Software Requirements Specification

for

Jiwa

Version 1.0 approved

In partial fulfillment of the requirements of CMSC 128

CMSC 128 D-2L

March 21 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Jiwa	07/06/22	Finalized all functions	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to discuss the features and specifications of the online GWA verifier system, Jiwa Version 1.0, developed for the use of the Scholarships, Honors, and Awards Committee (SHAC). The system is designed to verify student records automatically to speed up the process of checking the completeness of the record. Provided in this document is a detailed overview of our software product, its parameters, and goals. This document describes the project's target audience and its user interface, hardware, and software requirements.

1.2 Document Conventions

The conventions used in writing this SRS document are simple. Arial bold-faced text for the headings and indentation denoting the body for each section are used. Bullets are used to denote specific related things. The rest of the document will be written using the conventional font, Arial.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, testers, and users or system customers. Developers read it to understand what system and its features need to be implemented. Users specify requirements and may read them to check if their needs are satisfied. The manager uses this to plan a software development process intended to develop a software as prescribed by the users, and testers use this document to create different test cases in validating the system.

The remainder of this document contains four chapters. The second chapter shows an overview of the system's functionality. It generally describes what the system is. It also introduces different types of stakeholders and their interaction with the system–briefly describing the features and interactions that can be done with the system according to user class. It also describes what platform the system will operate including the hardware platform, operating system, and other applications and software components needed to help ensure a smooth-running system. In addition, the chapter also mentions the system constraints and assumptions regarding the product. This chapter is useful for the users or system consumers.

The third chapter provides the requirements specified by the users and a description of the system interfaces in a detailed and more technical manner. This chapter is primarily useful for developers and project managers as it describes the functionalities of the software in technical terms and also for users as it also features different use cases.

The fourth chapter describes the external interface requirements. It provides a visual description of the user interface and lists down the framework needed to develop it. This chapter also shows the interface requirements of the software and the hardware–the main hardware component that regulates the system, and software components that provide means to interact with hardware. This chapter is useful for the developers as it describes the user interface and the interaction between *the hardware and the software. It is also for the users as it details the hardware and software needed to use the system.*

Lastly, the fifth chapter specifies the non-functional requirements or the constraints on services and functions offered by the system. This chapter is useful for developers and project managers as it details the standard of the system, and also for users as it sets the expectations for the system.

1.4 Project Scope

This product is a new general weighted average (GWA) verifier website application developed by the students of CMSC 128 D-2L to meet the needs of the stakeholders - the CAS Scholarships, Honors, and Awards Committee. The application allows SHAC members to add student records, verify them, and edit them in the case of inconsistencies found in the student file. A summary of student records is stored and shown, and this allows SHAC members to deliberate students without much technicality or difficulty.

1.5 References

Bandakkanavar, R. (2019, October 17). Software requirements specification document with example. Krazytech. Retrieved June 7, 2022, from https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database

2. Overall Description

2.1 Product Perspective

The project aims to provide CAS SHAC members with a better overview of graduating students' records. To be precise, the project accepts student records provided by a SHAC member and the project allows the member to view and edit the records when necessary. Above all, we hope to prioritize a comfortable user experience alongside their freedom to input the data into the system. Additional features may be implemented in the future, in order to improve its convenience

2.2 Product Features

The system will:

- *Allow user authentication*
- *Allow the administrator to authorize an authenticated user*
- *Allow an authorized user to upload documents*
- *Allow an authorized user to view the summary of records of all the students in the system*
- *Allow an authorized user to sort the summary of records of all the students in the system*
- *Allow an authorized user to view the history of all the changes made in the record*
- *Allow an authorized user to edit the student record*
- *Allow an authorized user to download and print the student record*
- *Allow an authorized user to see the inconsistencies in the formatting of student record*
- *Allow an authorized user to edit account information*

2.3 User Classes and Characteristics

Users of the application should be able to add student information such as their full name, their course and major. They should be able to add information about the subjects they have taken, the semester and school year it was taken, and their grade on the subject. Their running GWA will also be included. They will also be able to upload files containing the said information. The application will have two types of users, the administrators, and the general users. The administrators should be able to do what the general users can do and have additional features.

The general users should be able to do the following:

- *Add a student entry*
  - *Manual input*
  - *Upload file(s)*
- *Verify the grade and standing of the students they added*
  - *Running GWA*
  - *Gap year/semester*
- *Edit the students they added*
  - *Correct final GWA*
  - *Correct running GWA*
  - *Correct semesters*
- *Delete the students they added*
- *Change their name, password, and recovery questions*

The administrators should be able to do the following:

- *Add a student entry*
  - *Manual input*
  - *Upload file(s)*
- *Verify any student's grade and standing*
  - *Running GWA*
  - *Gap year/semester*
- *Edit any student's information*
  - *Correct final GWA*
  - *Correct running GWA*
  - *Correct semesters*
- *Delete any student's record*
- *Delete any user*
- *Change any user's name*
- *Approve password and recovery questions change*
- *Approve new user*

2.4 Operating Environment

The Operating environment for JIWA (Just Input, Wait and Authenticate) is as listed below.

- *Client/server system*
- *Server: Linux OS*
- *Client OS: Windows*
- *Database: MongoDB*
- *Platform: Node.js*
- *Browser: Google Chrome*
- *Local Area Network connection (LAN)*

2.5 Design and Implementation Constraints

Design and implementation of the system is limited by the minimal time of the developers in the implementation of security and session. Since this is an academic requirement for CMSC 128 students, the time and availability of each member of the group may influence the productivity of the entire team. The team will not be responsible for system deployment and maintenance.

- *The system shall be designed using Models, API routes, and CRUD functionality*
- *The system interface shall be developed using React, UI design with CSS, connecting frontend to backend.*
- *The system shall be developed using MongoDB with express API and mongoose to store data.*
- *The system shall be developed with JSON Web Token, bcryptJS, and form validations for a secure user authentication.*
- *Portability: The software shall be able to run on multiple platforms and devices as long as there is a web browser installed on the device.*
- *Language requirements: The software shall be in English.*

2.6 User Documentation

The user manual can be found within the software itself. It can be accessed by clicking on the manual section of either the navigation bar or the dashboard. It contains instructions and images on how to use this application.

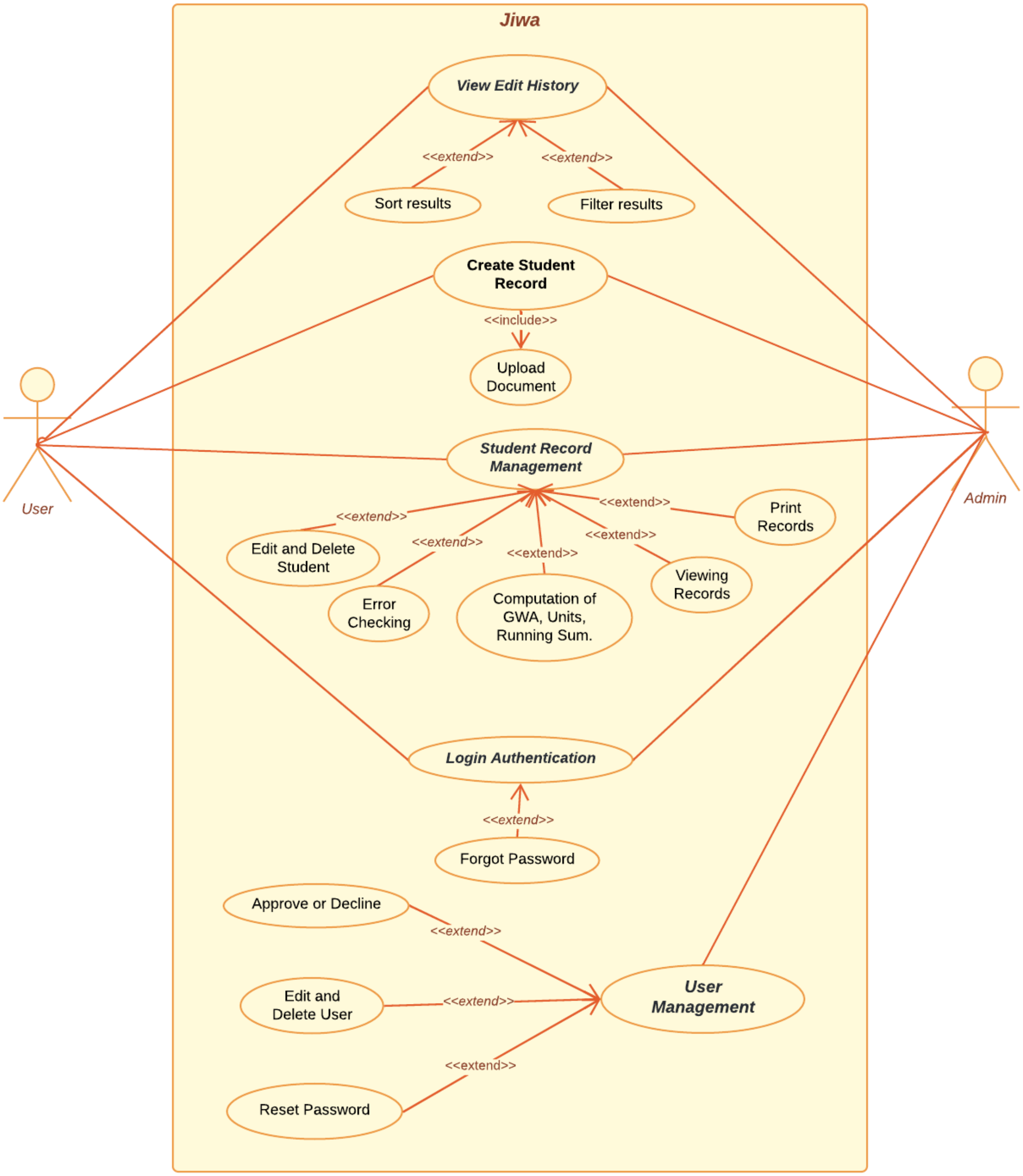
2.7 Assumptions and Dependencies

The following are the main assumptions:

- *The College of Arts and Sciences can provide the required hardware to support user load.*
- *Computer systems available to the user have a chromium-based browser and NodeJs installed.*
- *The user is familiar with the basics of using a keyboard and mouse.*
- *Student data to be uploaded or inputted will be made available prior to the usage of the application.*
- *The users will be working in the same area and at the same time while using this app*

3. System Features

3.1 Use-Case Diagram



3.2 Create Account

3.1.1 Description and Priority

High priority is given to this feature. It includes creating a new account for users which will be authorized by the administrator.

3.1.2 Stimulus/Response Sequences

- *The user will click the Create Account button on the Home Page.*
- *The user will enter his or her information on the Sign up page.*
- *The user will click on the Register button and the new user is saved.*

3.1.3 Functional Requirements

REQ-1: The Password requirement should be met for the Account creation to push through.
REQ-2: There must be input field validations.
REQ-3: Register button should be disabled if there is an input error.

3.3 User Authentication

3.2.1 Description and Priority

High priority is given to this feature. It includes basic user login and logout functionalities, as well as the Forgot Password feature.

3.2.2 Stimulus/Response Sequences

- *The user will enter his or her credentials.*
- *The user will click the login button.*
- *The user will be redirected to the Dashboard if login is successful.*
- *The user will see his or her name on the top right of the screen, which is clicked to see the profile options.*
- *The user clicks logout.*

3.2.3 Functional Requirements

REQ-1: There must be input field validations.
REQ-2: Passwords must meet the requirements for the login to push through.

3.4 User Management

3.3.1 Description and Priority

High priority is given to this feature. It includes viewing, searching, editing, deletion, accepting, and declining of users. It also includes resetting of user passwords and viewing individual user uploads. Only the administrator can access this feature

3.3.2 Stimulus/Response Sequences

- *The user clicks user management in the dashboard or manage in the navigation bar.*
- *The user will be redirected to the SHAC Users tab, and all users will be displayed.*
- *The user can click a particular user to view.*
- *The user can click delete user to delete a non-admin user.*
- *The user can click view uploads to see student records created/uploaded by the selected user.*
- *The user can click the save changes button to save non-admin user edit changes.*

- *The user can click the Registrations tab to view registration requests.*
- *The user can confirm or decline a registration request.*
- *The user types “confirm registration” then clicks the confirm action button to confirm a registration request.*
- *The user types “decline registration” then clicks the confirm action button to decline a registration request.*
- *The user can click the Reset Requests tab to see reset password requests.*
- *The user can click the Reset button to reset non-admin user passwords.*

3.3.3 Functional Requirements

REQ-1: The user must be an administrator.
REQ-2: The user must type confirmation texts to perform saving and deletion of non-admin users.
REQ-3: The user must not be able to edit the administrator.
REQ-3: The user must type confirmation texts to perform confirming and declining of registration requests.
REQ-4: The user, after creating or generating a password, must click the confirm action.

3.5 Edit Profile

3.4.1 Description and Priority

High priority is given to this feature. It includes editing of personal information, changing of password, and updating of security questions. All users can perform this feature. An administrator, however, does not have updating of security questions functionality.

3.4.2 Stimulus/Response Sequences

- *The user clicks the dropdown in the navigation bar.*
- *The user clicks the Edit Profile option in the dropdown.*
- *The user can click the save changes button to change his/her personal information.*
- *If there are inputs in the Change Password card, the user can click change password*
- *The user can click the change password button to change his/her password.*
- *If there are inputs in the Security Questions card, the user can click the update security questions button to change security questions and answers.*
- *The user inputs his/her current password to be able to save changes.*

3.4.3 Functional Requirements

REQ-1: There must be a confirmation modal that asks for the user's password before saving the changes.
REQ-2: There must be non-empty fields aside from the middle name to be able to edit the user profile.
REQ-3: Password requirements must be met to be able to change password.

3.6 Upload Documents

3.5.1 Description and Priority

High priority is given to this feature. It includes the uploading of necessary documents to a specific student record by the administrator.

3.5.2 Stimulus/Response Sequences

- *The user clicks the Browse files button or drag files to the upload box to stage files for upload.*
- *The user clicks the Upload button.*
- *The user may be prompted to fix inconsistencies if there are.*
- *The user clicks Finish after all the inconsistencies have been fixed.*
- *The user is shown a failure or success screen with buttons to view uploaded records, view all records, or upload again.*

3.5.3 Functional Requirements

REQ-1: The upload box must be able to upload csv files.
REQ-2: Inconsistencies must be shown to the user using popups or text.

3.7 Create Student Record

3.6.1 Description and Priority

High priority is given to this feature. It includes creation, editing, and saving of a student record. Any type of user can do this functionality

3.6.2 Stimulus/Response Sequences

- *The user clicks upload documents in the dashboard or upload in the navigation bar.*
- *The user clicks the create file button.*
- *The user fills in every input field.*
- *The user adds courses and terms to the table.*
- *The user saves the student record.*
- *The student record undergoes error checking.*

3.6.3 Functional Requirements

REQ-1: There must be input fields validation.
REQ-2: Add and save button must be disabled if an error exists.
REQ-3: GWA, total values per student data, and total values per term must be computed automatically.
REQ-4: A confirmation modal appears when canceling or saving created student data.

3.8 View All Student Records

3.7.1 Description and Priority

High priority is given to this feature. It includes viewing, searching, filtering, sorting, deletion, and printing of student records. This is also where the user can navigate to a particular student record.

3.7.2 Stimulus/Response Sequences

- *The user clicks view all records in the dashboard or records in the navigation bar.*
- *The user can search for student records based on the chosen filter.*
- *The user can sort the student records based on nothing, first name, last name, degree, email, and GWA.*
- *The user can print the student records displayed on the current page.*
- *The user can delete student records he/she created/uploaded. If the user is the administrator, he/she can delete all student records.*
- *The user can view a particular student record.*

3.7.3 Functional Requirements

REQ-1: There must be input fields for range when filtering by GWA.
REQ-2: Print window must appear after clicking the print button.
REQ-3: A notification dialog must appear if the delete button was clicked but no student is selected.
REQ-4: A confirmation modal must appear when deleting student record/s.

3.9 Student Record

3.8.1 Description and Priority

High priority is given to this feature. It includes viewing, editing, error checking, saving, and printing of a student record. It also includes automatic computations of GWA, computed/weight, and total number of units and running sum per term and per student record. Only the user who created/uploaded the student record or the administrator can edit the selected student record.

3.8.2 Stimulus/Response Sequences

- *The user goes to all student records.*
- *The user clicks the view button of the selected student record.*
- *The detailed data of the selected student record will be displayed.*
- *The user can click the edit button to edit the student record if he/she is the uploader or the administrator.*
- *The user can save or cancel the changes he/she made.*
- *After clicking the save button, the user adds a brief description about the changes made.*
- *The user can print the student record.*

3.8.3 Functional Requirements

REQ-1: There must be no errors in the display of the student data.
REQ-2: Edit button must not appear if the user is not the uploader or the administrator.
REQ-3: Print window must appear after clicking the print button.
REQ-4: Errors in grade and number of units must not be allowed in editing.
REQ-5: Save button must be disabled if there are errors.
REQ-6: Confirmation modal must appear if the cancel button was clicked.
REQ-7: Documentation modal with text field and another save button must appear after clicking the save button in the editing page.

3.10 View Edit History

3.9.1 Description and Priority

High priority is given to this feature. It includes viewing, sorting, searching, and printing of the authorized user's edit history. This is also where an authorized user can scan and trace back the changes made on a specific student record.

3.9.2 Stimulus/Response Sequences

- *The user clicks the view button to Edit History.*
- *The details of every edit and changes made to a student record will be displayed on the screen.*
- *The user clicks on the "Sort By" drop button which shows options on how to sort the edit history such as student last name, student first name, and degree.*
- *The user clicks on the "Filter By" drop button which shows different parameters that can be searched in the edit history such as student last name, student first name, user first name, and user last name.*
- *The user clicks on the "Print" button on the upper right side of the screen and a confirmation will show so the user can print the edit history.*

3.9.3 Functional Requirements

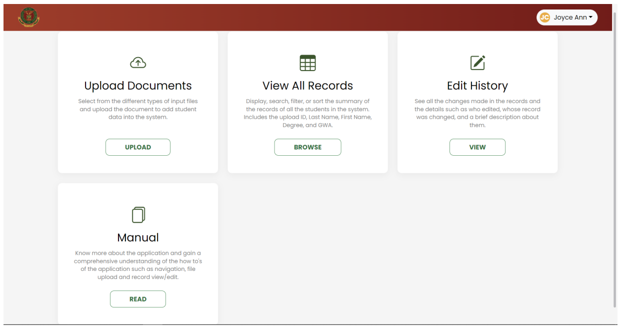
REQ-1: Print window must appear after clicking the print button

4. External Interface Requirements

4.1 User Interfaces

The application GUI provides a navigation bar, buttons, tables, and modals, as well as a responsive design for easier navigation and control. A minimalistic style was implemented in order to attune to the user's ease of access and readability. The color theme was drawn from the colors of the university seal: maroon, forest green, and yellow. The frameworks used for the user interface are React, a JavaScript library, and Bootstrap, a CSS framework.

4.1.1. Dashboard



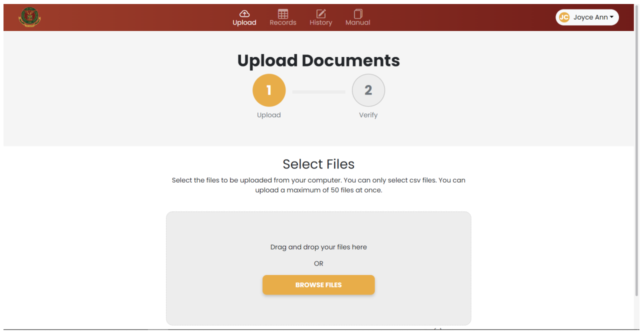
Select from the different types of input files and upload the document to add student data into the system.

Display, search, filter, or sort the summary of the records of all the students in the system. Includes the upload ID, Last Name, First Name, Degree, and GWA.

See all the changes made in the records and the details such as who edited, whose record was changed, and a brief description about them.

Know more about the application and gain a comprehensive understanding of the how to's of the application such as navigation, file upload and record view/edit.

4.1.2 Upload documents



Upload Documents

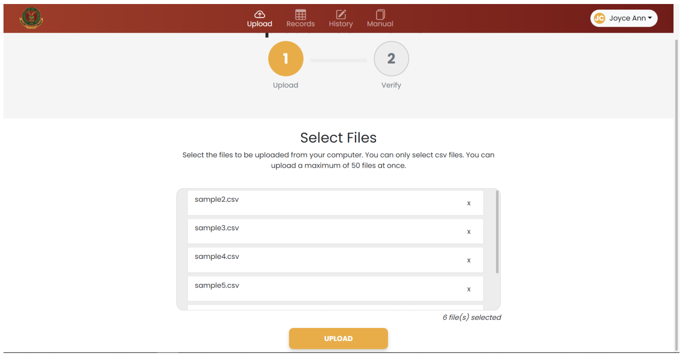
Select Files

Select the files to be uploaded from your computer. You can only select csv files. You can upload a maximum of 50 files at once.

Drag and drop your files here

OR

Upload Documents page.



Select Files

Select the files to be uploaded from your computer. You can only select csv files. You can upload a maximum of 50 files at once.

sample2.csv

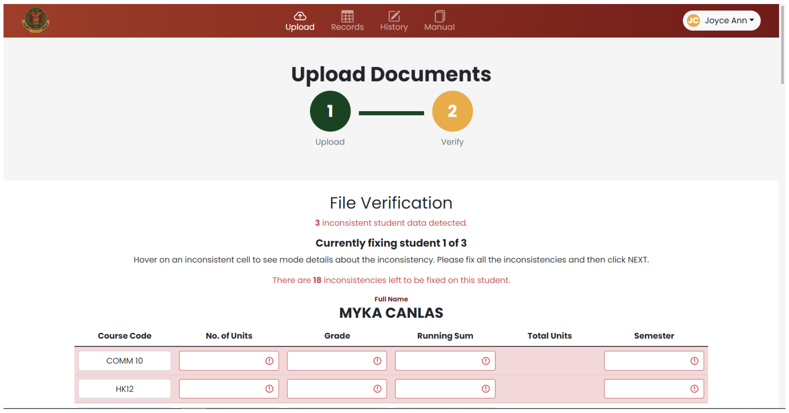
sample3.csv

sample4.csv

sample5.csv

6 file(s) selected

Selected files



Upload Documents

File Verification

3 inconsistent student data detected.

Currently fixing student 1 of 3

Hover on an inconsistent cell to see mode details about the inconsistency. Please fix all the inconsistencies and then click NEXT.

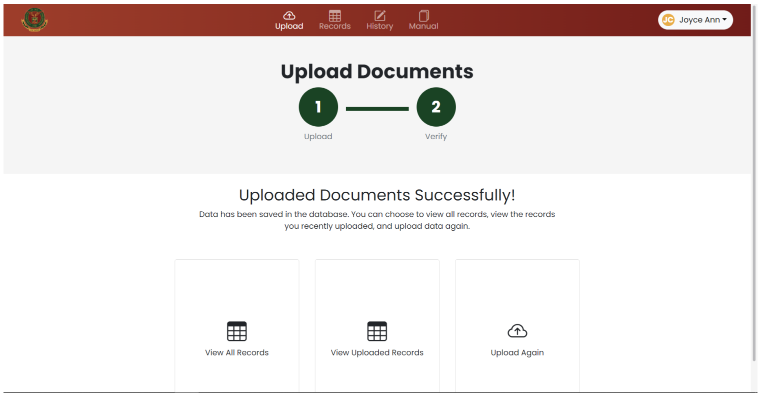
There are **18** inconsistencies left to be fixed on this student.

Full Name

MYKA CANLAS

Course Code	No. of Units	Grade	Running Sum	Total Units	Semester
COMM 10	ⓘ	ⓘ	ⓘ		ⓘ
HK12	ⓘ	ⓘ	ⓘ		ⓘ

File with inconsistencies detected.



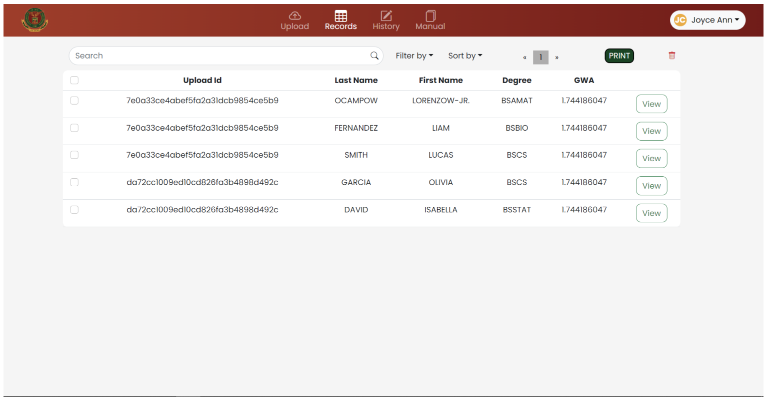
Upload Documents

Uploaded Documents Successfully!

Data has been saved in the database. You can choose to view all records, view the records you recently uploaded, and upload data again.

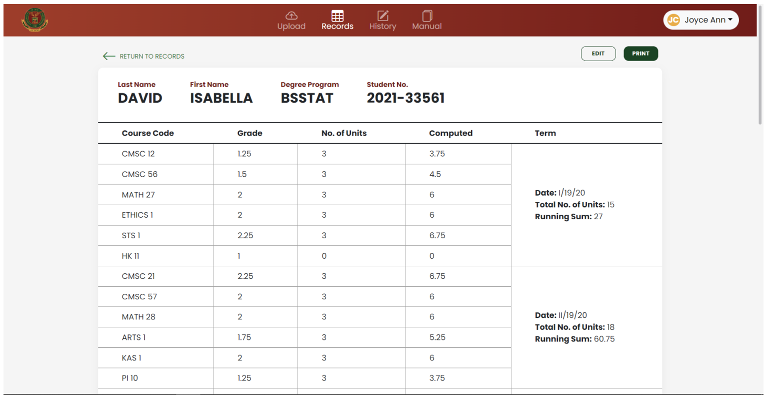
Display after files have been uploaded successfully.

4.1.3 Viewing records



□	Upload Id	Last Name	First Name	Degree	GWA	
□	7e0a33ce4abef5fa2a31dcb9854ce5b9	OCAMPOW	LORENZOW-JR.	BSAMAT	1.744186047	View
□	7e0a33ce4abef5fa2a31dcb9854ce5b9	FERNANDEZ	LIAM	BSBIO	1.744186047	View
□	7e0a33ce4abef5fa2a31dcb9854ce5b9	SMITH	LUCAS	BSCS	1.744186047	View
□	da72cc1009ed10cd826fa3b4898d492c	GARCIA	OLIVIA	BSCS	1.744186047	View
□	da72cc1009ed10cd826fa3b4898d492c	DAVID	ISABELLA	BSSTAT	1.744186047	View

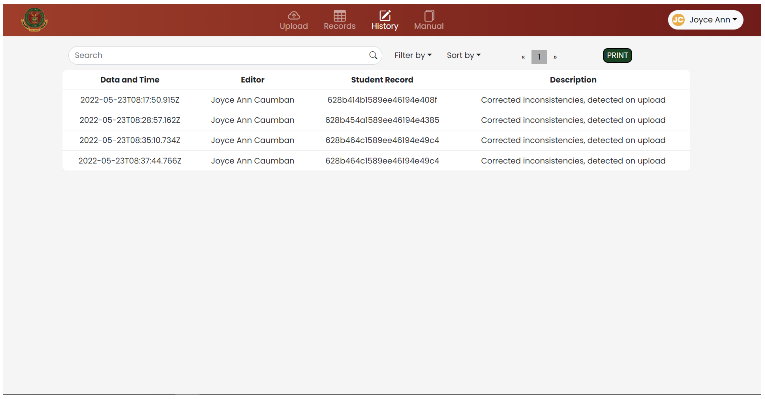
View All Records page.



Last Name	First Name	Degree Program	Student No.
DAVID	ISABELLA	BSSTAT	2021-33561

Course Code	Grade	No. of Units	Computed	Term
CMSC 12	1.25	3	3.75	**Date:** 1/19/20 **Total No. of Units:** 15 **Running Sum:** 27
CMSC 56	1.5	3	4.5	
MATH 27	2	3	6	
ETHICS 1	2	3	6	
STS 1	2.25	3	6.75	
HK 11	1	0	0	
CMSC 21	2.25	3	6.75	**Date:** 11/19/20 **Total No. of Units:** 18 **Running Sum:** 60.75
CMSC 57	2	3	6	
MATH 28	2	3	6	
ARTS 1	1.75	3	5.25	
KAS 1	2	3	6	
PI 10	1.25	3	3.75	

View student record page



Data and Time	Editor	Student Record	Description
2022-05-23T08:17:50.915Z	Joyce Ann Caumban	628b414b1589ee46194e408f	Corrected inconsistencies, detected on upload
2022-05-23T08:28:57.162Z	Joyce Ann Caumban	628b454a1589ee46194e4385	Corrected inconsistencies, detected on upload
2022-05-23T08:35:10.734Z	Joyce Ann Caumban	628b464c1589ee46194e49c4	Corrected inconsistencies, detected on upload
2022-05-23T08:37:44.766Z	Joyce Ann Caumban	628b464c1589ee46194e49c4	Corrected inconsistencies, detected on upload

Edit history page

4.2 Hardware Interfaces

The back-end and database components will be deployed in a local distribution. Client computers will be able to access the system by using a standard browser (preferably a Chromium-based browser) running on a Windows operating system via the given URL of the said application.

4.3 Software Interfaces

Software Used	*Description*
Operating System	*We have chosen to use the Windows operating system for it's the most common and most user-friendly*
Database	*To save the student and user data, we have chosen to use the MongoDB database*
Node	*To implement the project we have chosen to use Node for its flexibility*

4.4 Communications Interfaces

Standard HTTP protocol would be used as the protocol for communication. This will be done through the use of a standard web browser.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

At a minimum, the system must be available despite the periodical blackouts that occur all throughout the UPLB Campus. The target users of this system intend to use this application within University premises only, and as such are subject to the campus' intermittent connection. In spite of this, the system must still be useable to some degree.

In addition, in order to prevent repetition of operations within a single handler, certain database properties must be joined. Due to the constraints of Mongoose, a traditional join is not possible—instead, the system will use Mongoose's populate function, which is similar to a join.

5.2 Safety Requirements

There are currently no safety requirements.

5.3 Security Requirements

The server on which the application resides will have its own security to prevent unauthorized write, read, and delete access.

5.4 Software Quality Attributes

AVAILABILITY: Students will be available to all users in the system connected to the server
RELIABILITY: The student data and GWA will have correct values at all times

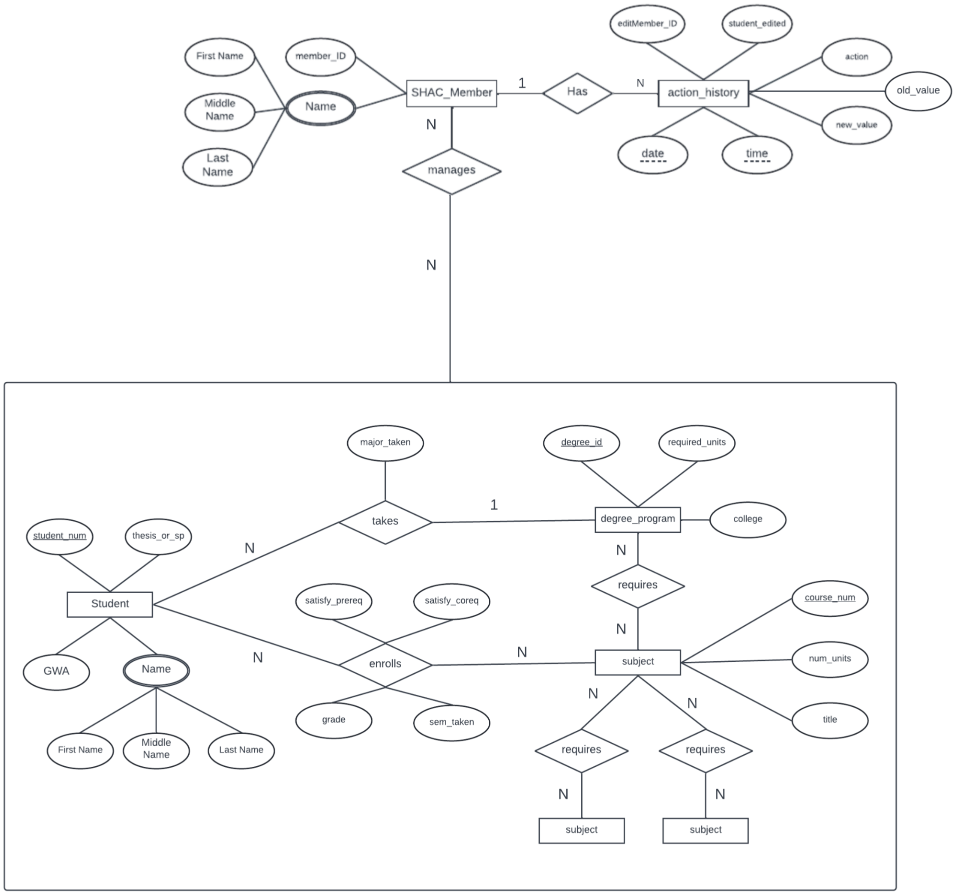
6. Other Requirements

Appendix A: Glossary

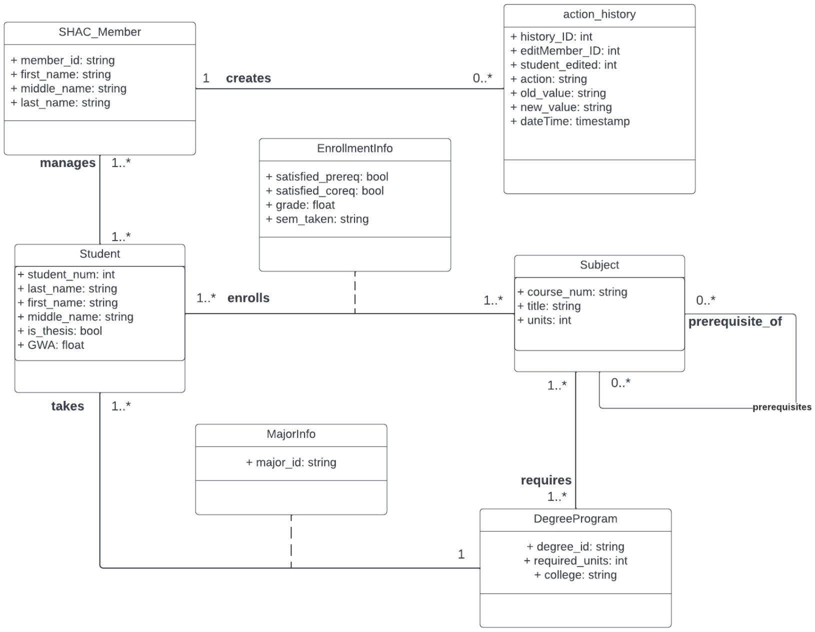
Term	***Definition***
SHAC	*Scholarships, Honors, and Awards Committee*
CAS	*College of Arts and Sciences*
GWA	*Graded Weighted Average*

Appendix B: Analysis Models

1. *Entity Relationship Diagram*



2. *Unified Modeling Language*


SHAC_Member
+ member_id: string
+ first_name: string
+ middle_name: string
+ last_name: string
1
creates
0..*
action_history
+ history_ID: int
+ editMember_ID: int
+ student_edited: int
+ action: string
+ old_value: string
+ new_value: string
+ dateTime: timestamp
manages
1..*
1..*
EnrollmentInfo
+ satisfied_prereq: bool
+ satisfied_coreq: bool
+ grade: float
+ sem_taken: string
Student
+ student_num: int
+ last_name: string
+ first_name: string
+ middle_name: string
+ is_thesis: bool
+ GWA: float
1..*
enrolls
1..*
Subject
+ course_num: string
+ title: string
+ units: int
0..*
prerequisite_of
1..*
0..*
prerequisites
takes
1..*
MajorInfo
+ major_id: string
requires
1..*
1
DegreeProgram
+ degree_id: string
+ required_units: int
+ college: string