**Documentation for Student’s Progress Application**

**Table of Contents:**

1. User Documentation
2. Programmer Documentation

* Introduction
* Programmer Manual
* Test Cases

**User Documentation:**

This document describes the architecture and design for the Capstone Student’s Progress application being developed for the Metropolitan Community College. It is an application that tracks capstone steps submission of students for faculty. It requires zero computer skills.

Application has fields as student ID, class, instruction’s last name, submit object (object what student submitted: reports, diagrams, application and etc.), submit date(date format is 2019-04-22). On the left side of application you will see a panel, where you will get all records about student’s submission.

*To add data to database about student’s submission:*

1. Fill up all fields on the right.
2. Press button “add”.

*To retrieve data about student’s progress:*

1. Fill out first field on the right ‘student’s ID number’
2. Press button “retrieve”.

“Clear” button clears all fields in case if the user made a mistake and want to restart fill up the fields.

In case of not working application, please, exit the app and restart it again.

*We hope you will find this application useful!*

**Programmer Documentation**

1. **Introduction**

This document describes the architecture and design for the Capstone Student’s Progress application being developed for the Metropolitan Community College. It is an application that tracks capstone steps submission of students for faculty. It requires zero computer skills.

The purpose of this document is to describe the architecture and design of the application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

Users and the customer – they want assurances that the architecture will provide for system functionality and exhibit desirable non-functional quality requirements such as usability, reliability, etc.

Developers – they want an architecture that will minimize complexity and development effort.

Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain on into the future.

**Programmer manual :**

The database is written on MySQL and has 3 tables in it: projects, students, instructors. The application is written in Java. The use case view is below.

1. Start an application
2. Start database server.
3. On the left panel automatically must be seen all records from database.
4. Display records for required student by filling up fields on the right , only student ID is required.
5. While (! connection)   
    do window message error;  
    if (connected)  
    do sql() and display;

*Add records to database:*

1. while (! Connected )
2. do window message error;  
    if (connected)  
    do sql() and add to database;

**Test cases:**

Test JButton”Clear”. It works. Test passed.

Test JPanel. Panel shows records from database. Test passed.

Test Scroll Pane. Pane lets scroll from the top to the button. Test passed.

Test button “Add to database”. Filled out the text fields. Pop up Error window works. The records are added to database. Test passed.

Test button “Add to database”. Empty text fields. Pop up Error window works. Passed test.

Test button “Retrieve”. Empty text fields. Pop up Error window works. Passed test.

Test button “Retrieve”. Filled out the studentID field. Data is on Panel. Passed test.