*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Final Deliverable

Project Title: Professional Program Management System

**Team Members:** Joana Fraga, Jose Casanova

**Product Owner(s)**: Nagarajan Prabakar

**Mentor(s)**: Nagarajan Prabakar

**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 Professional Program Management System, a web portal designed for the use of those FIU students taking part in the professional program. This document is comprised of multiple sections, each one detailing the system through its design, plan, and user stories. Proper understanding of this document will give substantial insight into the development of this application and all of its internal components.*

**Table of Contents**

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

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

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

**User Stories**

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

Pending User Stories …………………………………………………………………...………………………………..….. 11

**Project Plan**

Hardware and Software Resources ……………………………………………………...……………………….… 12

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

*Sprint 1*  …………………………………………………………………………………………………………………………... 13

*Sprint 2*  …………………………………………………………………………………………………………………………... 14

*Sprint 3*  …………………………………………………………………………………………………………………………... 14

*Sprint 4*  …………………………………………………………………………………………………………………………... 15

*Sprint 5*  …………………………………………………………………………………………………………………………... 16

**System Design**

Architectural Patterns ………………………………………………………………………………………………….. 16

System and Subsystem Decomposition …………………………………………………………………………….… 16

Deployment Diagram …………………………………………………………………………………………………….…... 18

Design Patterns ………………………………………………………………………………………………………….….... 18

**System Validation**  …………………………………………………………………………………………………………………….19

**Glossary**  ………………………………………………………………………………………………………………………………….27

**Appendix**  ………………………………………………………………………………………………………………………………….28

Appendix A - UML Diagrams ………………………………………………………………………………………………. 28

Appendix B - User Interface Design …………………………………………………………………………….…... 50

Appendix C - Sprint Review Reports …………………………………………………………………………...…… 58

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents …………………………………………………………………….…………… 61

**References** ……………………………………………………………………………………………………………………...………...61

# 

# Introduction

FIU is known to possess a variety of students, each coming from different walks of life and each with their own schedules and special needs. This is no more apparent than in those students taking classes as they partake in their daily full-time occupations. These professionally-minded students face greater struggles in their studies than those who do not possess occupations, not only because of the less amount of time they are able to allocate to their studies, but also because of the difficulty in finding a suitable time in which to take classes.

This document explores a relevant solution to this problem and is separated into many parts, each one detailing a particular aspect of the implementation. This section deals with the existing way this issue is resolved as well as the rationale behind a more enhanced solution. The second section contains all user stories that were and still are to be implemented. The third section holds the project requirements as well as the plan for each sprint. The fourth section details the various designs used in the creation of the new system. The fifth section discusses the tools and methods used for system validation. The sixth section contains a glossary of important terms and acronyms used throughout this project. The seventh section is comprised of various appendices, and finally, the eighth and last section lists all important references made in this document.

## 

## Current System

A system of introducing individuals about the professional program currently exists in the form of a webpage (<https://msit.fiu.edu/>). In essence, this site serves as an introduction to the system only, and does not allow users functionality beyond information-gathering. Through this system, users are able to view the current schedule and the classes offered therein, learn about how tuition is paid, and read about the program in general. Contact information is provided such as an email and phone number. Each component of the site is information-based; the handling of every aspect requiring information to be kept is handled through different resources, such as tuition through mailing, requests and questions through email, course selection through the MyFIU web portal, etc.

## Purpose of New System

The idea behind the **PPMS** is to provide a centralized location where those interested might not only be able to learn about the professional program itself, but also to be able to interact with the system as a whole without requiring the use of so many different facets in order to use the system properly. The Professional Program Management System acts as a hub supplementing other aspects of the process by providing in-house tools such as a means for students to generate and resolve requests to administrators, or for students to review their schedules, or for administrators to be able to assist students through other means. It enhances the information-based system of old by allowing those using the new system to make changes to it.Thus, the system is now volatile, and affected by its users.

# User Stories

The following section provides the detailed user stories that were implemented in this iteration of the Professional Program Management System 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

**User Story 663: Initial backend setup/home page prototype**

* Description: As an online user, I would like to be able to connect to the application remotely and access to the home page. Home page will be just a prototype.

**Acceptance Criteria**

* The system should be able to load the home page prototype remotely.
* The goal for this user story is to test the connections string

**User Story 669 : Create an Account**

* Description: As a prospecting student I want to be able to create an account with my own username and password so that I may access the Professional Program Management System.

**Acceptance Criteria**

* System needs to validate uniqueness of username
* System needs to validate password with a minimum of 8 characters

**User Story 770: View all available schedules for Q&A sessions**

* Description: As a member of this portal I want to view the current schedules for the Q&A sessions so that I can decide if I would like to attend.

**Acceptance Criteria**

* System needs to validate that user is login
* All available schedules should be displayed in a table format

**User Story 771: Administrator Should be Able to Schedule a Time for Q&A Session**

* Description: As an administrator of this portal I want to be able to schedule times for the Q&A sessions so that user can see all available sessions.

**Acceptance Criteria**

* System should validate data entered by the administrator user
* Only administrators are allowed to schedule new Q&A sessions

**User Story 672: Login and Logout**

* Description: As a potential user of this portal I want to be able to login and logout of this system seamlessly so that i can access the PPMS website securely.

**Acceptance Criteria**

* System should validate and authenticate username and password
* System should open different views for different types of users.

**User Story 673: Reset Password**

* Description: As a user of this portal I want to be able to reset my password so that I can access my account even when I forgot my credentials.

**Acceptance Criteria**

* System should send reset password link to the user’s email address

**User Story 774: Submit a Ticket**

* Description: As a user of this portal I want to be able to submit a ticket within the system so that I can have a problem/concern resolved.

**Acceptance Criteria**

* User should be able to add a description to the ticket
* Ticket should be put in the ticket pool with a time-stamp

**User Story 775: View My Tickets**

* Description: As a user of this portal I want to be able to view all tickets I have submitted so that I can check their status.

**Acceptance Criteria**

* User should be able to see any ticket he/her submitted
* User should be logged in to be able to see her/his tickets.

## 

**User Story 776: Change Ticket Status/Reply to a Ticket**

* Description: As an administrator of this portal I want to be able to change the status of any tickets so I can close and/or resolve a ticket.

**Acceptance Criteria**

* Only administrator are allowed to change the status of a ticket
* Administrators should be logged in to change the status of a ticket

**User Story 677: View All Tickets.**

* Description: As an administrator I want to be able to see all tickets submitted by any user so that I can resolve any ticket.

**Acceptance Criteria**

* Only administrator can see the pool of all tickets
* Administrators should be logged in to see the tickets

**User Story 678: View All Tickets**

* Description: As an administrator I want to be able to see all tickets submitted by any user so that I can resolve any ticket.

**Acceptance Criteria**

* Only administrator can see the pool of all tickets
* Administrators should be logged in to see the tickets

## Pending User Stories

All user stories for this semester were successfully completed.

# 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:

Minimum Hardware Requirements:

* 1 GHz processor
* 256 MB free of RAM
* Internet connection
* A server in which to house the files (optional)

Minimum Internet Browser Requirements (only one):

* Internet Explorer 8
* Safari 4
* Firefox 4.1
* Chrome 50 (recommended)

Minimum Software Requirements:

* MySQL version 14.14 distribution 5.5.14
* PHP version 5.5.9
* Angular 2
* Visual Studio 2013 (for running HTML/Angular code locally)
* XAMPP 5.5.9 (if running server code locally)
* sendmail (if using Unix server)

## 

## Sprints Plan

### Sprint 1

After discussion, the velocity of the team were estimated to be 40.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* User Story 663: Initial backend setup/home page prototype

The team members indicated their willingness to work on the following user stories.

* Joana Fraga
* User Story 663: Initial backend setup/home page prototype.

Server setup.

* Jose Casanova
* User Story 663: Initial backend setup/home page prototype.

### Sprint 2

After discussion, the velocity of the team were estimated to be 48.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* User Story 669: Create an Account
* User Story 672: Login and Logout
* User Story 673: Reset Password

The team members indicated their willingness to work on the following user stories.

* Joana Fraga
* User Story 669: Create an Account
* Jose Casanova
* User Story 672: Login and Logout
* User Story 673: Reset Password

### Sprint 3

After discussion, the velocity of the team were estimated to be 56.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* User Story 670: View all available schedules for Q&A sessions
* User Story 673: Reset Password
* User Story 674: Submit a ticket

The team members indicated their willingness to work on the following user stories.

* Joana Fraga
* User Story 670: View all available schedules for Q&A sessions
* User Story 674: Submit a ticket
* Jose Casanova
* User Story 673: Reset Password

### Sprint 4

After discussion, the velocity of the team were estimated to be 72.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

* User Story 671: Administrator should be able to schedule a time for Q&A session.
* User Story 675:View my tickets.
* User Story 676: Change ticket status.
* User Story 677:View all tickets.
* User Story 678: Cancel/Close a ticket.

The team members indicated their willingness to work on the following user stories.

* Joana Fraga
* User Story 671: Administrator should be able to schedule a time for Q&A session.
* User Story 675:View my tickets.
* Jose Casanova
* User Story 676: Change ticket status.
* User Story 677:View all tickets.
* User Story 678: Cancel/Close a ticket.

# 

# 

# 

# 

# 

# 

# 

# 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 architectural patterns most well-suited to represent this system are: **client/server** as the primary and **repository** as the secondary. Both architectural patterns commit to the system’s strengths of maintaining a central database as the main tool in which all requests that govern user information must go through.

The client/server architectural style relates to the database being held server-side where the user is to make requests or update through their client, or personal devices in this case. As the users interact with the system, information is queried from the central database and sent client-side to be visually represented. Likewise, when the user submits a form that requires an update to the database, the information is passed through to the database, and once stored successfully, the user’s page is reloaded to showcase the changes they provided to the database, if necessary.

The repository architecture makes a decent fit for this system due to the system’s innate independence with regards to its subsystems. Subsystems rarely interact with one another, instead opting to connect with the central database to send requests or receive data. This central database acts as the main controller for all the subsystems, as the responsibility falls on it to be able to provide information and update itself in a timely manner so that all user requests are handled without delay or bottlenecking.

## System and Subsystem Decomposition

This system can be divided up into **seven** subsystems. Each of these subsystems are relatively independent from one another and act only in tandem with the database subsystem itself. Information between subsystems is shared if necessary by requesting it from the database. What follows is a list of the different subsystems that impact the system.

* **Repository**: The central subsystem with which all others interact. Has the ability to make calls to the central database and to return or update it, passing all results to all others subsystems as they are requested.
* **Registration**: The subsystem responsible for handling user requests to register. This subsystem obtains information from the user and sends it to the central repository for storage. It is also responsible for validating all user input to make sure it complies with all registration standards and security checks.
* **Login**: The subsystem responsible for handling user requests to log into the system and maintain track of their sessions, as well as clearing all instances of their session once they log out. This subsystem obtains information from the user and requests information from the central repository for its validity. Once it obtains what it needs from the repository, it ensures the information entered is accurate and indicative of a user in the system and logs them in or provides an error message accordingly.
* **Password Reset**: The subsystem responsible for for submitting requests to reset a user’s password. It connects to the central database both to obtain the credentials of the information entered by the user, and to update it once all proper validation and verification schemes have been completed. It is the responsibility of this subsystem to check all user input and ensure that it is valid and correct.
* **Information**: This subsystem deals with obtaining information about the program, such as the dates for the Information Sessions, or a list of a student’s registered courses, or simply providing a summary of the program itself. It is a simple subsystem that need only communicate to the database in order to retrieve the data regarding the data above and showcase them to the user.
* **Ticketing**: This subsystem is comprised of both the ticketing view for both standard users and administrators as well as the components necessary to be able to create or update tickets. This subsystem makes calls to the central repository in order to obtain information about all existing tickets, both on a user and system level, depending on the actor, and to send information back to the repository representing any updates or ticket creations. It falls to this system to ensure all information entered on the tickets is valid, and that the requests safely make it through to the database.
* **Administrative Tools**: This subsystem is an umbrella subsystem that can be further split up into very simple subsystems but is kept together for the purposes of similarity between each of its components. It involves all tasks administrators are capable of doing such as obtaining a list of every student in the system, or adding courses or Information Session dates to the schedule. It interfaces with the repository, both to receive information and well as to send updates, and much like all other subsystems it is responsible for validating the information that is being sent.

## Deployment Diagram

Please see Appendix A Figure 35.

## Design Patterns

Three design patterns were utilized for the creation of this system, namely **Singleton**, **Abstract Factory**, and **Dependency Injection**. The Singleton design pattern deals with controlling the number of instances each class can have, as every class was designed to be a controller, passing in input provided by the user or obtained in some other way, and sending the result to the database once finished. No class holds any data once finished sending it to the database.This allows for safer implementation as only a single object will be able to manipulate the database at any given time. Abstract Factory is another notable design pattern, as classes such as Register or Login act as factories for their own uses, and so users of the system can fulfil their requests modularly, with no other classes being able to perform the same level of responsibility. Finally, Dependency Injection is useful for enforcing independant design from the client, a fact that is further enforced by the small amount of coupling between each of the subsystems.

# System Validation

**User Story 663**

**Unit Test**

* Test case ID: 679
* Description/Summary of Test: User can access homepage
* Pre-condition: User has internet access, user open the browser
* Expected Results:Homepage is displayed
* Actual Result: Homepage is displayed
* Status (Fail/Pass):Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login (Link is not yet established)
* Homepage with AboutUs (Link is not yet established)
* Homepage with ContactUs (Link is not yet established)

**User Story 669**

**Unit Test**

* Test case ID: 681
* Description/Summary of Test: User shall be able to successfully create an account
* Pre-condition: User needs internet access, navigate to the Home Page, and click “Register”
* Expected Results: User’s Main Page shall be displayed
* Actual Result: User’s Main Page is displayed
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Register option
* User is able to Register using username and password (More data input must be required by the user later)
* User’s info is correctly stored on the database

**User Story 672**

**Unit Test**

* Test case ID: 682
* Description/Summary of Test: User shall be able to successfully login and logout
* Pre-condition: User needs internet access, navigate to the Home Page, and click “Login”, and enter ‘username’ and ‘password’.
* Expected Results: User’s Main Page shall be displayed
* Actual Result: User’s Main Page is displayed
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option
* User is able to Login using username and password

**User Story 673**

**Unit Test**

* Test case ID: 685
* Description/Summary of Test: User shall be able to successfully reset the password
* Pre-condition: User needs internet access, navigate to the Home Page, and click on “Reset Password”. User shall be able to enter New Password information, and click “Submit”.
* Expected Results: User’s Main Page shall be displayed
* Actual Result: User’s Main Page is displayed
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Reset Password option.
* User is able to Enter a new password.
* User’s info is correctly stored on the database.
* User is redirected to User’s Home Page.

**User Story 670**

**Unit Test**

* Test case ID: 686
* Description/Summary of Test: User shall be able to successfully see all available Q&A Sessions.
* Pre-condition: User needs internet access, navigate to the Home Page. User needs to login with a valid username and password, and click on “Q&A Sessions”.
* Expected Results: User shall be able to see all available Q&A Sessions.
* Actual Result: User is able to see all available Q&A Sessions.
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. User is able to login.
* User Home page with Q&A Sessions option.
* User is able to see all available Q&A Sessions.

**User Story 674**

**Unit Test**

* Test case ID: 688
* Description/Summary of Test: User shall be able to successfully submit a ticket.
* Pre-condition: User needs internet access, navigate to the Home Page and login with a valid username and password. Then user needs to click on “My Tickets” then on “Submit Ticket”. User is prompted to enter all the required information to be able to submit a ticket. At the end of the process user should be able to click “Submit”.
* Expected Results: User shall be able to submit a ticket.
* Actual Result: User is able to submit a ticket.
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. User is able to login.
* User Home page with My Tickets option.
* My Tickets page with Submit ticket option.
* User is able to Submit a ticket.
* Database successfully stores the new ticket.

**User Story 771**

**Unit Test**

* Test case ID: 699
* Description/Summary of Test: Administrators shall be able to Schedule a Time for Q&A Session.
* Pre-condition: Admin needs internet access, navigate to the Home Page and login with a valid username and password. Admin click on “Q&A Sessions” and a list of available Question and Answer Sessions should be displayed. Admin click on “Add Q&A Sessions” and input the required information.
* Expected Results: Administrators shall be able to Schedule a Time for Q&A Session.
* Actual Result: Administrators is able to Schedule a Time for Q&A Session.
* Status (Fail/Pass): Pass

**Integration Test**

* Admin is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. Admin is able to login.
* Admin Home page with Q&A Sessions option.
* Q&A Page with add Q&A option.
* Admin is able to see all available Q&A Sessions.

**User Story 775**

**Unit Test**

* Test case ID: 693
* Description/Summary of Test: User shall be able to successfully see all previously submitted tickets by her/him.
* Pre-condition: User needs internet access, navigate to the Home Page. User needs to login with a valid username and password, and click on “My Tickets”.
* Expected Results: User shall be able to see all previously submitted tickets by her/him.
* Actual Result: User is able to see all previously submitted tickets by her/him.
* Status (Fail/Pass): Pass.

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. User is able to login.
* User Home page with My Tickets option.
* User is able to see all available My Tickets.

**User Story 676**

**Unit Test**

* Test case ID: 695
* Description/Summary of Test: Admin shall be able to successfully change ticket status or reply to a Ticket
* Pre-condition: Admin needs internet access, navigate to the Home Page and login with a valid username and password. Admin clicks on “All Tickets”. After that Admin clicks on the ticket number. The ticket details should be displayed, then admin can edit the ticket (cancel or resolved), or can reply to the ticket.
* Expected Results: Admin shall be able to successfully change ticket status or reply to a Ticket.
* Actual Result: Admin is able to successfully change ticket status or reply to a Ticket.
* Status (Fail/Pass): Pass

**Integration Test**

* Admin is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. Admin is able to login.
* Admin Home page with All Tickets option.
* Admin is able to see all available All Tickets.
* All Tickets page with Link to update ticket.
* Admin is able to update a ticket.

**User Story 677**

**Unit Test**

* Test case ID: 691
* Description/Summary of Test: Admin shall be able to successfully see all previously submitted tickets.
* Pre-condition: Admin needs internet access, navigate to the Home Page. Admin needs to login with a valid username and password, and click on “All Tickets”.
* Expected Results: Admin shall be able to see all previously submitted tickets by any user.
* Actual Result: Admin is able to see all previously submitted tickets by any user.
* Status (Fail/Pass): Pass

**Integration Test**

* Admin is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. Admin is able to login.
* Admin Home page with All Tickets option.
* Admin is able to see all available All Tickets.

**User Story 678**

**Unit Test**

* Test case ID: 697
* Description/Summary of Test: User shall be able to successfully change ticket status.
* Pre-condition: User needs internet access, navigate to the Home Page and login with a valid username and password. User clicks on “My Tickets”. After that User clicks on the ticket number. The ticket details should be displayed, then user can edit the ticket (cancel or close).
* Expected Results: User shall be able to successfully change ticket status.
* Actual Result: User is able to successfully change ticket status.
* Status (Fail/Pass): Pass

**Integration Test**

* User is able to connect to the server, Homepage prototype is properly displayed.
* Homepage with Login option. User is able to login.
* User Home page with My Tickets option.
* User is able to see all available My Tickets.
* My Tickets page with Link to update ticket.
* User is able to update a ticket.

# 

# Glossary

**Actor**: A user that interacts with the system.

**Class Diagram:** A diagram that illustrates dependencies in terms of properties and behaviors of classes and the relationships among them.

**Deployment Diagram:** Diagram that provides a clear view of how hardware elements map to the system logic.

**Entity Relationship Diagram**: Diagram that expresses the way in which stored data relates.

**Florida International University (FIU)**: MainUniversity facilities located at 11200 SW 8th St. Miami, FL 33199.

**Graphical user Interface (GUI)**: Visual representation of a system that allows users to interact with it.

**Professional Program**: An FIU program specialized for IT/CS students who need to take classes with schedules different from those offered in most normal courses.

**Professional Program Management System (PPMS)**: A centralized web portal for students part of the professional management program.

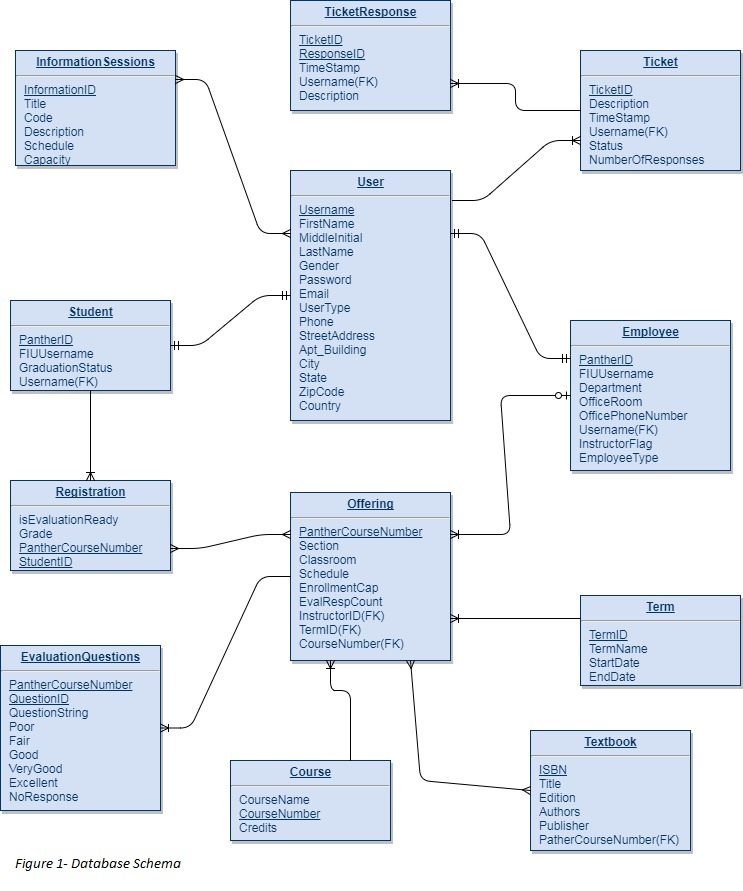
**Sequence Diagram**: A diagram showcasing interactions between objects and their timeline.

**Unified Model Language (UML)**: Unification of modeling rules that provide a generalized graphical representation of a system.

**User Story**: Description of a software from an end-user perspective.

# Appendix

## Appendix A - UML Diagrams

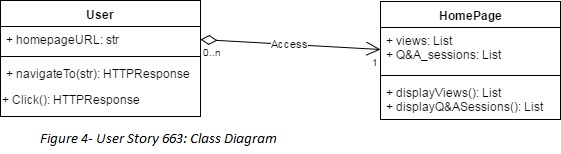


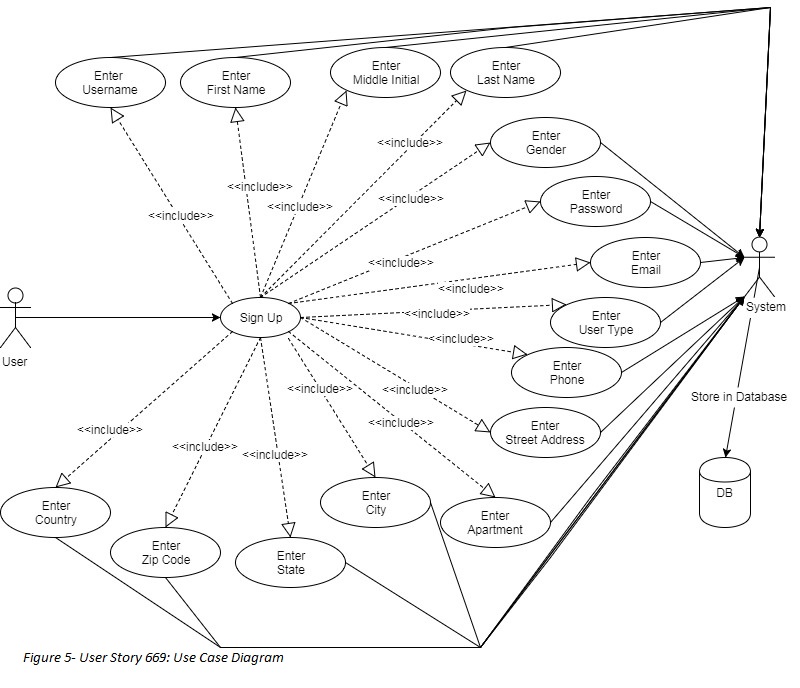
## 

## 

## 

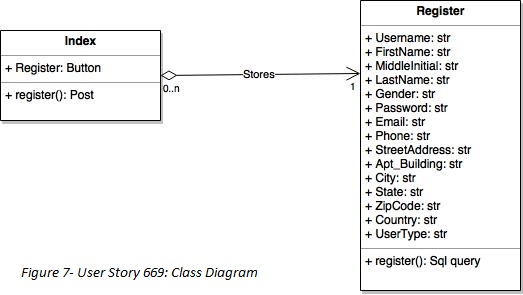
## 

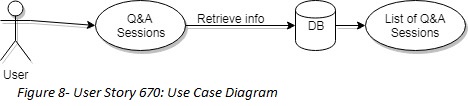


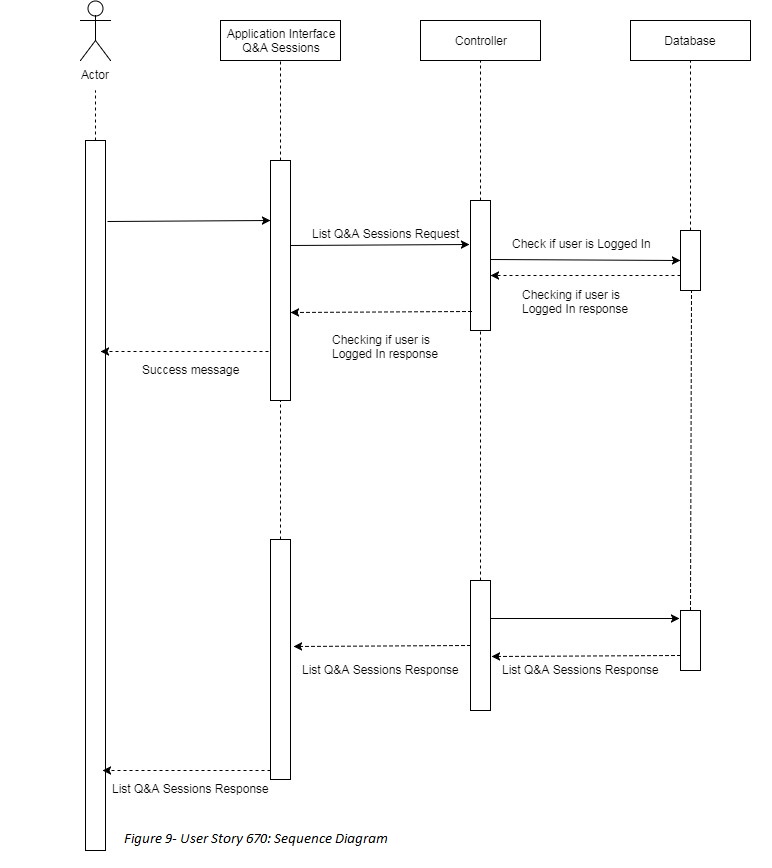


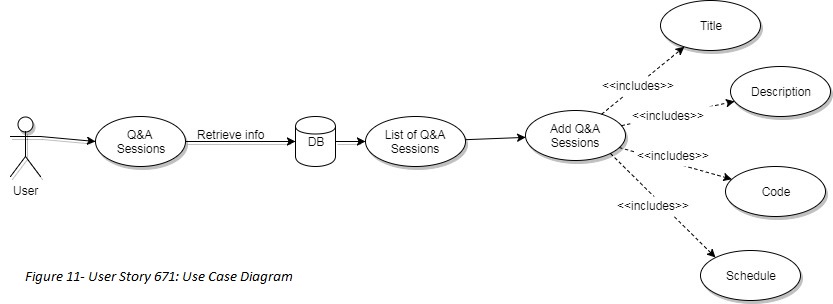
## 

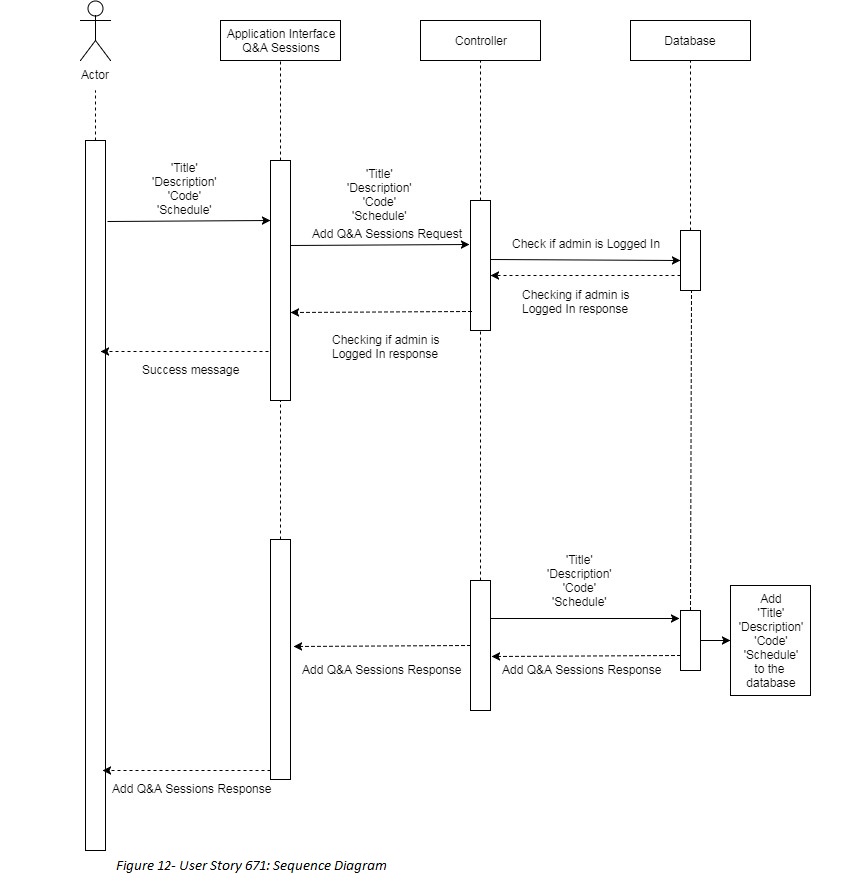
## 

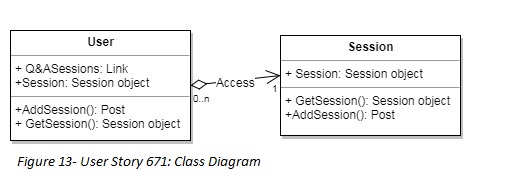


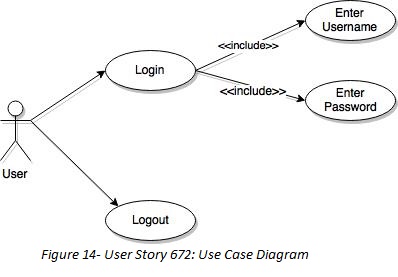


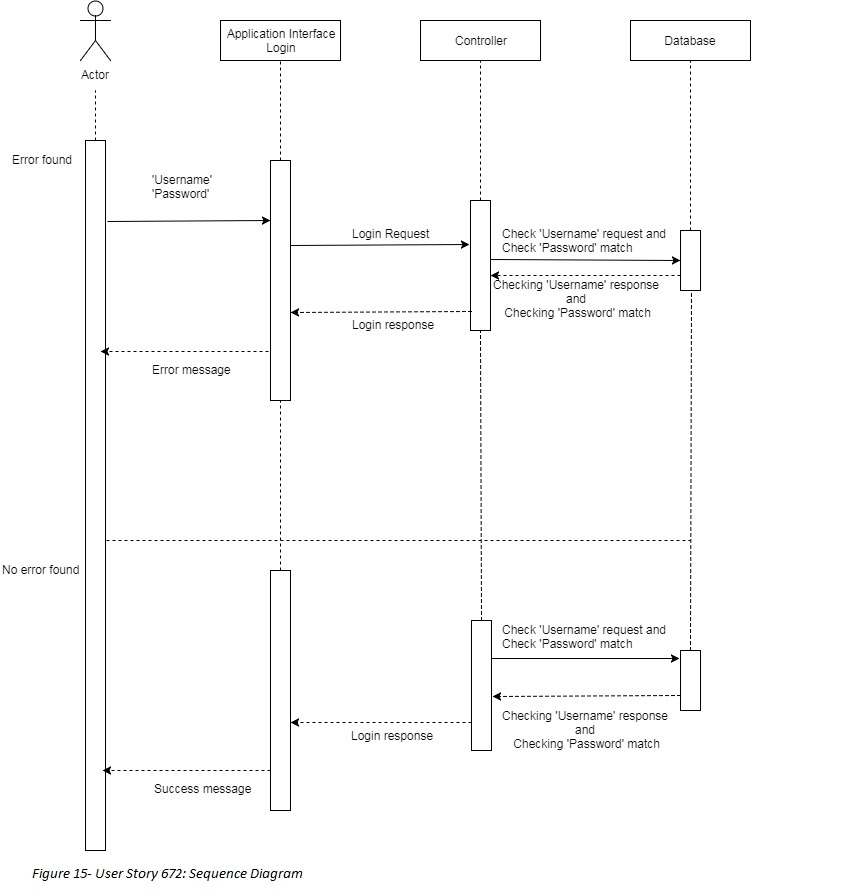


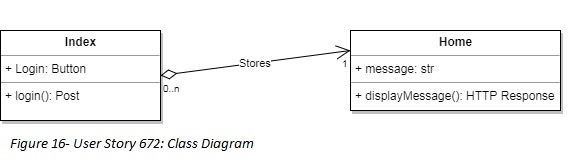


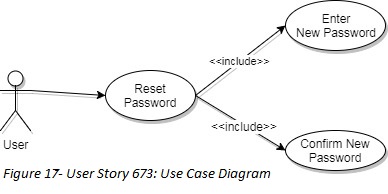


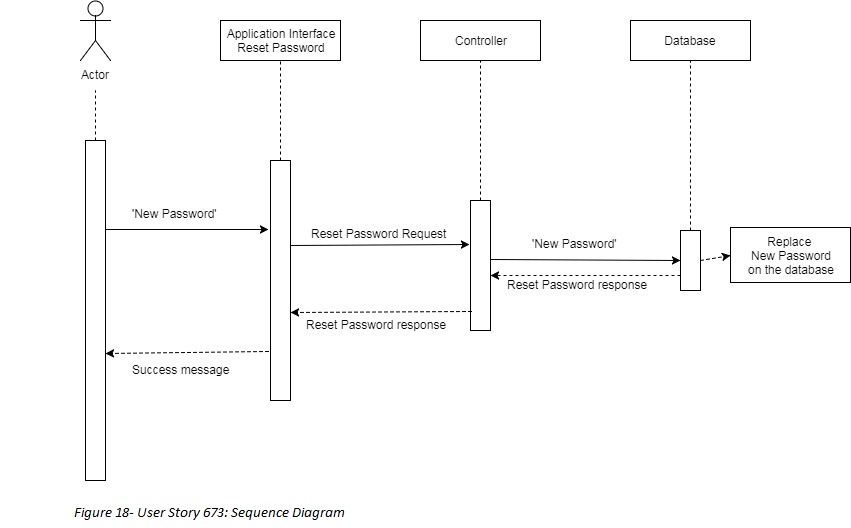
.

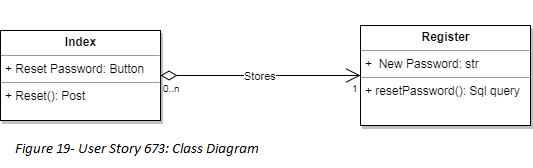


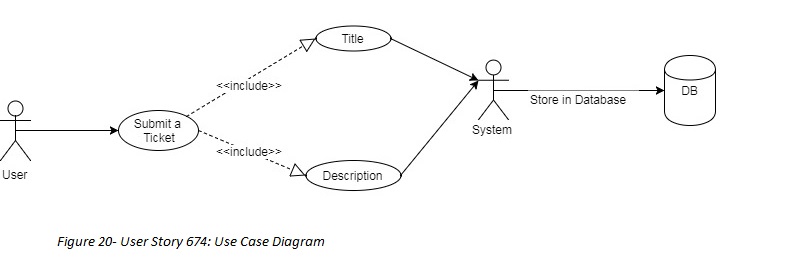


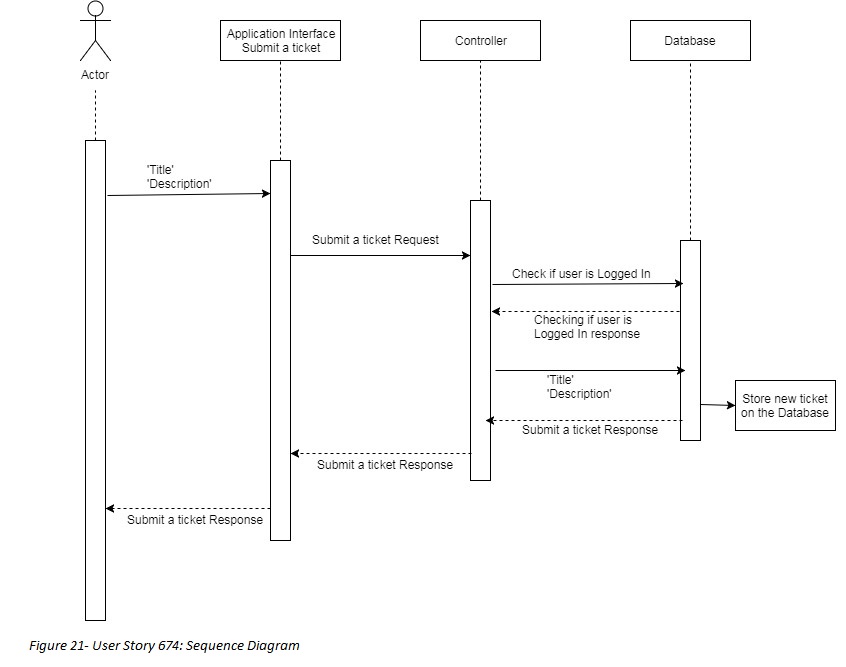


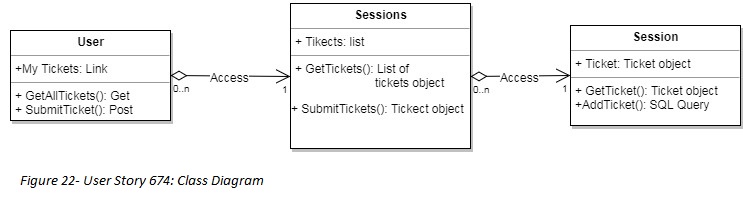


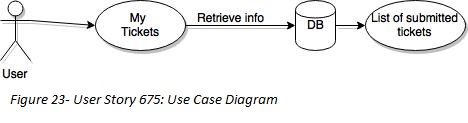


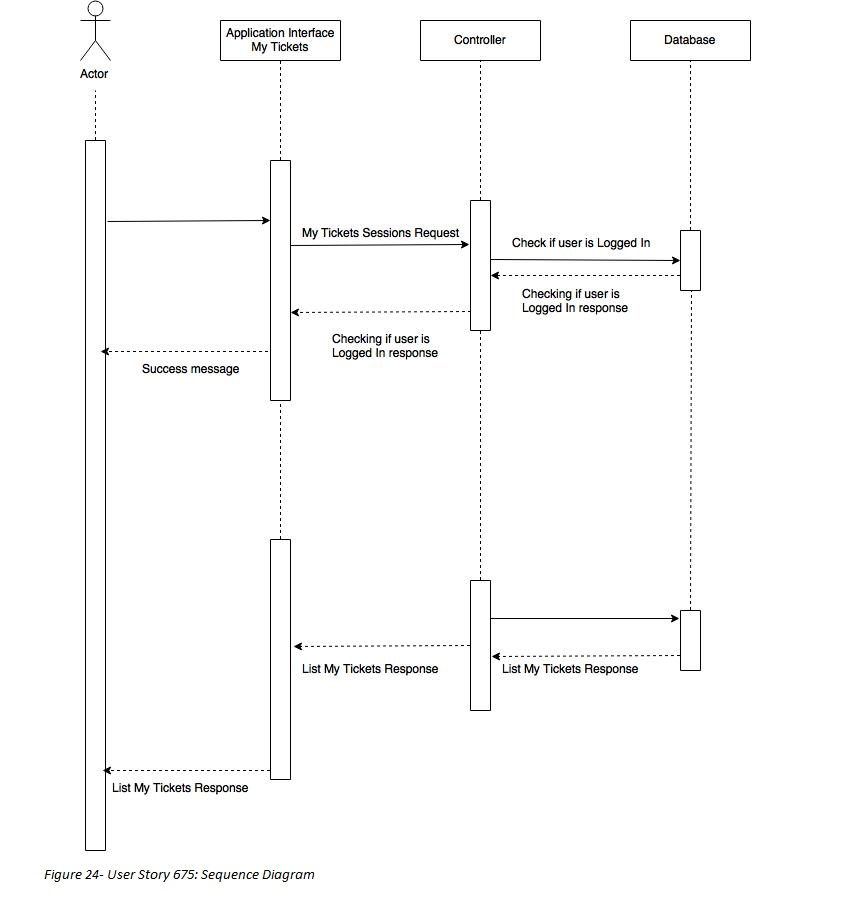


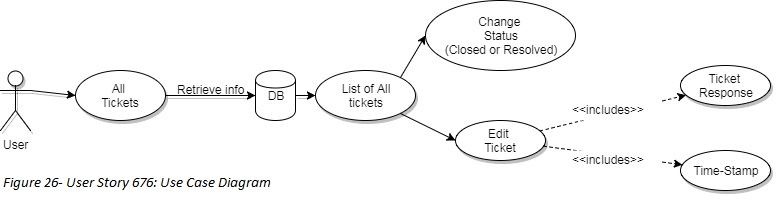
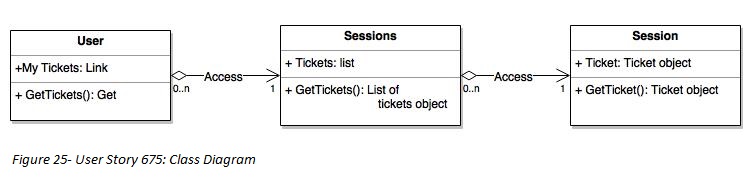


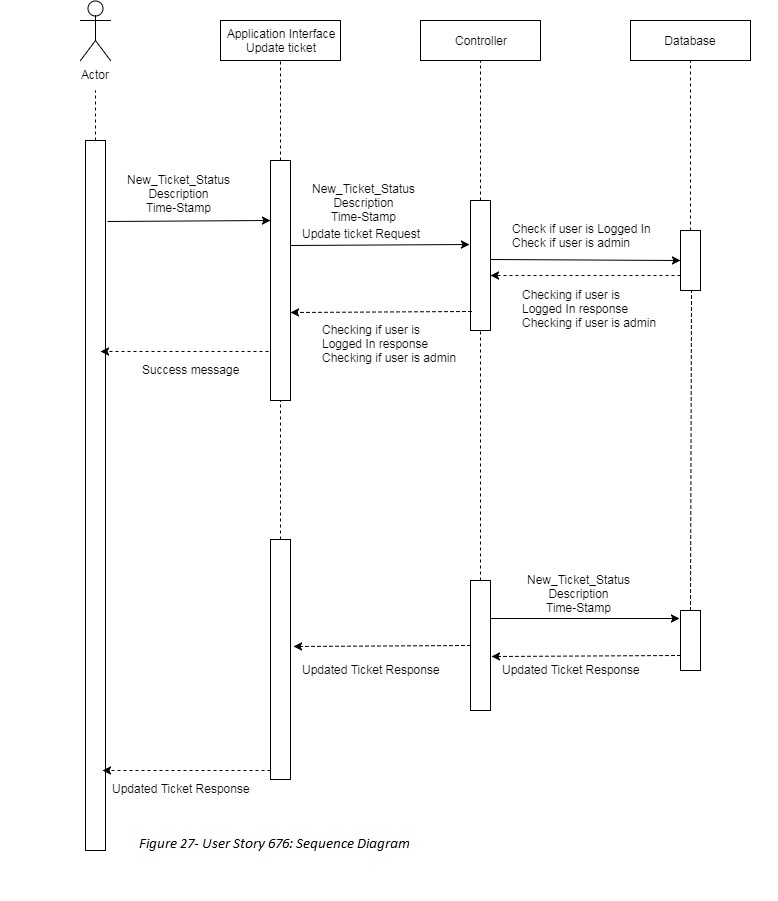


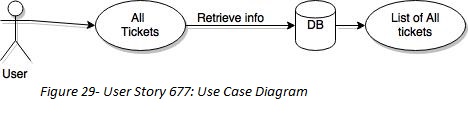
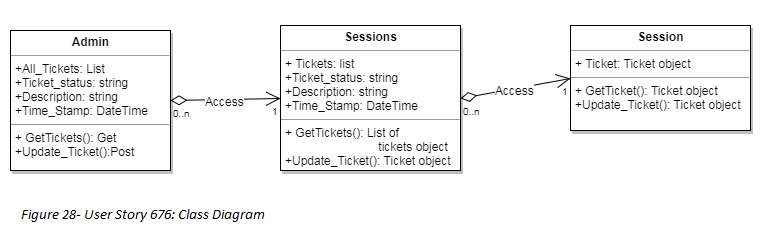


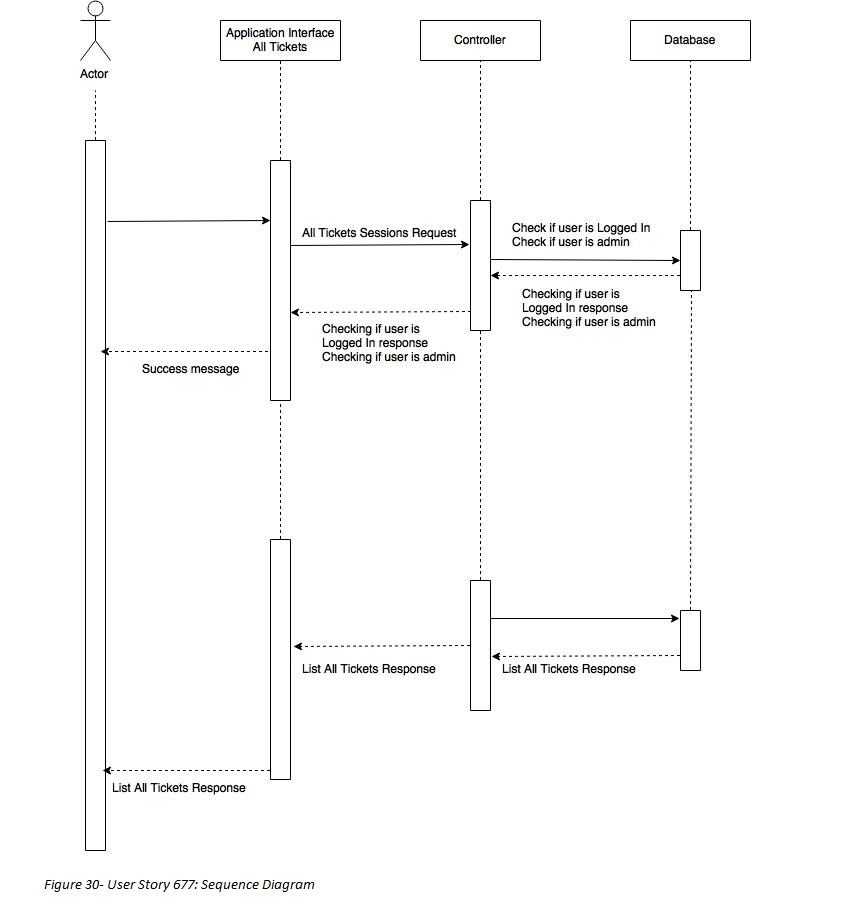


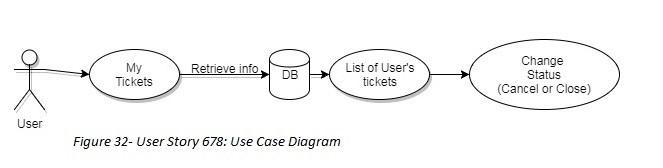
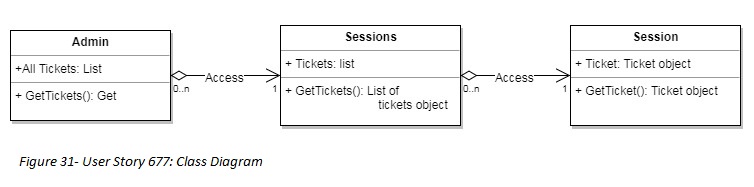


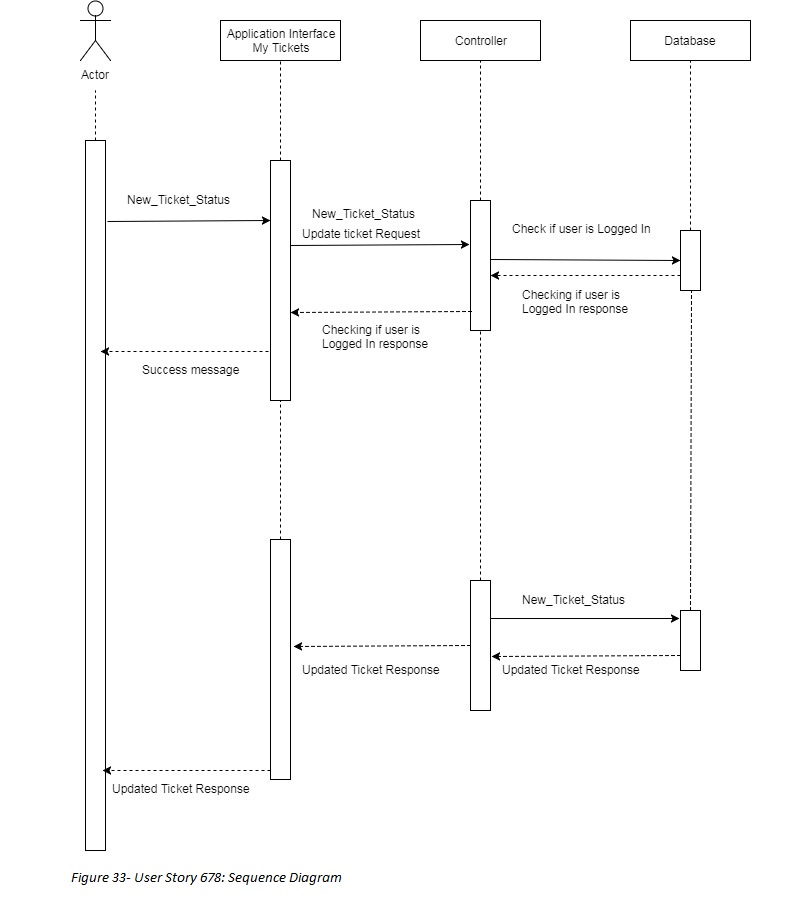


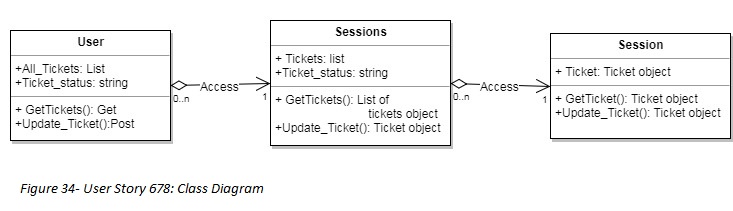












## deployment.jpg

## 

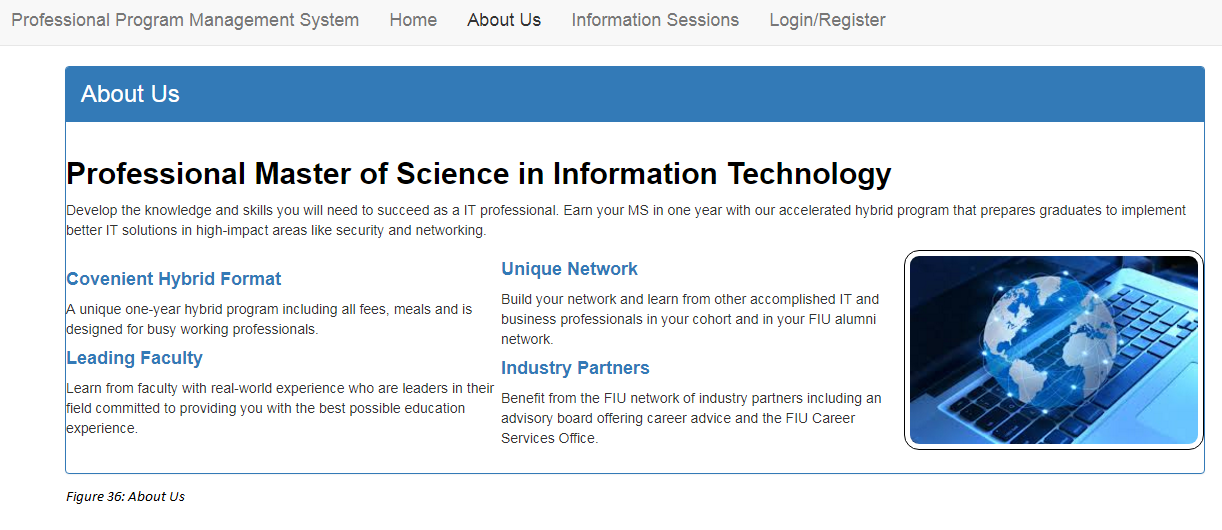
## 

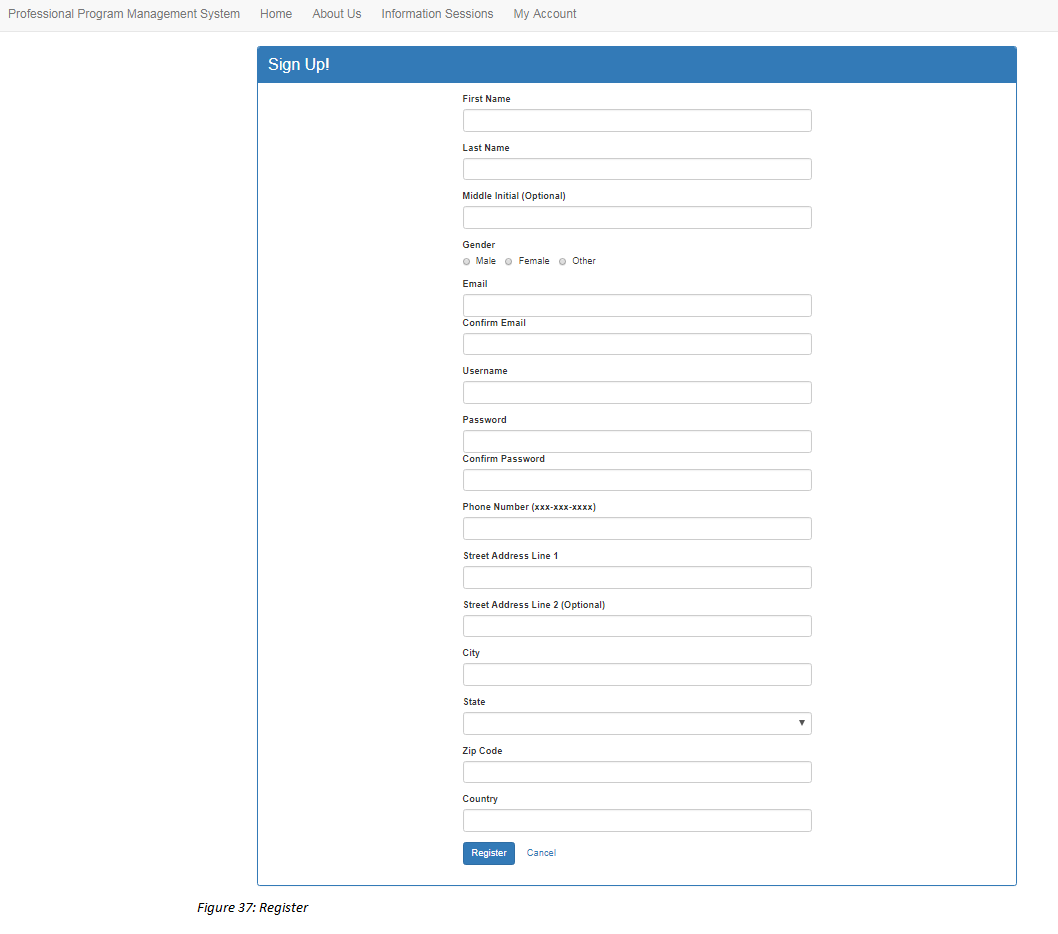
## 

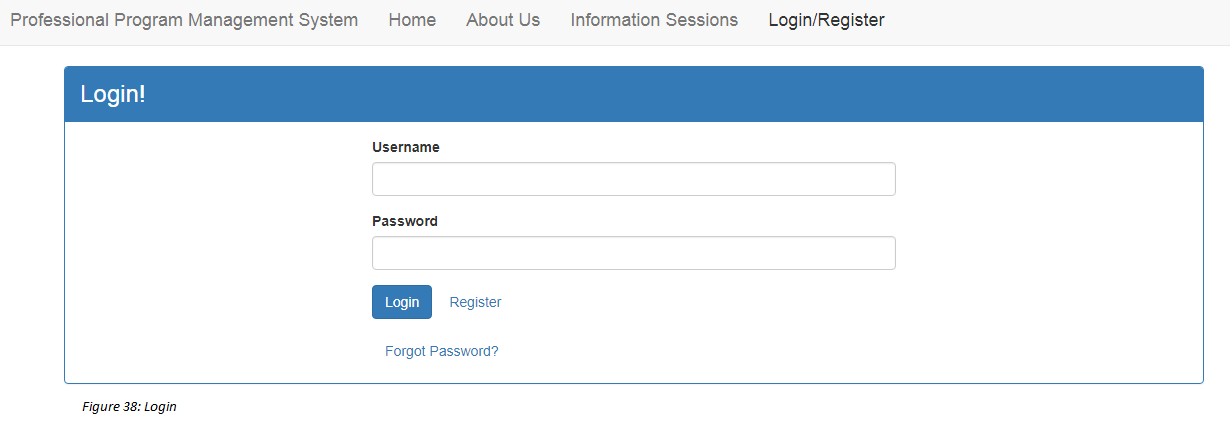
## 

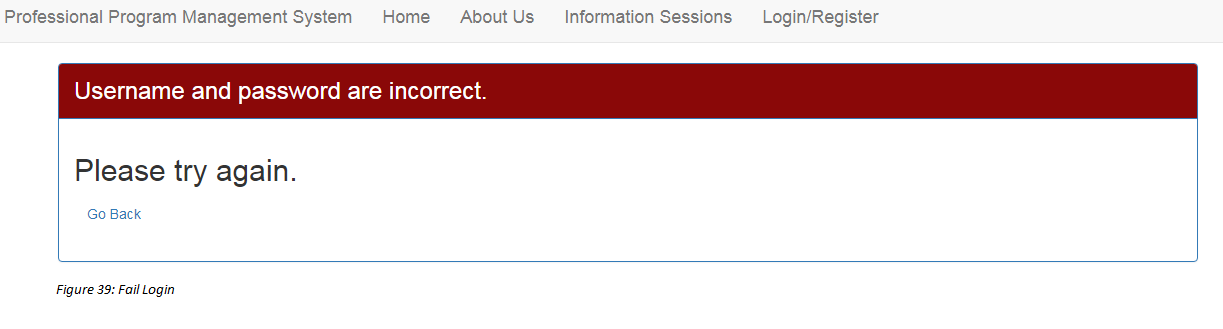
## Appendix B - User Interface Design

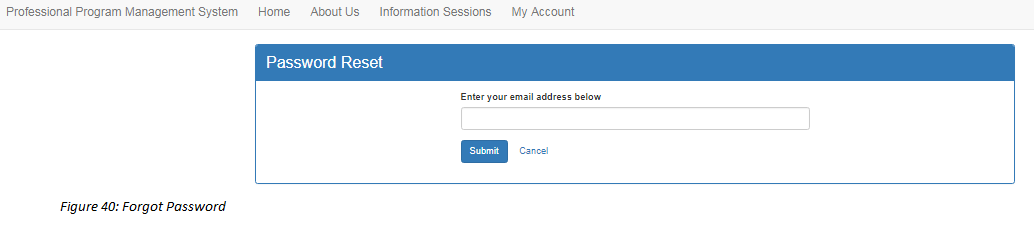
## 

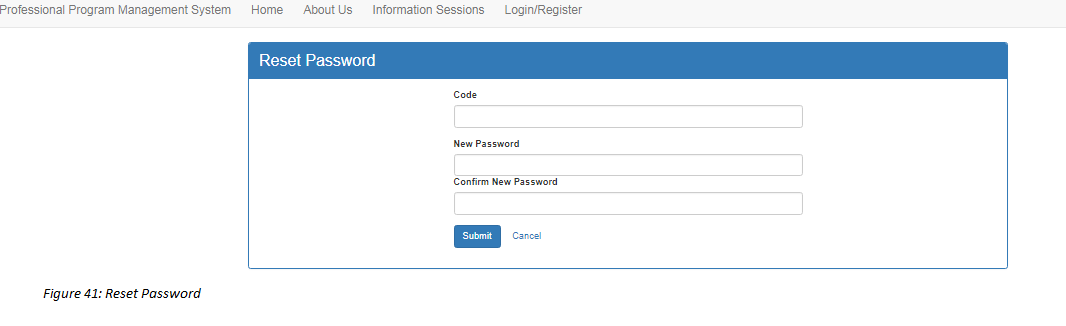


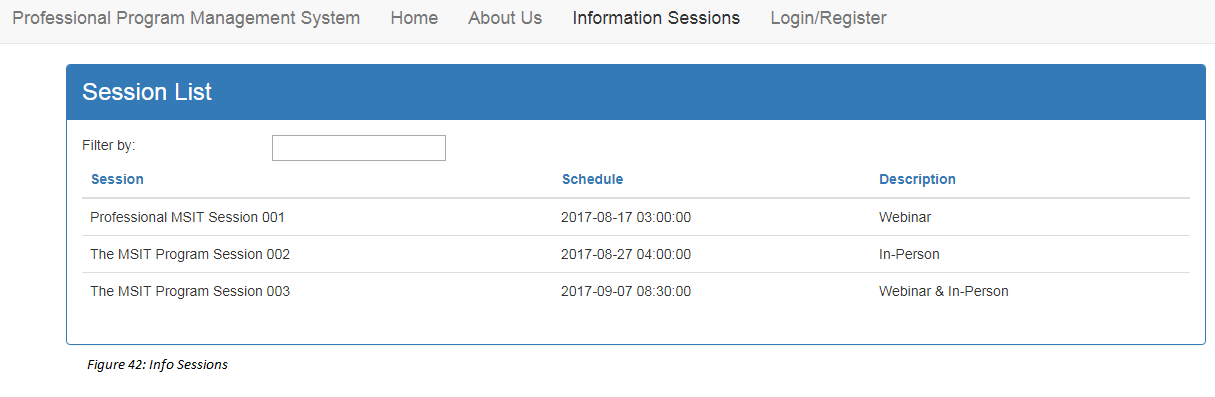


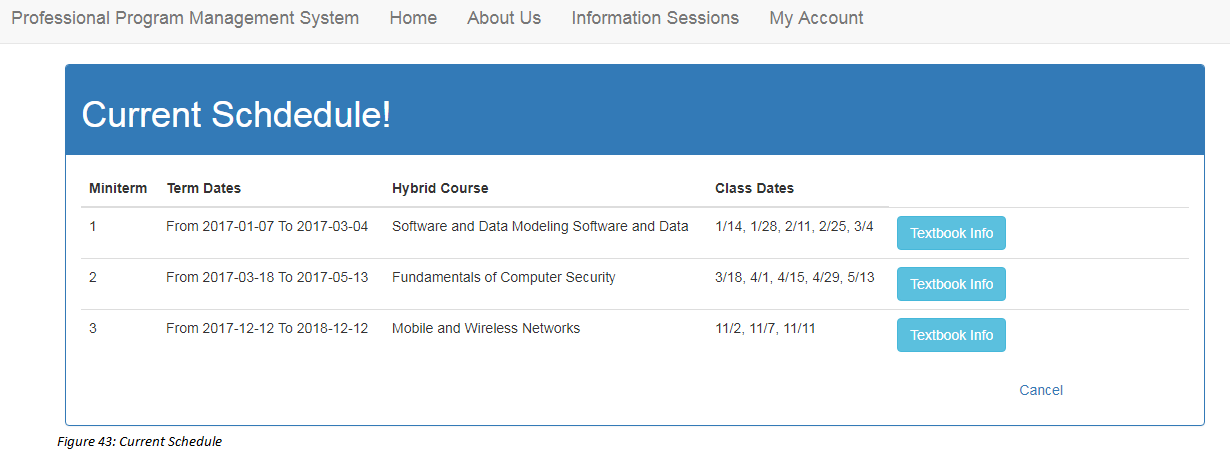


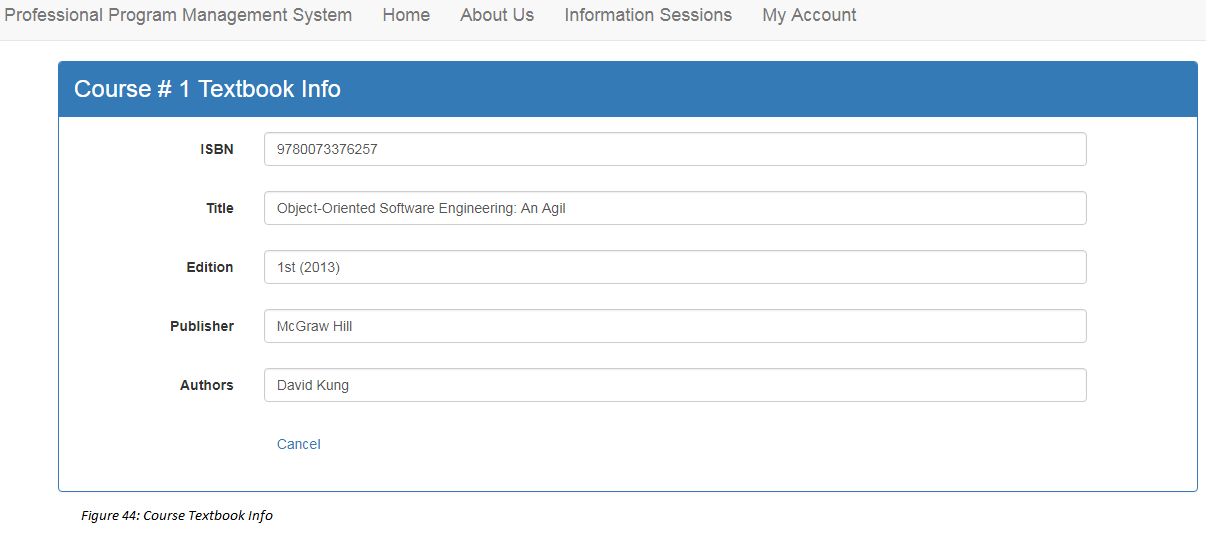


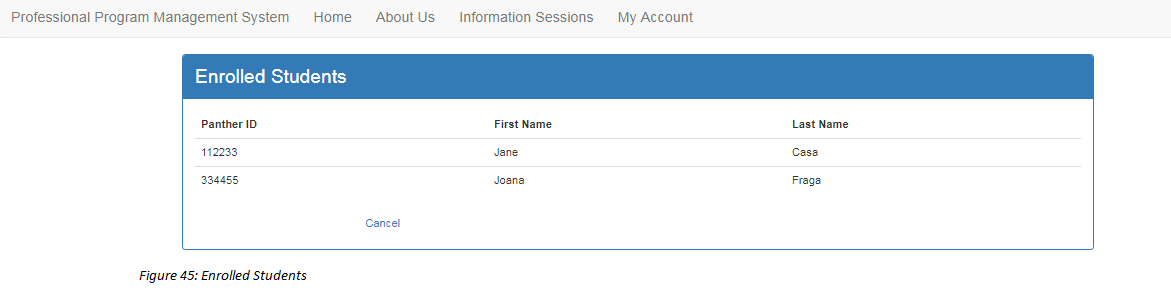


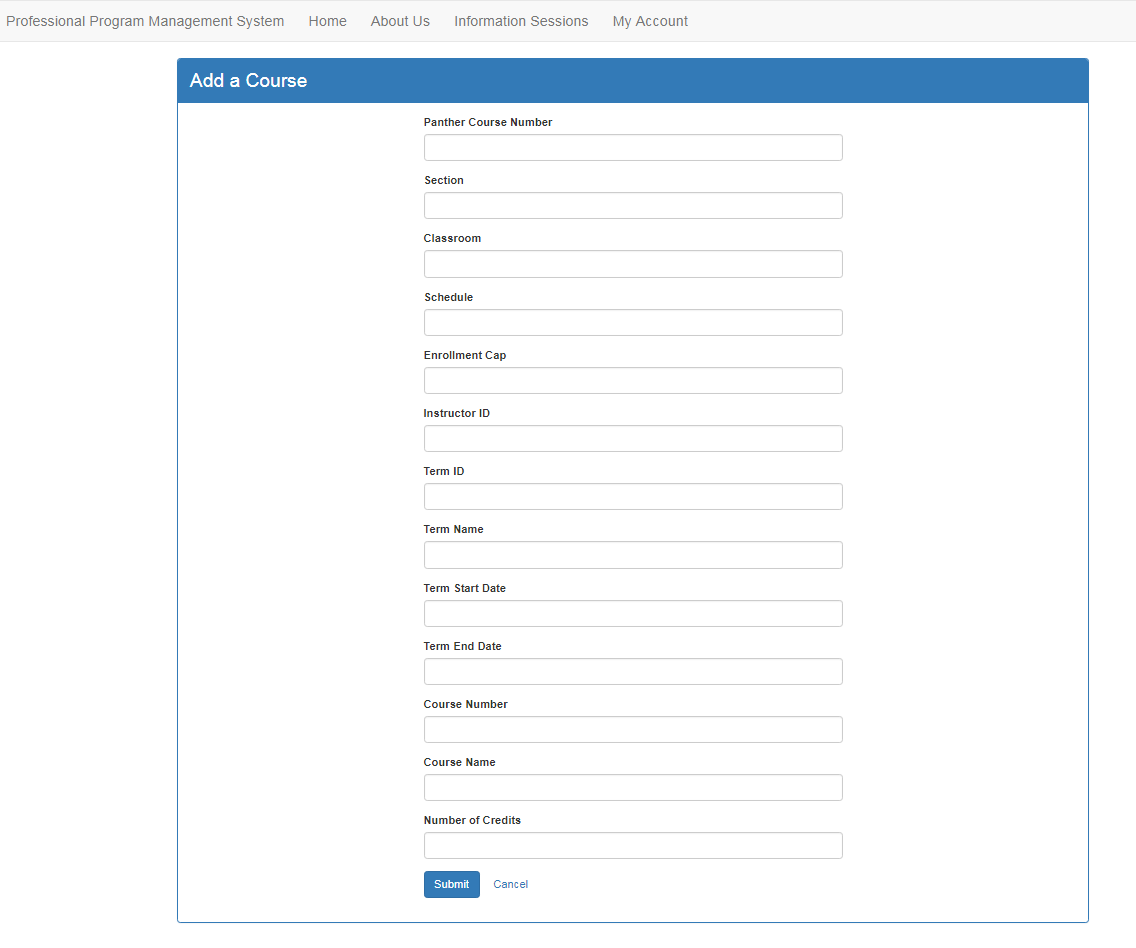


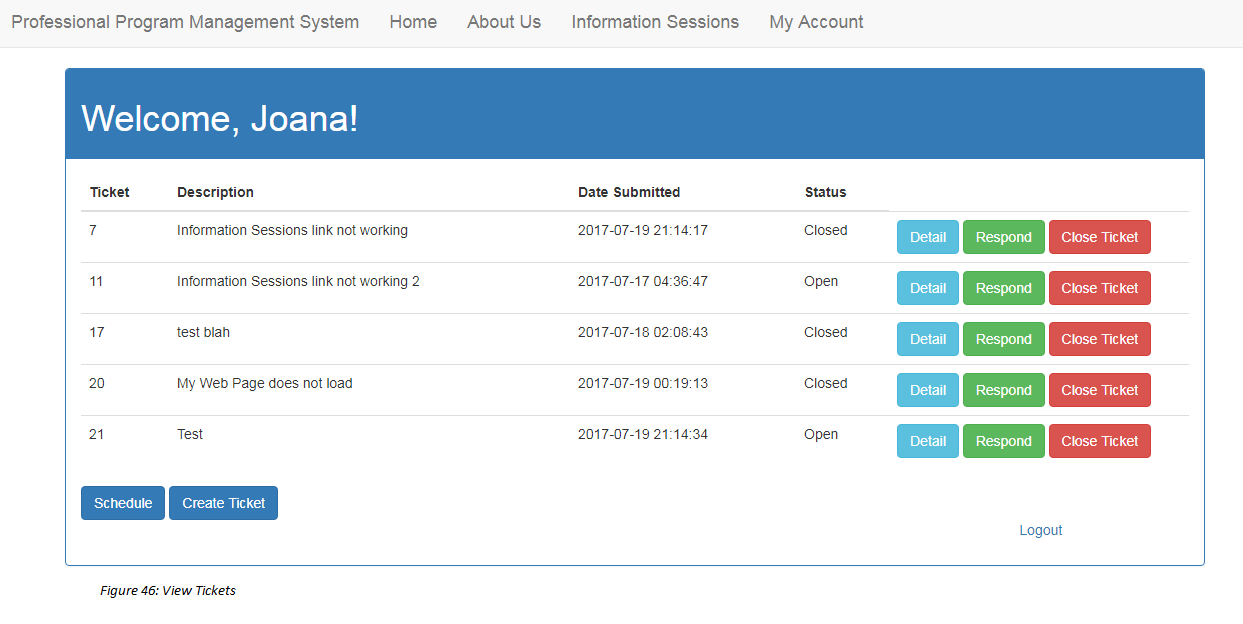


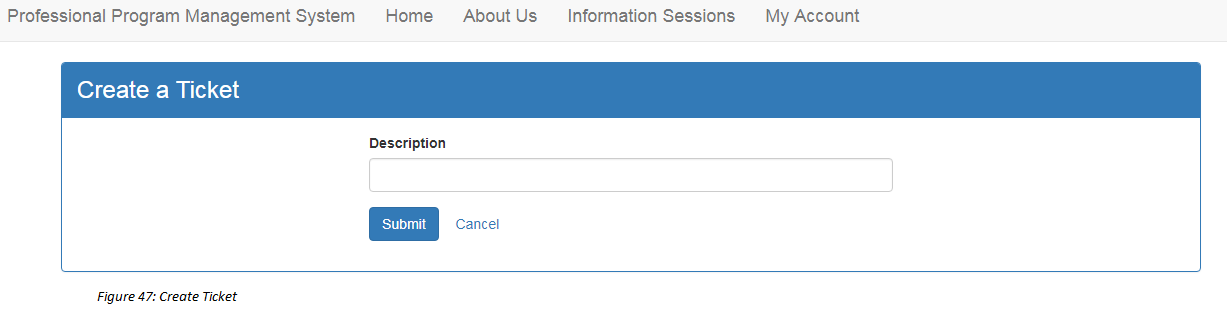


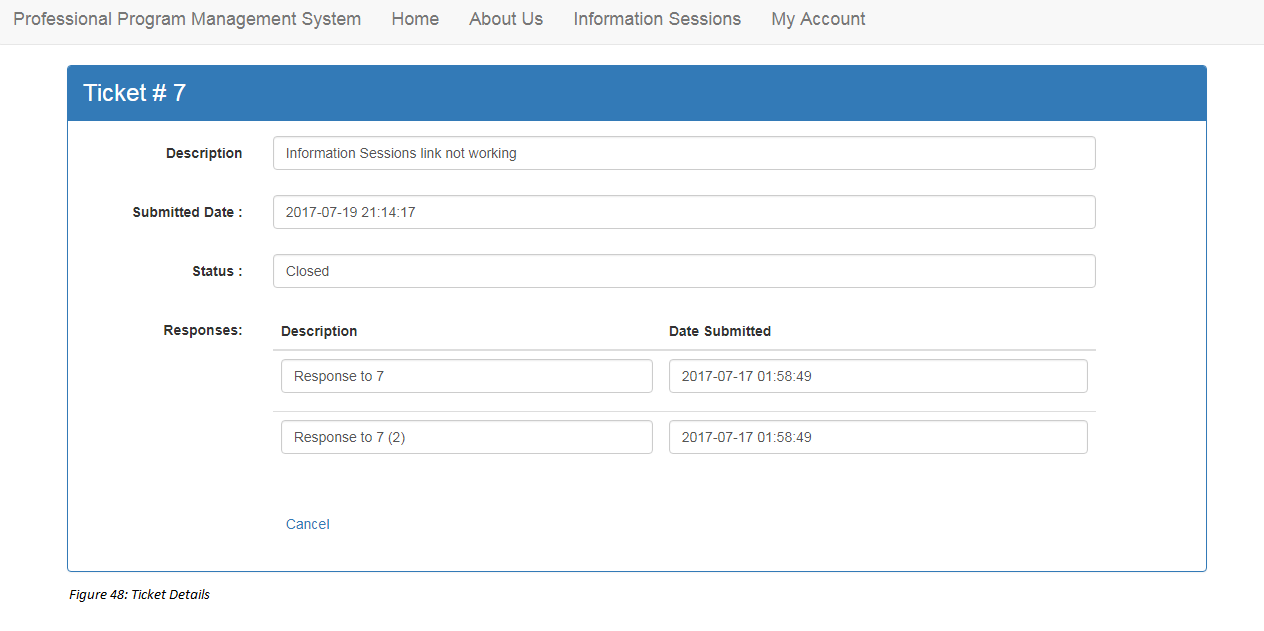


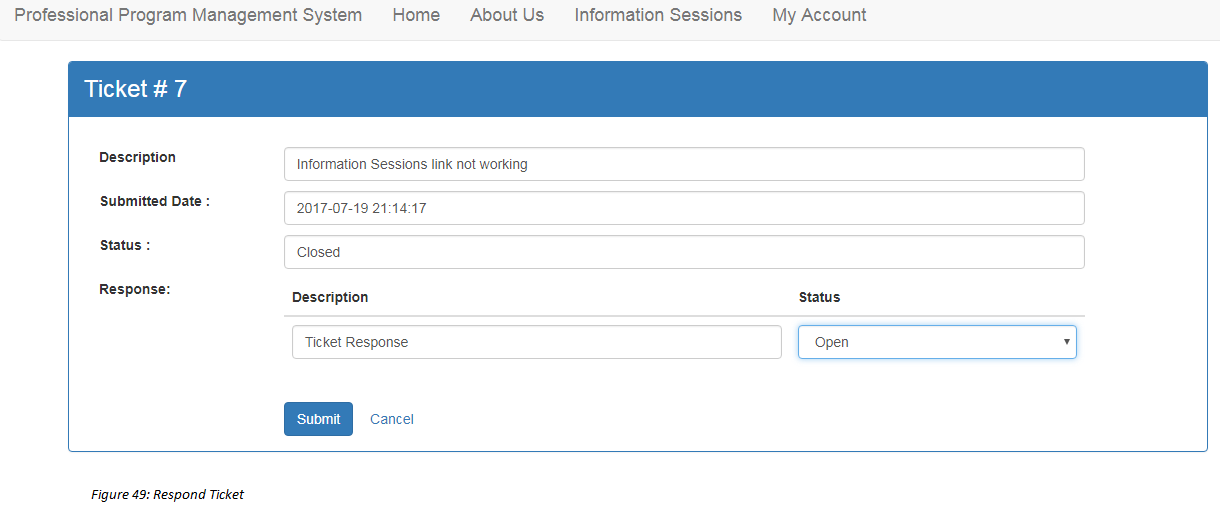












## 

## Appendix C - Sprint Review Reports

* **Sprint 1 Review Meeting**

Attendees: Joana Fraga, Jose Casanova, Nagarajan Prabakar

Start time: 6/3/2017 6:40 PM

End time: 6/3/2017 8:00 PM

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

* User Story 663: Initial backend setup/home page prototype

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

* User Story (Besides the extensive research about the technologies to developed this application, we only focused on 1 user story: 663)
* How this should be reflected on the user story definition in Mingle:
  + User Story 663: Sprint 1
  + No backlog user stories
* **Sprint 2 Review Meeting**

Attendees: Joana Fraga, Jose Casanova, Nagarajan Prabakar

Start time: 6/19/2017 5:50 PM

End time: 6/19/2017 7:00 PM

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

* User Story 669: Create an Account
* User Story 672: Login and Logout

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

* User Story 673: Reset Password
* How this should be reflected on the user story definition in Mingle:
  + User Story 669: Sprint 2
  + User Story 672: Sprint 2
  + User Story 673: Product Backlog
* **Sprint 3 Review Meeting**

Attendees: Joana Fraga, Jose Casanova, Nagarajan Prabakar

Start time: 7/10/2017 4:30 PM

End time: 7/10/2017 6:00 PM

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

* User Story 670: View all available schedules for Q&A sessions
* User Story 673: Reset Password
* User Story 674: Submit a ticket

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

* None
* How this should be reflected on the user story definition in Mingle:
  + N/A
* **Sprint 4 Review Meeting**

Attendees: Joana Fraga, Jose Casanova, Nagarajan Prabakar

Start time: 7/10/2017 4:30 PM

End time: 7/10/2017 6:00 PM

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

* User Story 671: Administrator should be able to schedule a time for Q&A session.
* User Story 675:View my tickets.
* User Story 676: Change ticket status.
* User Story 677:View all tickets.
* User Story 678: Cancel/Close a ticket.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

* None
* How this should be reflected on the user story definition in Mingle:
  + N/A

## 

## 

## 

## 

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

Contained within software package.

# References

Guidance Used

* Stack Overflow: <https://stackoverflow.com/>
* YouTube: <https://www.youtube.com/>
* PHP Manual: <http://php.net/manual/en/>
* MIST Site: <https://msit.fiu.edu/>

Technologies Used

* Digital Ocean: <https://www.digitalocean.com/>
* MySQL/MySQL Workbench: <https://www.mysql.com/>
* XAMPP: <https://www.apachefriends.org/index.html>
* Angular 2: <https://angular.io/>
* PuTTY: <http://www.putty.org/>
* FileZilla: <https://filezilla-project.org/>
* GitHub: <https://github.com/>