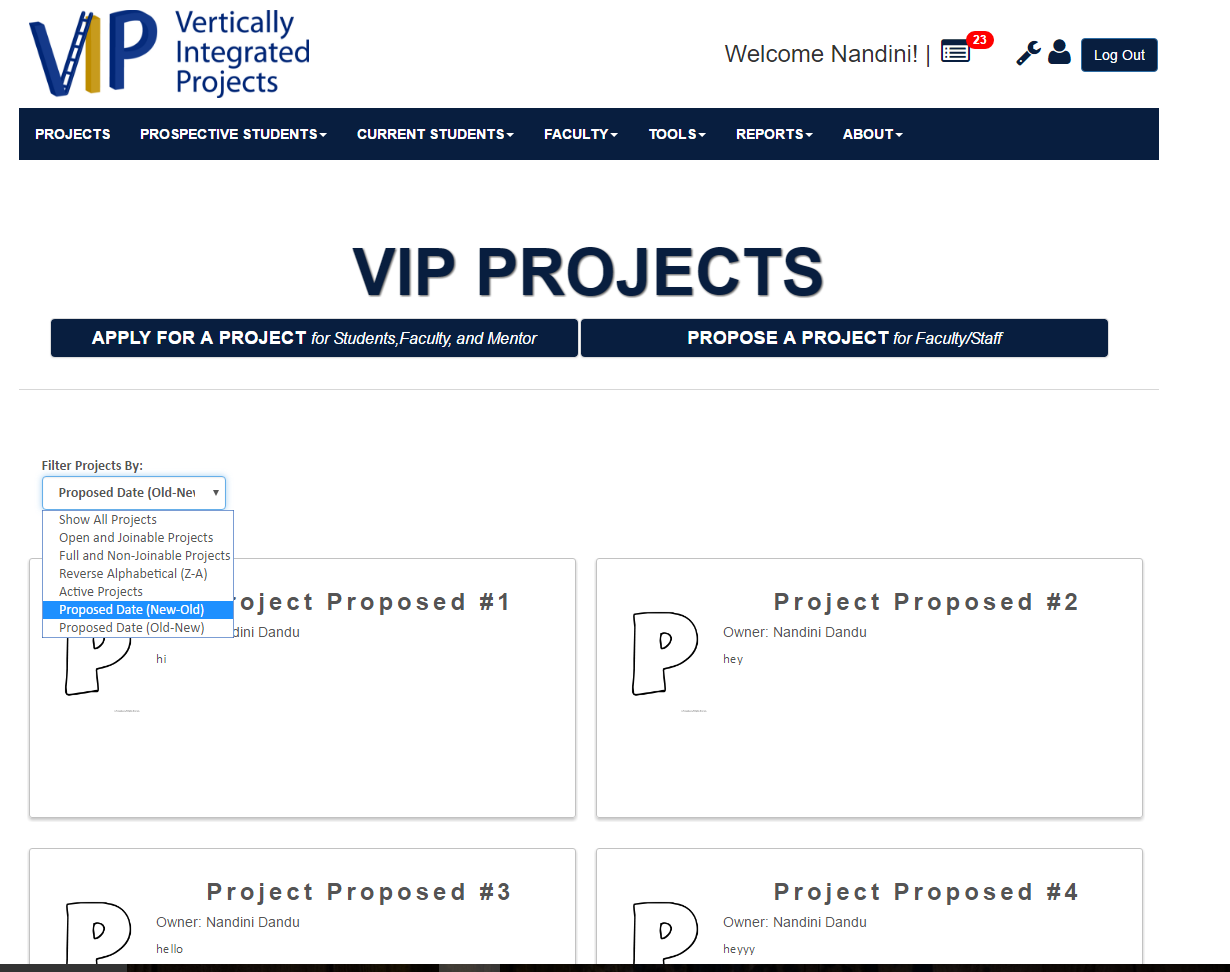
VIP Projects



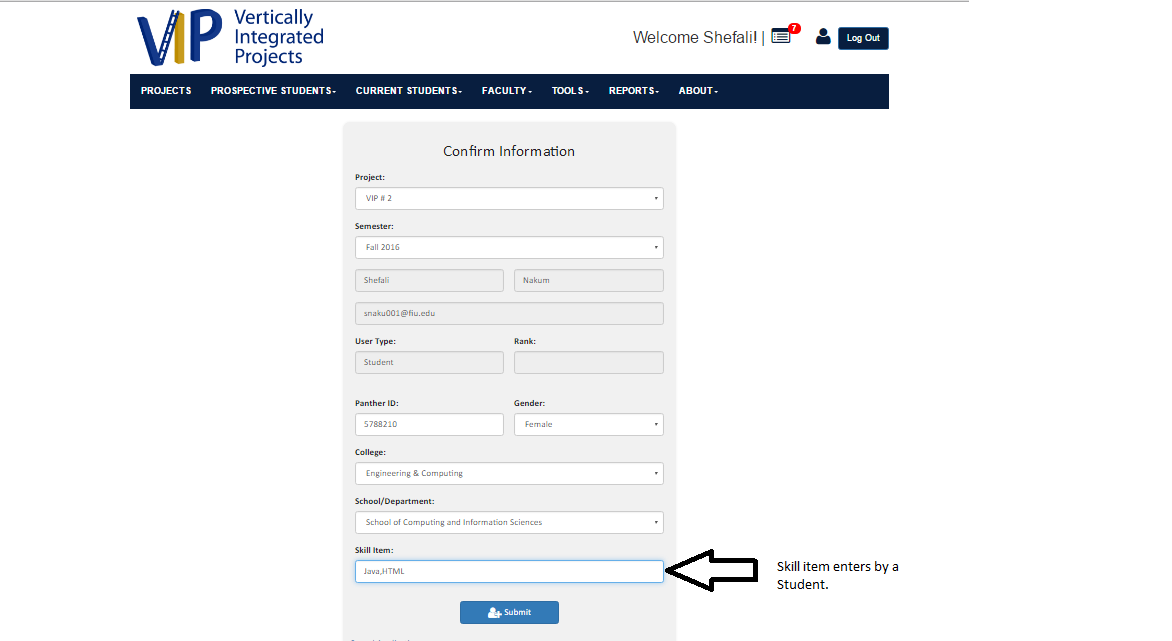
Vip Projects Detail view

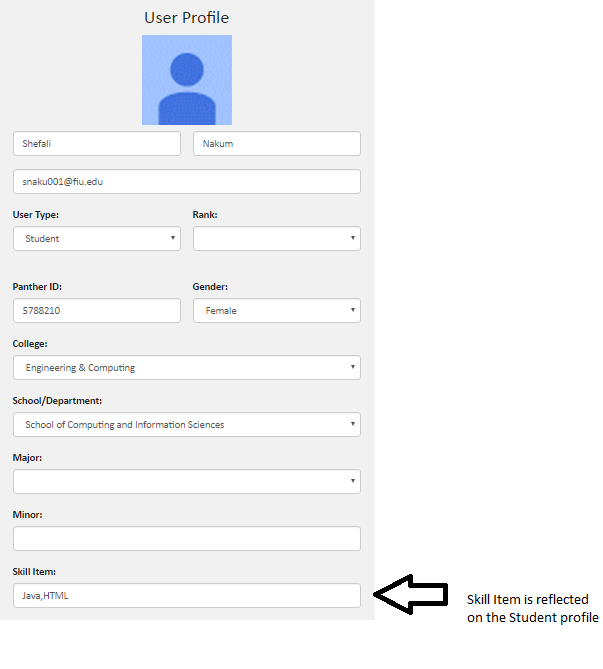
## Visual User Guide

Developer could be either the student or a professor who would like to sort and know the projects.

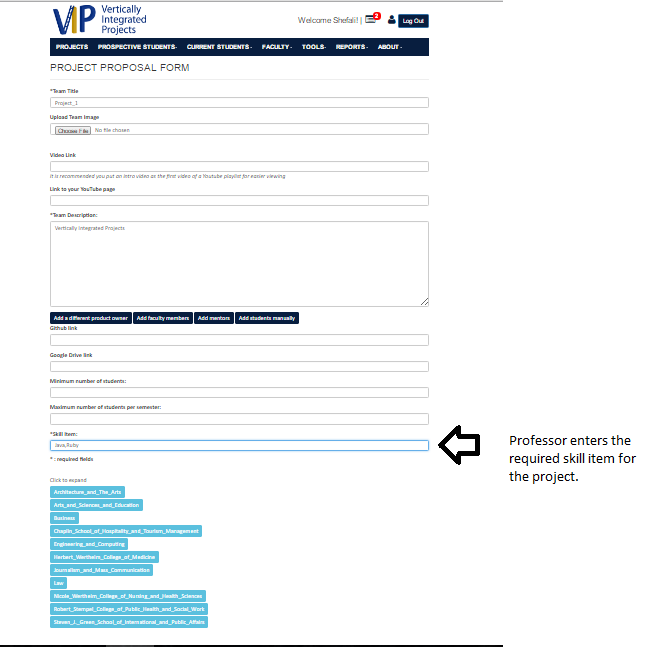


* **STUDENT APPLYING FOR PROJECT**





* **PROFESSOR PROPOSING A PROJECT**



# How to Become a CoPI: A Guide

CIS Senior Project

Joseph Gonzalez

Florida International University

VIP 4.0



***Preface***

In VIP, PI/CoPI are super users, in charge of approving projects, and verifying accounts, and able to access every feature of the site. In order to facilitate developing for the site, it is essential to become a PI/CoPI on the localhost server. This guide will briefly explain the process of manually adjusting your account user type to PI/CoPI via the MongoDB.

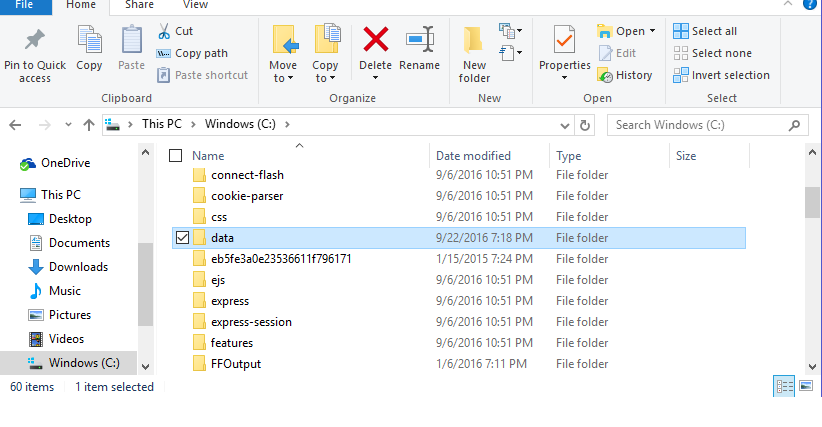
***Prerequisites***

For this guide, it is assumed you have completed the following:

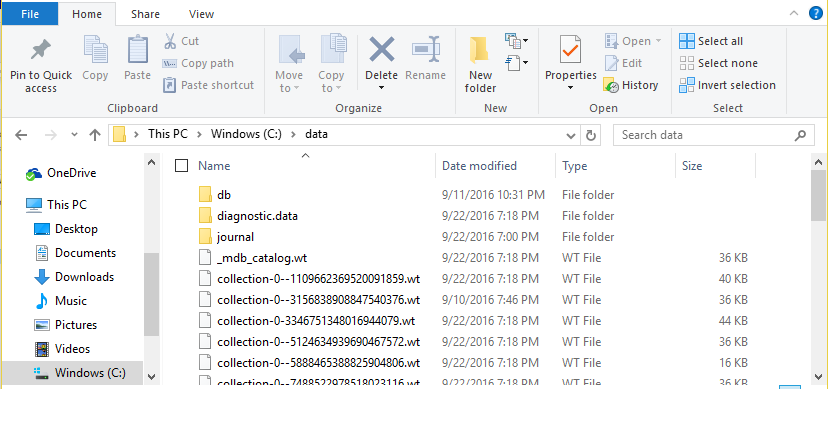
* Download and Install MongoDB
* Download and Install Node.js
* Downloaded the current version of VIP onto your system.

***Step 1: Setup***

Before we begin, let’s make sure your files are where they need to be. You will need to open your C drive (or whichever drive you installed MongoDB) and navigate to the “data” directory.



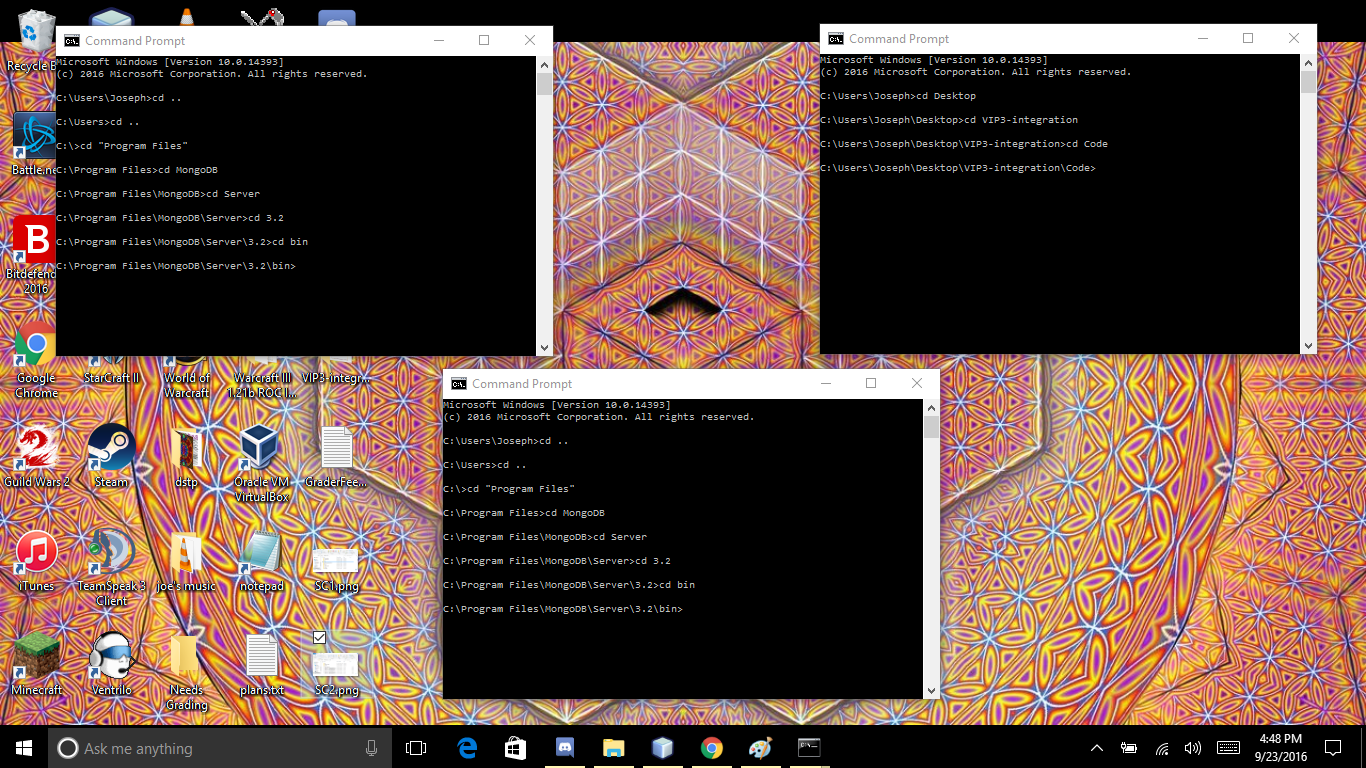
This is going to be the directory where your MongoDB collections are stored. Among the VIP files, should be a data file. You need to take those files, and place them in this data file. The results will look something like this



Once this is setup, you’re ready to begin.

***Step 2: Running VIP***

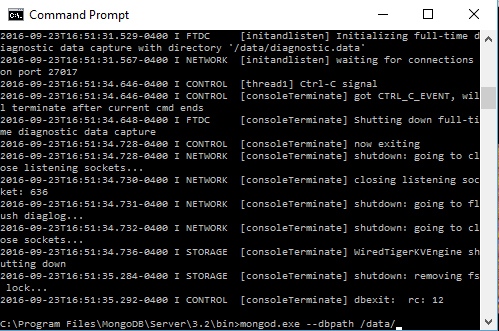
So, we begin by opening 3 command prompts. In 2 of these command prompts, you will need to navigate to the “bin” folder in MongoDB, and in the last, you will need to navigate to the “Code” folder in the VIP directory.



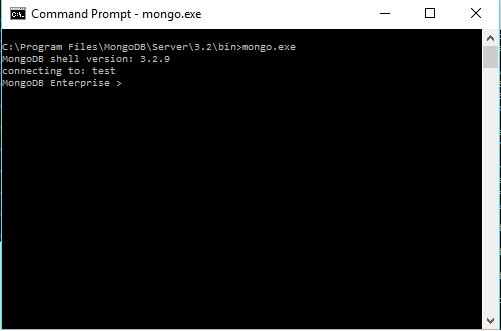
First, in one of the Mongo Command Prompts, we need to run mongod.exe, but we need to set it down the database we set up in step one. To do this enter the following into command prompt

Mongod.exe --dbpath /data/

(enter this verbatim)

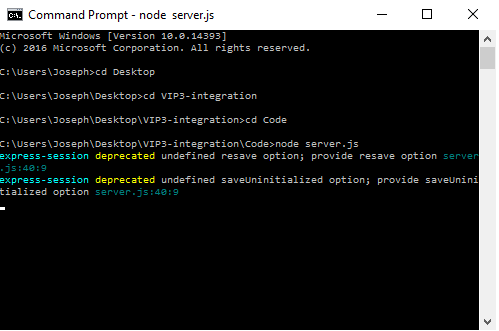


Press enter, and in the other MongoDB command prompt, run mongo.exe

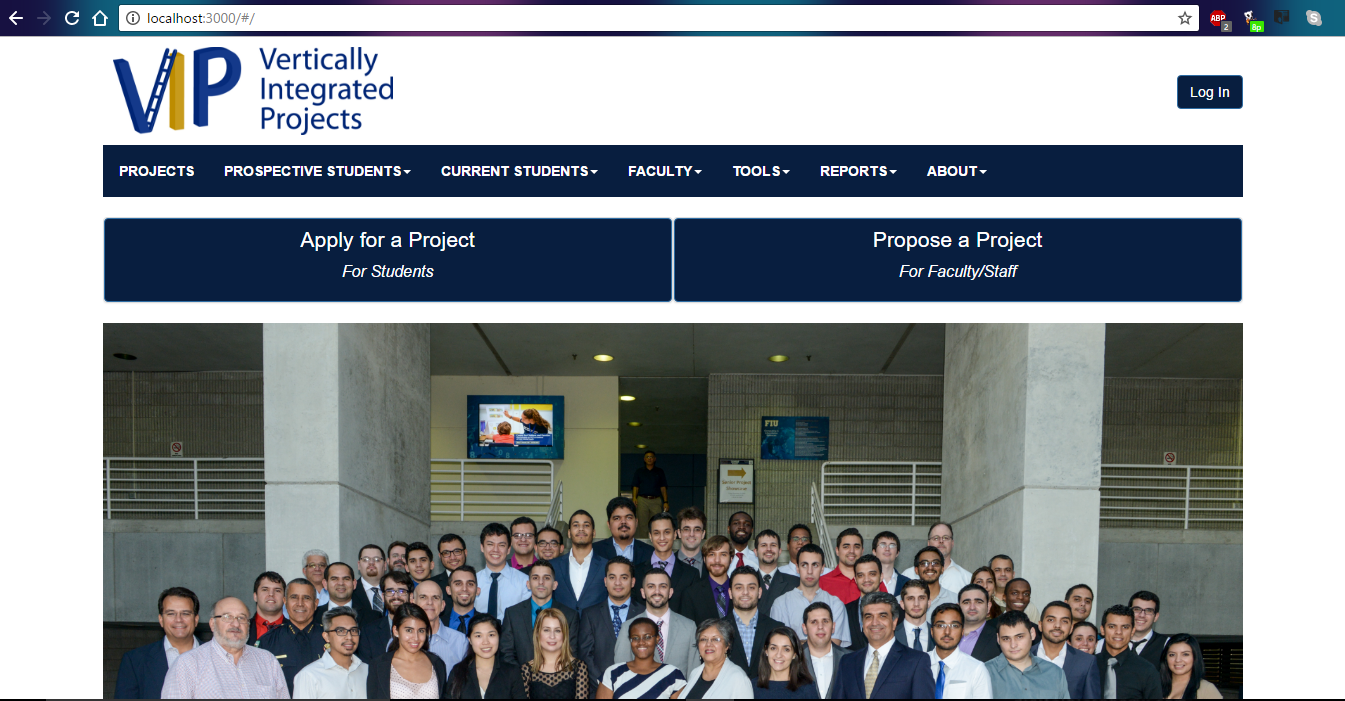


Finally, in the last command line, we now run the server, by entering the following:

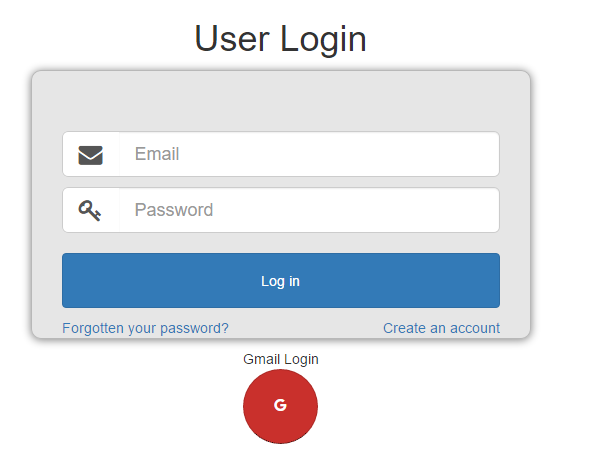
node server.js



From here, go to your preferred web browser, and in the URL bar enter localhost:3000 to access your locally hosted VIP server.



At the moment, your account is not in our database. To add it, need to login to VIP on your localhost for the first time. You can do so by clicking the “Login” button on the top right hand of the screen as seen above. From there, simply click the login via google button on the next page.

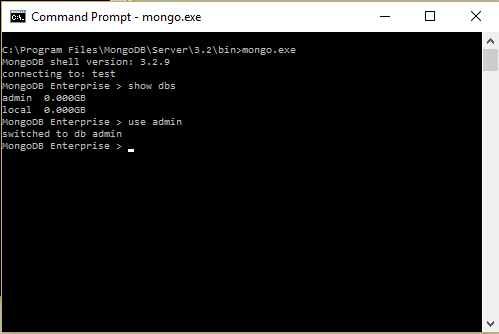


(Note: Once you login you will be redirected to the live server (vip.fiu.edu) simply go back to localhost:3000)

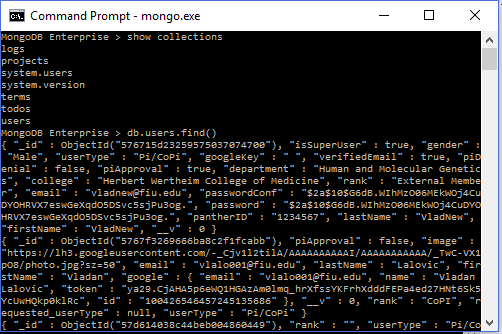
Now that you’ve logged in, your account information has been added to the users collection in the database, which means we can modify it in the database.

***Step 3: Modifying the Database***

Now that your user information is in the database, we can modify your account status manually. First, we need to switch to the right database. Open up the command prompt where you are running mongo.exe, and enter “show dbs” minus the quotes. This will give a list of the databases on the system. Among them you should see “admin”. This is the database we’ll be using. To access it, simply enter “use admin” minus the quotes.



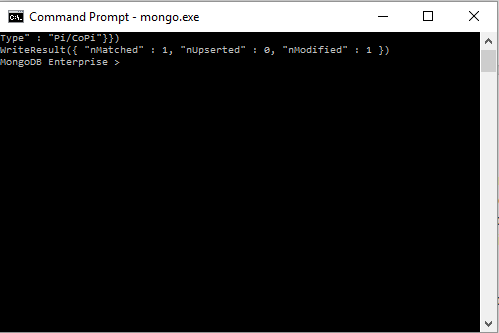
From here, if you enter “show collections” you will see a list of the collections in the database. Among them is the “users” collection which contains your user information. To view the tuples in the collection enter “db.users.find()” without the quotes.



This command lists all of the user accounts on the site. Among them now should be yours, quickly look through the collection to confirm.

From here, we now use the update() function to modify your userType from Student to PI/CoPI. The update() command takes 2 arguments, an identifier so that mongo knows which tuple to alter, and a target tuple that you’re going to alter. For this example we’re going to use your last name as the identifier, and user type as the target. So to alter your account enter following line into the command prompt:

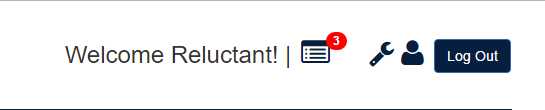
db.users.update({ “lastName” : “ <your last name> “ } , {$set”:{ “userType” : “Pi/CoPi”}})



The write result line indicated how many tuples were matched, and how many were modified, as seen above it should be 1 and 1. Re-enter the db.users.find() command to verify your updated user Type, or simply return to the localhost VIP and observe your new Pi/CoPi status to confirm.

***Conclusion***

Now you are officially a PI/CoPI on the VIP local host, and are able to access all the features of the site at your leisure, making it navigating the entire site a breeze. When on the site you should see some updated features, such as the tool icon, and notifications.

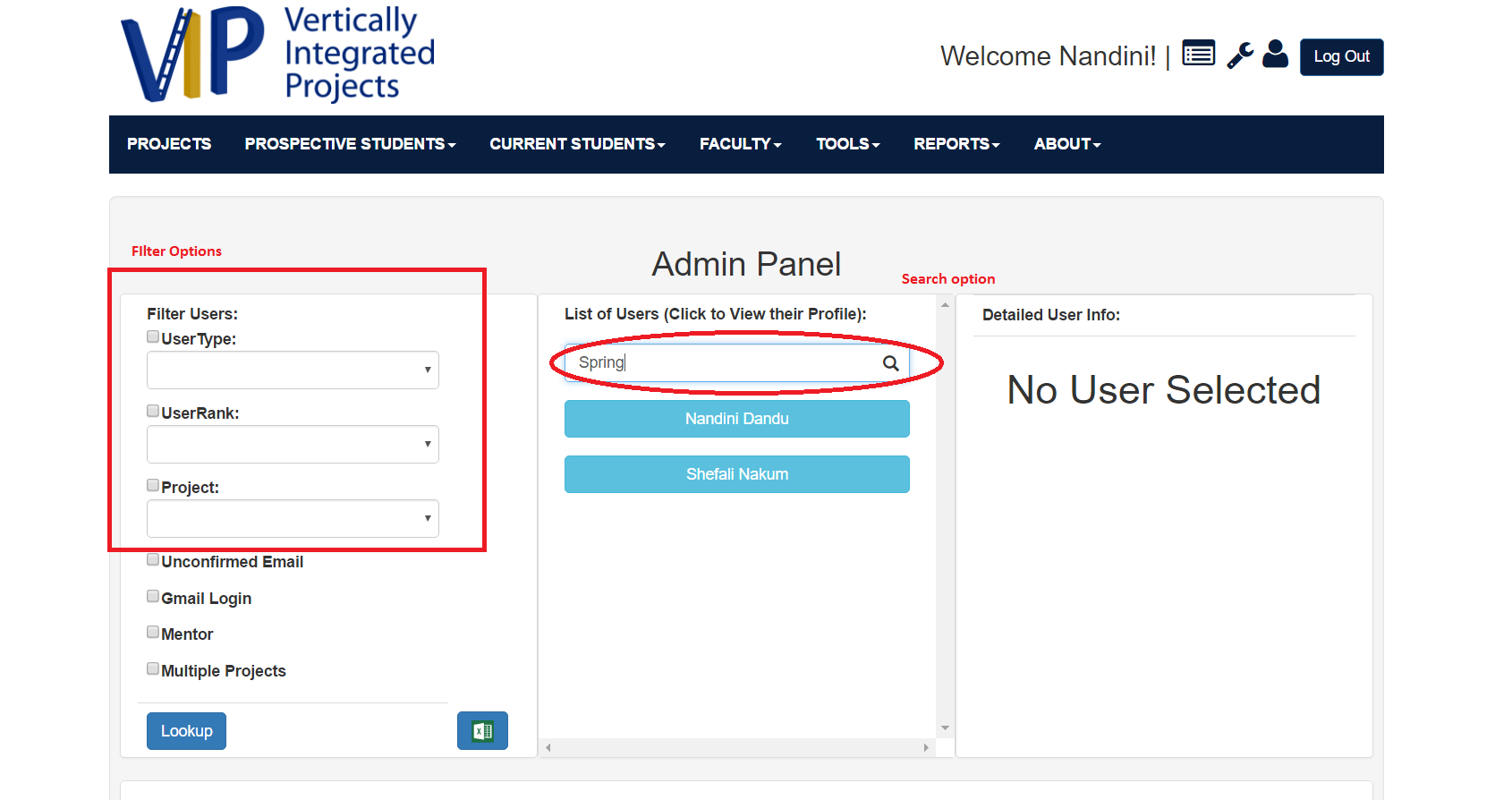


Now you are ready to begin working on the VIP site!

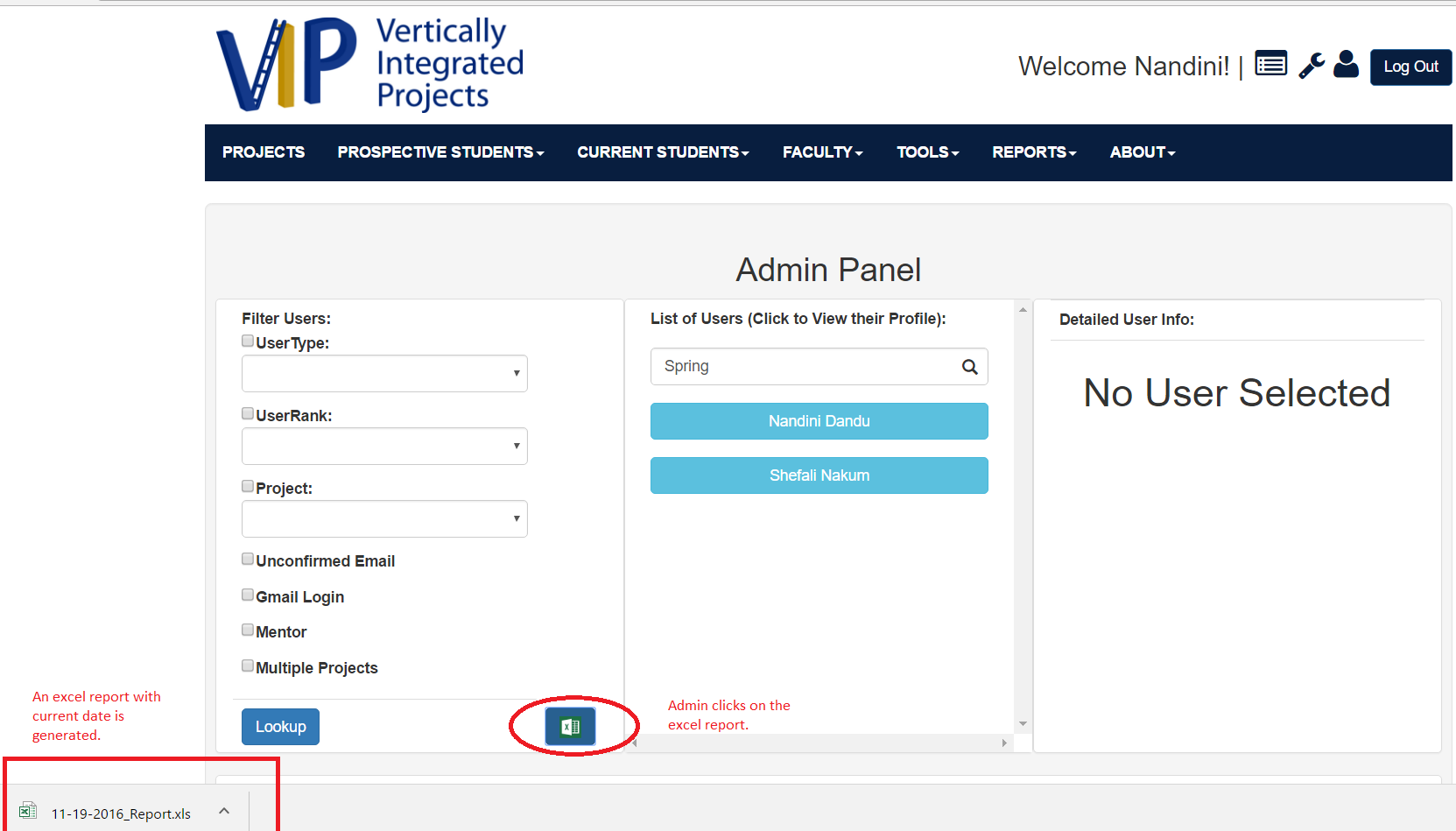
# User Guide:

Following are the visual pictures of the excel report generation.

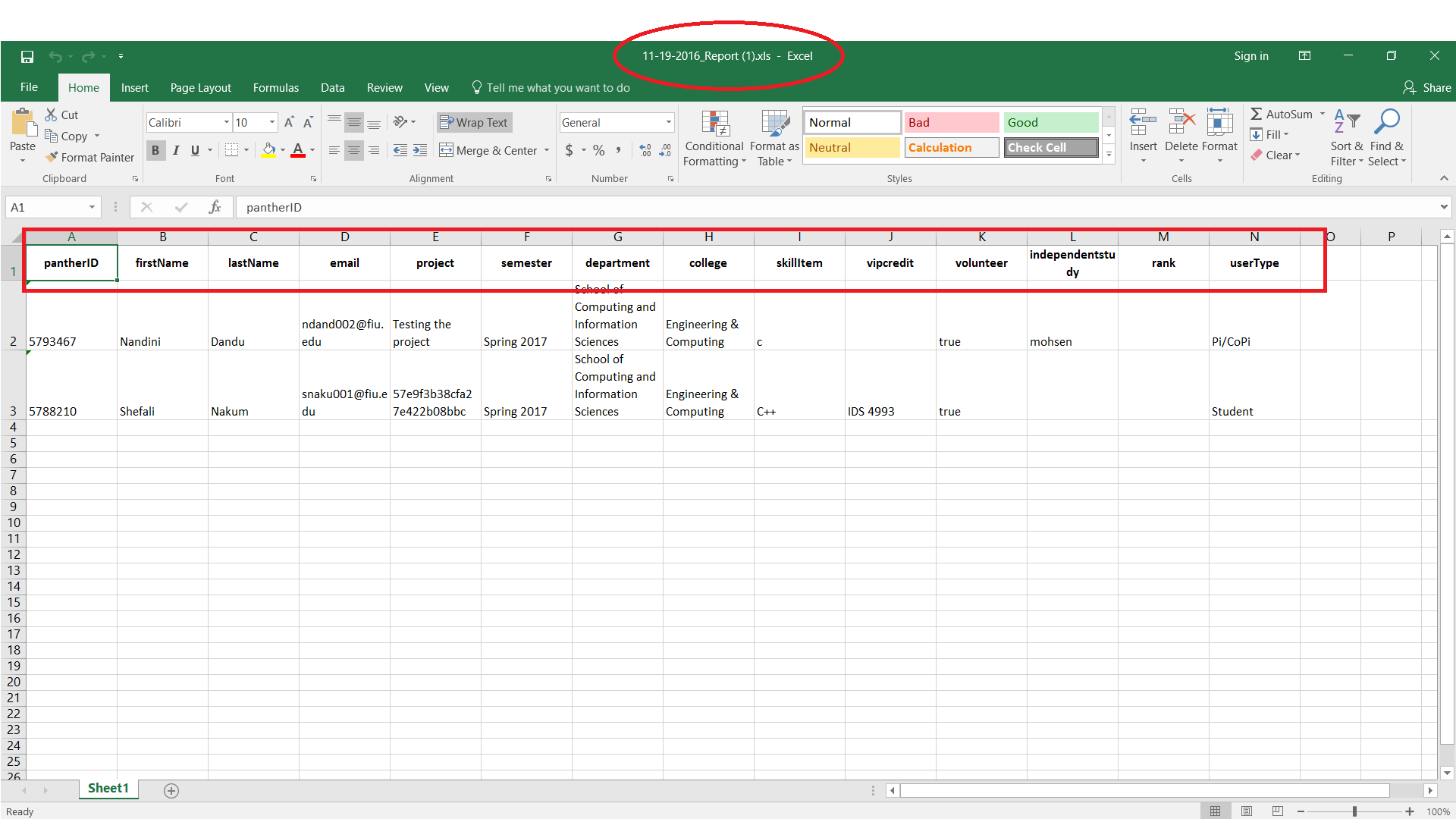
1. The admin is in the login page. Admin selects the filters and clicks on lookup to get the required details or else he simply searches using the search box.



Admin clicks on the excel icon and a report is generated automatically. The name of the report generated with **current data and Report.xsls**

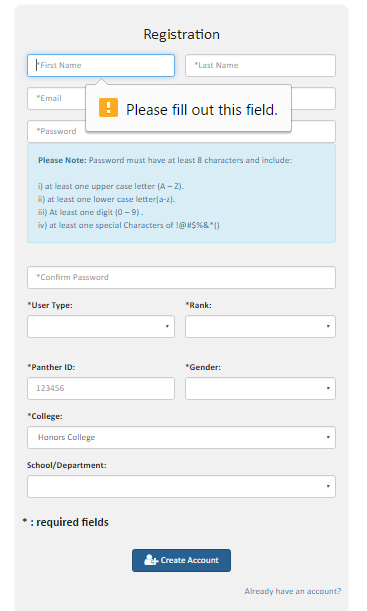


The excel report with the required information.

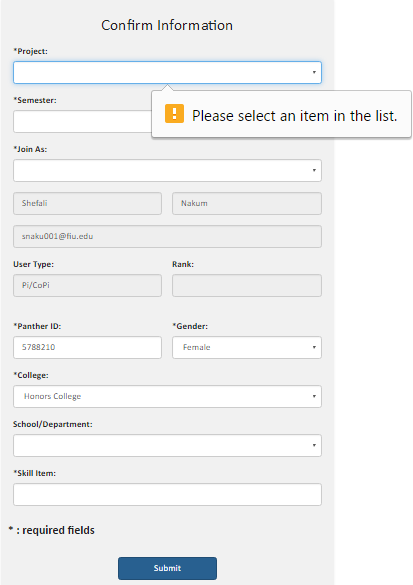


**Visual User Guide**

1. **Registration form with “Honor college” and all required fields, marked with a star\***



1. **Apply for a project form with “Honor college” and all required fields, marked with a star\***



1. **Proposal for a project form with “Honor college” and all required fields, marked with a star\***

**User Guide**

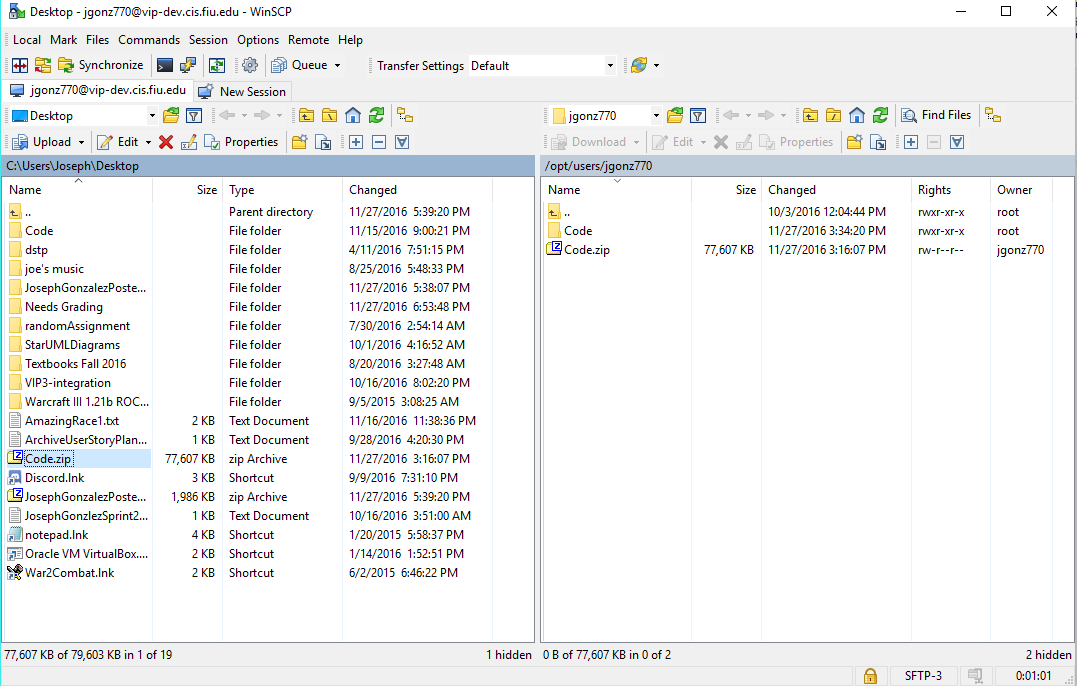
For this guide, I am going to go over how to update the Dev Server. For the Live server, do the exact same thing, but instead of logging into vip-dev.cs.fiu.edu via Putty and WinSCP you’ll be logging into vip.fiu.edu. All other instructions are the same.

So, first open WinSCP and login to the dev server (vip-dev.cs.fiu.edu)

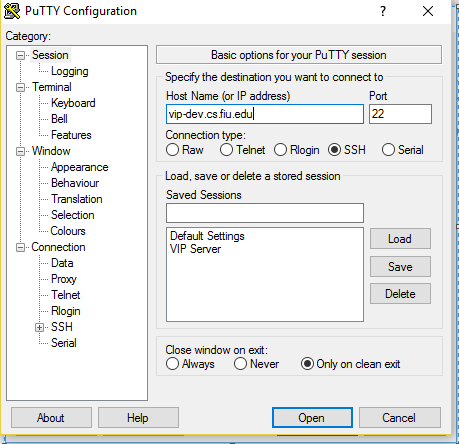


Login with your credentials, your email is your username (minus @fiu.edu) and your password (by default your password is the first letter of your first name, followed by your Panther ID, followed by the first letter of your last name such as “ e1234567g “.

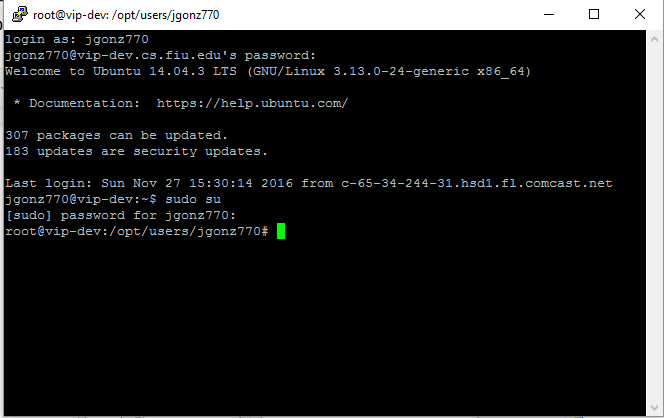
From there, take note, the left half of the screen is your computer, the right half is the dev server. Navigate to which ever directory you want to run the dev server from (in my case /opt/users/jgonz770) and simply drag and drop a zipped version of your code from your computer to the dev server in WinSCP, in this example the file is named Code.zip. (In the event this is an update, first delete the existing Code.zip file in the dev server, before moving the new one into place)



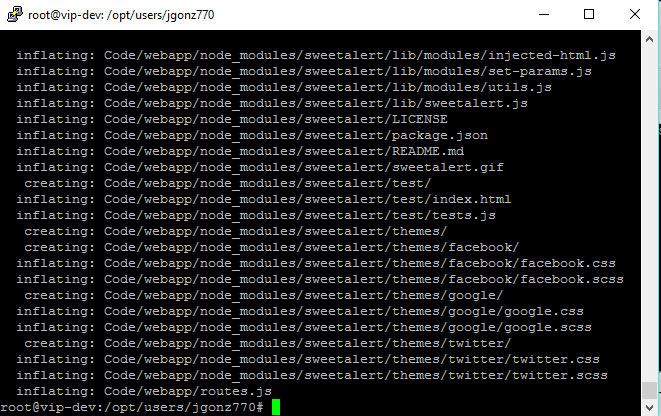
This is all we need to do in WinSCP, now open putty, login to the same site, and you will log in with the same credentials.



Your login credentials are the same as they were for WinSCP. Once logged in, you’ll need to become the root user. Simple enter “sudo su” then re-enter your password (by default Linux keeps passwords hidden, so don’t freak out if you don’t see anything while typing it in)



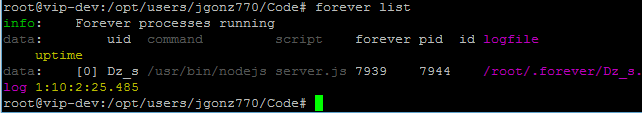
From here, navigate to the directory you placed the Code.zip file, and unzip it by entering “ unzip Code.zip “. (note, if you are performing an update, first delete the existing Code file by entering “ rm -rf Code “ ).



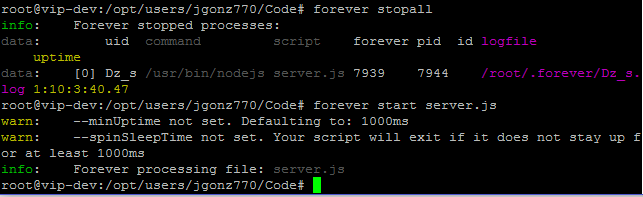
From here, navigate into the code file by entering “ cd Code “.



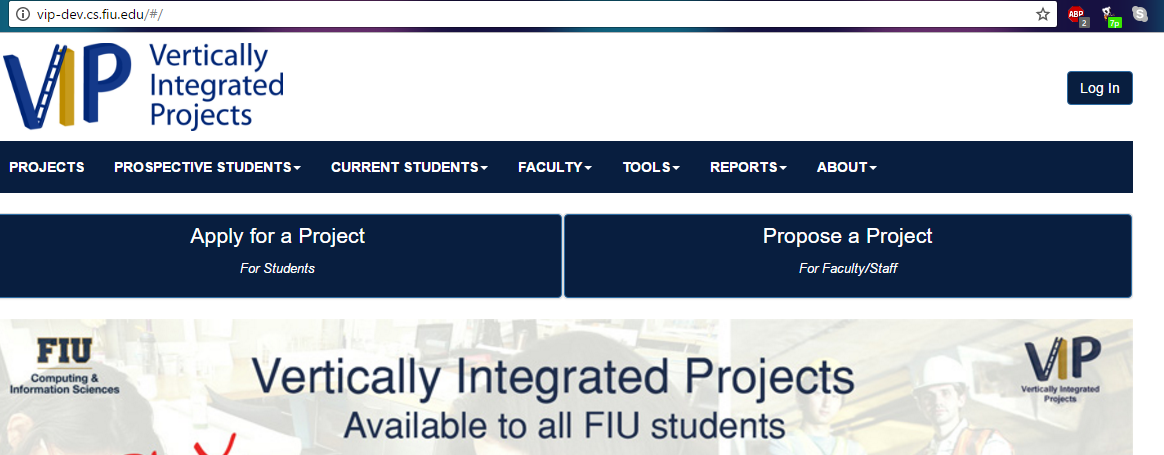
Now if you enter “forever list” you’ll see a list of all running processes on the forever node, this is used to run the site on the live server.



You are going to want to end the current processes by entering “ forever stopall “. Then simply start the site again by entering “forever start server.js”.



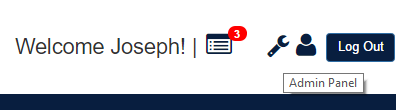
Now, pending you didn’t receive any error messages (this means there’s a problem in the code) the dev server is running on vip-dev.cs.fiu.edu



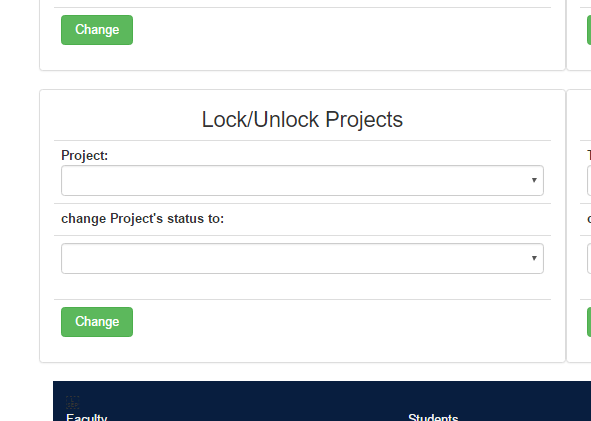
\*Note that once you enter “forever stopall” the site will be down, so make sure you don’t run it until you are ready to run the updated Code.

**User Guide**

Once logged into the site, navigate to the admin panel



From there, scroll down to the “Lock/Unlock Semester” Panel:

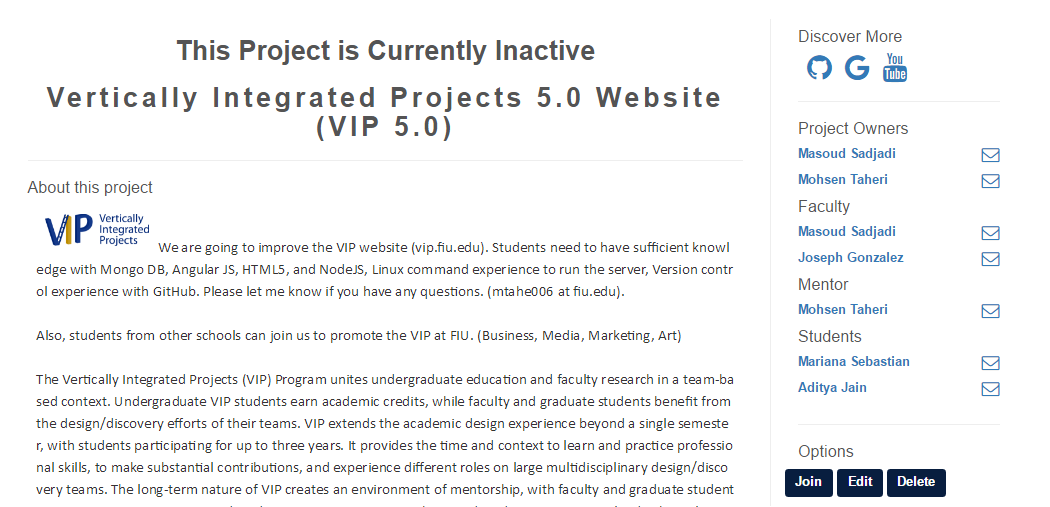


Select an option in the “Project” menu, and an option from the “change Project’s status to” menu. Then click the “Change” button underneath. You have now locked a Project.To test if the lock is working first, try to propose a new project for that locked Semester. In this example we’ve Locked “VIP 5.0”.

Now in the Project’s Page (<http://vip.fiu.edu/#/vip-projects>) we can scroll down and navigate to the VIP 5.0 Project’s detailed page:



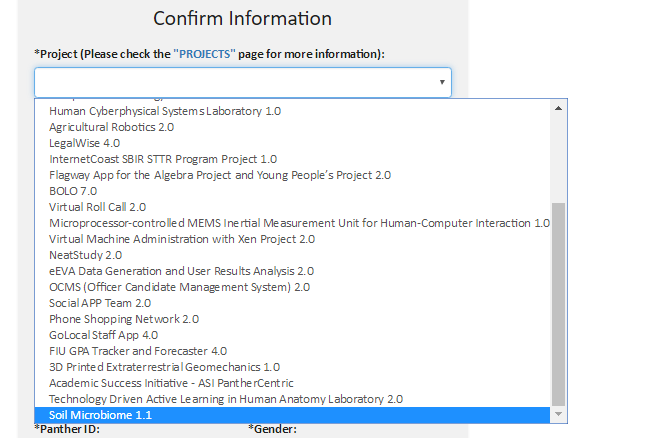
From here, note 2 things:



1. The line “This Project is Currently Inactive” above the project title is now there to indicate the project has been locked.
2. When attempting to select the “Join” button, the following message appears



In the Student Application Form (http://vip.fiu.edu/#/studentConfirmation/), also take note that the locked project will no longer appear as an option in the “Projects” dropdown menu:

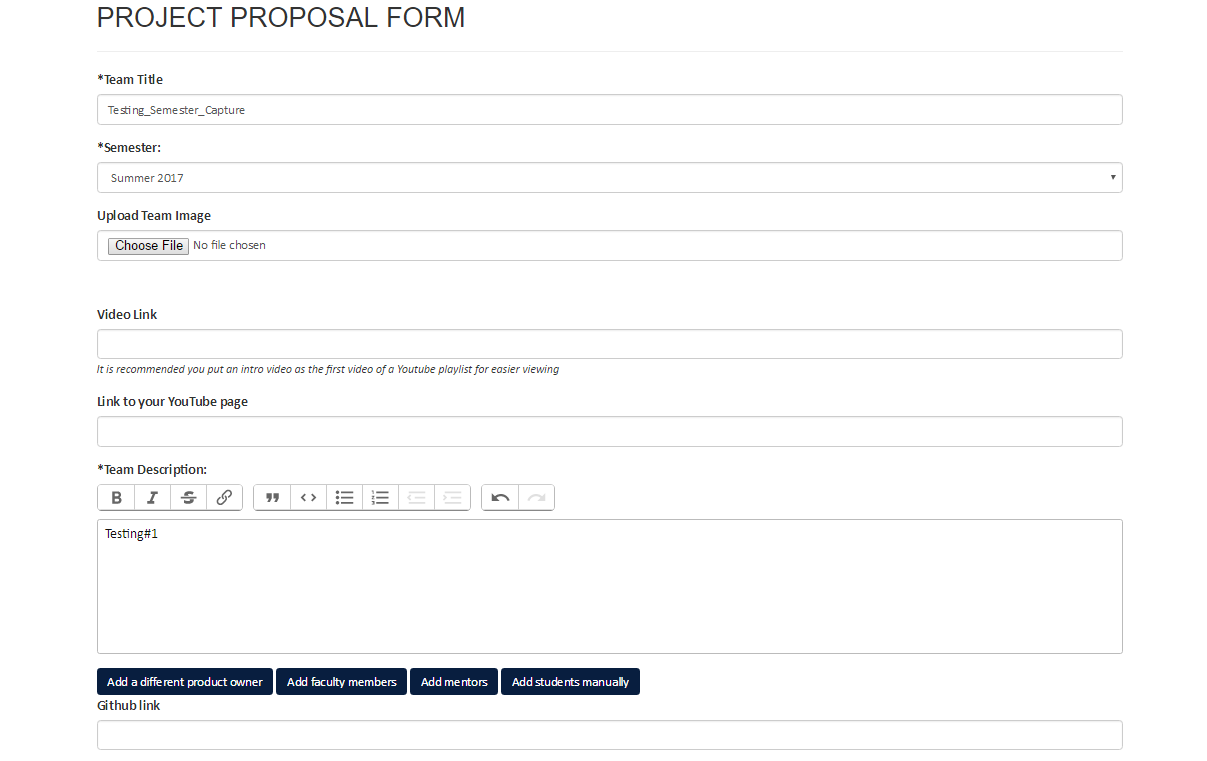


So thus the project is locked and inaccessible by any new members.

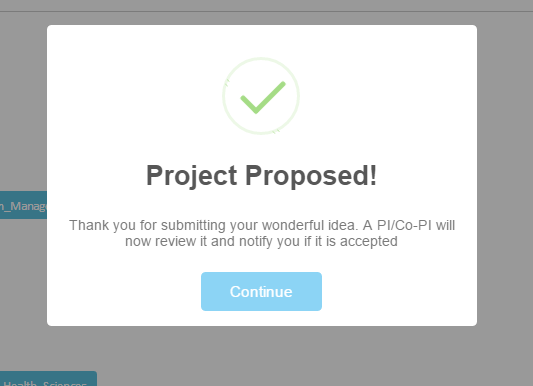
## Visual User Guide

Project Proposal Form:

1. Staff/Faculy Proposing a Project Title “Testing\_Semester\_Capture”



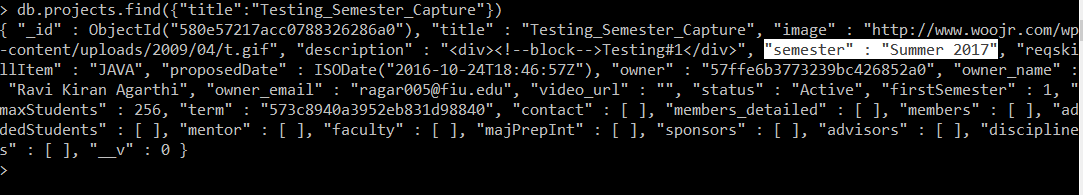
2. Submits the From



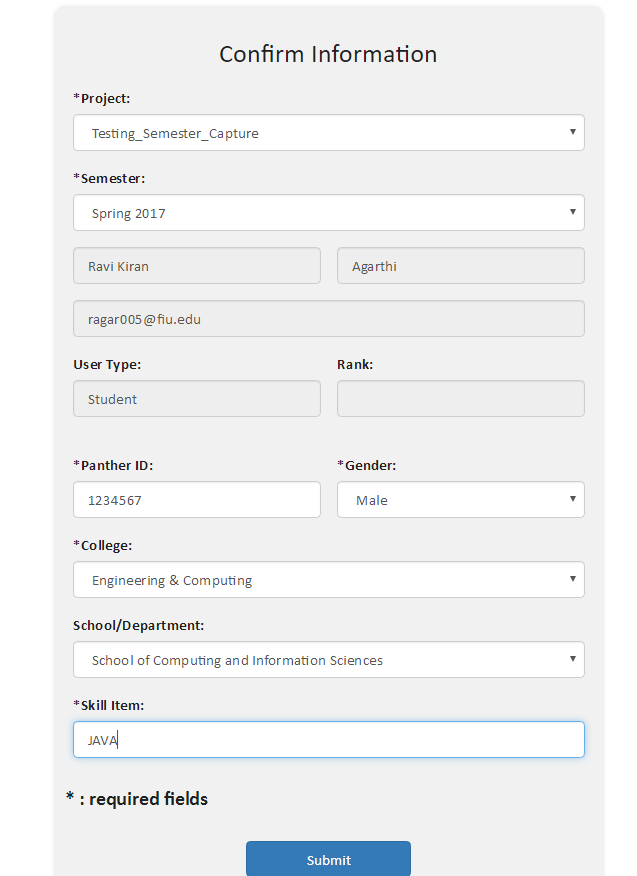
3. After Pi/CoPi approval



Semester is saved in Database



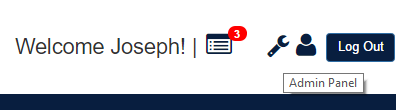
Apply for a Project From

1. Student applies for a Project

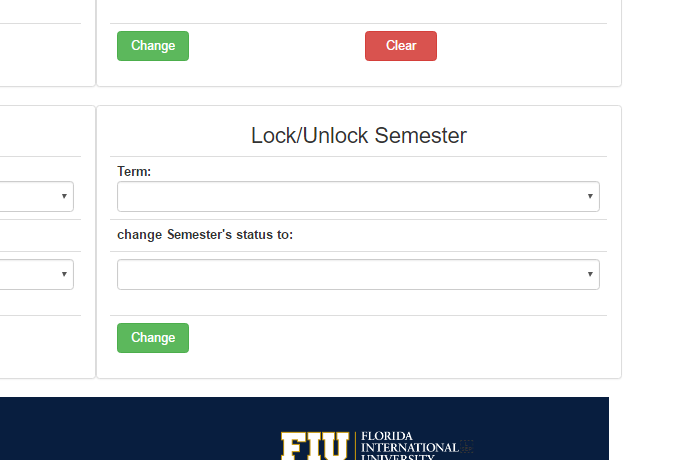
2. Capturing semester in Database

**User Guide**

Once logged into the site, Navigate to the login panel

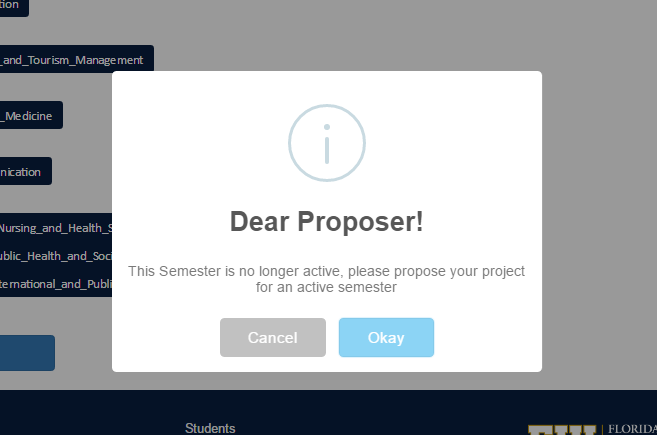


From there, scroll down to the “Lock/Unlock Semester” Panel



Select an option in the “Term” menu, and an option from the “change Semester’s status to” menu. Then click the “Change” button underneath. You have now locked a semester.To test if the lock is working first, try to propose a new project for that locked Semester. In this example we’ve Locked “Summer 2017”.

Once you attempt to propose the project during the now locked Semester, the following message should appear after you press “Submit”



Projects that belonged to a newly locked Semester will also now be locked, you can also confirm the lock by checking any project that should now be locked (refer to Use Story #1101 for more details).

# References

Github

StackOverflow

Youtube

Wikipedia

JavaScript

Google Docs