**Software Requirements Specification**

**for**

Learning Outcomes **Scraper**

**Version 1.1 approved**

**Prepared by Victoria Worthington**

**CSCI 36 Fall 2023**

**November 23, 2022**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Same name | 10/28/22 | Ensuring goals are aligned | 1.1 |
| Revision 2 | 11/23/22 | Ensuring goals are aligned | 1.2 |

# **Introduction**

## **Purpose**

We are creating a web application that can be used easily by Butte College faculty to monitor PLOs.

## **Document Conventions**

This document does not have any unusual conventions.

## **Intended Audience and Reading Suggestions**

The product manager, team leads, developers, and testers are expected to read this document.

## **Product Scope**

Our goal is to allow faculty to find when a PLO was updated, needs to be updated, and will be updated. We also want to allow forms to be submitted by faculty to when a PLO was updated, needs to be updated, or will be updated.

## **References**

The overall project document: <https://docs.google.com/document/d/1Hp_K6G0nsl3MKC0-4r8pBC7QjkZ7fIy9uiX1XjWn-m4/edit?usp=sharing>

The discord: <https://discord.gg/8wuSaGPTYg>

## **Product Perspective**

This project was created to add on to the previous PLO scraper project. To create a more useful tool, rather than just a scraper.

## **Product Functions**

* To scrape the Butte College PLO and Program page. This gets:
  + PLOs
  + Department Chair
  + Type of Program
  + Program Code
  + Program Name
* To store said information in a class, json, or on the database
* To take the data and display them on the website
* To store all the data above historically (have data from previous years)
* To find the following:
  + When a SLO needs to be updated
  + When a SLO was updated
  + When a SLO should be updated
* To be able to submit a form about the above information.
* To view previous forms submitted

## **User Classes and Characteristics**

Instructor: This user class will most likely use this software infrequently, 2 to 5 times a year to check PLOs. They may use it to ensure that their classes are following the requirements. They may use it to ensure that their classes are following their program requirements.

Administrator: This user class will use the software more frequently. They will use it when evaluating classes, programs, and institution. They will be able to use the software to ensure that all PLOs have been updated properly. They will be able to use it to see other faculties form submissions regarding updates.

## **Operating Environment**

* AWS Academy
  + Environment
    - t3.small
    - Ubuntu
* A VM / UTM
  + This can also be run off a VM
* Docker
  + Docker can be used to run the program as well.
* Libraries used
  + npm
  + Node.js
  + Cheerio
  + Ejs
  + Express

## **Design and Implementation Constraints**

Our biggest limitation is our lack of experience and funding. If we did want to create a large product/project, we really wouldn’t be able to due to these factors.

Some other limitations have to do with our software. AWS can work great, but also can turn and fail later. It also can be difficult to set up.

## **User Documentation**

Several READMEs and guides were created to help both users and developers of this software.

Some of these include:

1. Guide on how to set up:
   1. AWS
   2. GitHub
2. Guide on how to start software
3. Guide on how to use software
4. Documentation hub

## **Assumptions and Dependencies**

We have to assume that the last class has done everything possible to finish their web scraper correctly. This means that we should assume that it for the most part works. We will assume that AWS is not going to randomly stop functioning in the middle of the semester. We also depend on the individuals working on this project, that they will be able to complete the requirements we have created.

# **External Interface Requirements**

## **User Interfaces**

Most navigation will be done via the navbar. However, the second way of navigating will be done by clicking on links in lists. We will most likely continue this format and add onto these pages.

## **Hardware Interfaces**

We currently only support computer web-based scrolling. We do not have any CSS set up for mobile. We depend on using Node.js and other libraries mentioned to have a working product.

## **Software Interfaces**

We will be using these libraries:

1. Cheerio
   1. To scrape the butte college pages
2. Node.js
   1. To have an actively running server
3. Ejs
   1. To create modular web pages
4. Sequelize
   1. To take our data and put into our database

We will use MySQL as our DBMS. We will use Javascript as our main programming language. We will be using HTML to display data from our database in a way that a user will be able to read and understand.

## **Communications Interfaces**

We rely on Node.js and HTTP for our rest API calls. We currently do not have much security, so our product is very vulnerable.

# **System Features**

## 3.0 Universal Features

Universal wait time for pulling items from database and displaying onto page 5 seconds.

## Assessment Form

1. Description and Priority
   1. This form is copied off of the one Butte uses to evaluate PLOs. The assessment is used to show when a PLO and CLO has been assessed. it also includes notes, resources, and other information that someone might need when looking to assess a PLO/CLO.
   2. **High Priority, High Difficulty**
2. User Sequence
   1. The user will go to the “Discussion Form” page.
   2. They will enter their name, email, the date of the assessment, the contributors.
   3. Then select a program from the drop-down menu.
   4. They then are greeted with three more menus:
      1. Course
         1. if selected, a new menu is created to show the possible course learning outcomes.
      2. Program Learning Outcomes
         1. If the user tries to submit the form while this is empty, they will be unable to.
      3. Textboxes with the following captions:
         1. Strategies
         2. Resources
         3. Notes
   5. Then the user will be able to submit the form.
   6. They are shown something that indicates the form was submitted successfully or unsuccessfully.
3. Functional Requirements
   1. The form will not allow improper entries.
      1. For example, an email must have an @, the date shouldn't be a string.
   2. The following will be required in the form:
      1. Full name
      2. email
      3. The date of the assessment
      4. The contributors.
      5. Programs will be displayed in a drop-down menu. These programs are from the programs table in the database. This drop-down menu contains:
         1. The course drop-down tab.
         2. The program learning outcomes radial menu. These outcomes must come from that course.
         3. Three textboxes with the following captions:
            1. Strategies
            2. Resources
            3. Notes
      6. The course's drop-down menus display all of the existing programs from the database. This down-down menu contains:
         1. The course learning outcomes radial menu. These outcomes must come from that course.
   3. The textboxes can be written into.
   4. The form can be submitted. When it is submitted, the data from the form will be put on a table in the "reports" page.
   5. The following will cause the form to throw an error and inform the user of said error:
      1. The required entries were not filled out.
      2. A learning outcome wasn't selected.
      3. Something invalid was put into the form.
   6. The form will be successful if all the required fields were filled and no invalids were filled. User is informed of success.

## Assessment Report

1. Description and Priority
   1. Assessment report page creates a table of links where a user can easily find a previous assessment of a program.
   2. **High Priority, Medium Difficulty**
2. User Sequence
   1. User goes to the 'reports' page.
   2. User looks for an assessment they would like to investigate. They can find assessments by the:
      1. Program name.
      2. Course name.
      3. Date assessed.
   3. User selects that assessment. They are shown:
      1. The program name.
      2. The course name.
      3. The program learning outcome.
      4. The course learning outcome.
      5. The strategies.
      6. The resources.
      7. The notes.
      8. The assessment date.
      9. Who submitted the form.
      10. Additional contributors.
   4. The user can make a decision from the data and navigate elsewhere.
3. Functional Requirements
   1. The reports page should display a table of the following:
      1. Program name.
      2. Course name.
      3. Date assessed.
   2. This page should be sorted by date.
   3. The program name displayed in the table is a link, and should go to a new page.
   4. That new page displays the following: (The bolded are required)
      1. **The program name.**
      2. **The course name.**
      3. **The program learning outcome.**
      4. **The course learning outcome.**
      5. The strategies.
      6. The resources.
      7. The notes.
      8. **The assessment date.**
      9. **Who submitted the form.**
      10. Additional contributors.

## Learning Outcomes Timeline

1. Description and Priority
   1. This page separates PLOs and CLOs into three categories: high priority, medium priority, and low priority. The category a priority is in is determined by when it was last assessed.
   2. **Medium Priority, High Difficulty**
2. User Sequence
   1. User goes to page.
   2. User can select click on the following drop downs:
      1. High priority PLOs
      2. High priority CLOs
      3. Medium priority PLOs
      4. Medium priority CLOs
      5. Low priority PLOs
      6. Low priority CLOs
   3. Once clicked on, the drop down table will be displayed. It will show:
      1. outcome ID
      2. course/program
      3. last updated
      4. update by (6 years + last updated)
   4. The user can click on the table row again, it will create another drop down row showing the SLO description.
3. Functional Requirements
   1. Display the title.
   2. Display 6 drop down tables, when clicked on they open:
      1. High priority PLOs
      2. High priority CLOs
      3. Medium priority PLOs
      4. Medium priority CLOs
      5. Low priority PLOs
      6. Low priority CLOs
   3. The tables are organized properly and put everything in the correct table. If a program/course was updated 5-6 years ago, it is medium priority. If it was updated longer ago than that then it is high priority. Less, then it is low priority.
   4. Once clicked on, the drop down table will be displayed. It will show:
      1. The outcome ID
      2. The course/program
      3. The last updated date
      4. The update by date (6 years + last updated)
   5. The table row can be clicked on again, it will create another drop down row showing the SLO description.

## Departments Page

1. Description and Priority
   1. This page is used to display all departments that Butte College has (that are associated with a program.) You can also view the department's chair, affiliated programs, and the PLOs for that program.
   2. **Low Priority, Low Difficulty**
2. User Sequence
   1. User goes to departments page.
   2. They view all of the departments.
   3. They can click on a specific department, which shows them:
      1. Department name
      2. Department chair
      3. Associated programs
3. Functional Requirements
   1. Display a table of department names.
   2. Clicking on a department brings up a page that displays the following information:
      1. Department name
      2. Department chair
      3. Associated programs

## Programs Page

1. Description and Priority
   1. This page is used to display all programs that Butte College has. You can also view the program's description, the program code, and the program learning outcomes.
   2. **Low Priority, Low Difficulty**
2. User Sequence
   1. User goes to programs page.
   2. They view all of the programs.
   3. They can click on a specific program, which shows them:
      1. Program name
      2. Program description
      3. Program code
      4. Program learning outcomes
3. Functional Requirements
   1. Display a table of all of the programs. The columns are:
      1. Name
      2. Type (certificate, degree, etc.)
      3. Department
      4. Code
   2. Clicking on a program name brings up a page that displays the following information:
      1. Program name
      2. Program description
      3. Program code
      4. Program learning outcomes

# **Other Nonfunctional Requirements**

## **Performance Requirements**

Our product must be able to fetch the data from the database. It must do it in a timely manner. It should never have an empty database unless it was just created. It should scrape the Butte programs and departments into the database without errors or issues.

We have somewhat managed to solve this issue by saving a previous scrape and working off of that, rather than constantly needing to rescrape. In all honesty, the scrapes should only be happening four times a year, corresponding to the different learning period (Spring, Summer, Fall, Winter).

## **Safety Requirements**

N/A.

## **Security Requirements**

Currently, we do not contain any personal information that would be harmful if leaked. All of our information are things that could be made public. The worst thing is if the discussion posts were public, and the one who submitted the form didn’t want that.

## **Software Quality Attributes**

The main focus our software needs to have in terms of quality is:

* Accurate data
* Easy to read display

## **Business Rules**

* Programs have many courses, and courses can belong to many programs.
* Courses have many course outcomes, and course outcomes can belong to many courses. (in DB represented as 1:M instead of M:N)
* Programs have many program outcomes, and program outcomes can belong to many programs. (in DB represented as 1:M instead of M:N)
* Assessments can be about many program outcomes; program outcomes can have many assessments made about them.

# **Other Requirements**

Our website design should be reworked and approved by the team.

We must complete this project by December 9, 2022.

**Appendix A: Glossary**

SLO: Student Learning Outcomes

CLO: Course Learning Outcomes

PLO: Program Learning Outcomes

**Appendix B: Analysis Models**

The discord includes all of our analysis models: <https://discord.gg/2hMQsFR6Ak> (This discord will be archived once the class is completed)

**Appendix C: To Be Determined List**

1. Requirements
2. Timeline