CrAISIS Averted

Software Design Document

Date: 04/11/2023

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This software design document holds all the necessary details of CrAISIS Averted, a proposed system to optimize the enlistment process for Ateneo De Manila University (ADMU) Students. These details include all relevant information regarding the system's scope, architecture, and data as well as the design of each component and the system's user interface (UI) and user experience (UX) as a whole.

1.2 Scope

The main goal of the CrAISIS Averted is to subvert the inconveniences and major issues caused by the usual problems encountered during the enlistment process; this includes problems such as outdatedness and the lack of easy accessibility. We want to ensure that every student can secure slots for all of their required classes each and every term without being hindered by the site breaking down or restrictions for classes staying in place when they should not.

The system intends to focus on the student side of the enlistment process and will only make use of a prepared simulated database for details of slots, number of units, etc. Furthermore, the system will only deal with the improvement of the enlistment function of AISIS and will not include other parts of the enlistment process such as the load revision form, update of information, etc. In addition to this, while the processing of payment is included in the enlistment process it is during the latter half, at which point the user would have completed their enlistment. Because the main goal of CrAISIS Averted is the optimization of the actual enlistment function, the payment option, and the technical systems behind it, will not be within the scope and the system will not implement any features in regard to the teachers, registrars, or admin's side of the system.

1.3 Overview

This document examines the group's proposed software - it runs through an overview of the system as well as an in-depth analysis of the system's architecture including its design, decomposition, and the rationale behind it. It then covers the system's data as well as a comprehensive look into the design of each of the system's components and its UI and UX. The document also covers all related definitions and acronyms for both the system and its data as well as all sources referenced throughout.

1.4 Reference Material

- "Software Requirement Specification for AISIStant"
- Django documentation v4.1

1.5 Definitions and Acronyms

Below we can find the various definitions and acronyms which will be used throughout both the document and the implementation of the software:

- **AISIS**
    - Ateneo Integrated Student Information System
    - This is the official student portal of Ateneans wherein they can do various tasks such as viewing their IPS, checking other curriculums, checking their grades, and faculty evaluations are a few examples. Most importantly for the purpose of this document, the enlistment, and payment of tuition assessment fees every semester.
- **ADMU**
    - Ateneo De Manila University
- **Enlistment**
    - The act of enlisting or enrolling into a subject based on the student’s IPS which dictates which subjects they need to enroll in.
- **IPS**
    - Individual Program of Study

- This is the curriculum or plans that the student follows based on what their course is. This is also the basis for which classes the students need to enlist in during the enlistment period.

2. SYSTEM OVERVIEW

2.1 Product Perspective

The project will consist of a brand new user-friendly UI and UX which will allow the users to enlist with ease by displaying the subjects they will be enlisting in, the pre-enlisted classes (if applicable), and the subjects they have already enlisted all in one view. This will ensure that they could easily keep track of their schedule without leaving the page as the current AISIS system does not incorporate this feature.

As mentioned, the revamped system will include the schedule that the current user has up for display in a weekly schedule template while enlisting. Together in this display would be a list of the subjects the user would need to enlist in and thus when they pick a class to enlist in, it would show up in the weekly schedule view. The system will also mark all of the conflicting schedules of the other classes while within the enlistment page in order to aid the user when choosing their classes, allowing them to avoid overlapping classes. The current system does not indicate if the classes are conflicting until the user attempts to submit their enlistment, this is what we want to avoid for a more efficient enlistment system.

The project name CrAISIS Averted originates from the ideation that whenever students see the AISIS student portal, it is usually joked about because of how the use of the portal is quite stressful and often disastrous, especially during enlistment season. With this, the name would of course want to avert and stray away from this crisis.

2.2 Product Functions

- User's Week Calendar Interface
- Week Calendar of Available Scheds Interface

- Class Schedule Planner
- Automatic Enlistment
- Prevention for conflicting classes
- Login & Authentication

2.3 User Classes and Characteristics

As mentioned in the scope, we will mostly be focusing on the student side view of AISIS' enlistment process. Thus, the targeted main users for this CrAISIS Averted would be the **undergraduate students of the ADMU community**. Even for students who have done enlistment multiple times, the struggle is still imminent because of how the system works, and that is what we want to prevent. We may consider looking into admin matters in the future.

2.4 Operating Environment

The environment in which the software would operate should have a minimum requirement of Windows 8 and above or macOS Yosemite and above. However, generally, what matters more is access to a web browser. To be specific, we are developing this for most conventional and widely-used web browsers such as Chrome, Safari, and Firefox.

2.5 Design and Implementation Constraints

For the design and implementation of our product, the limitations would include software and hardware constraints wherein we will be using what is already available to the developers. This includes free software from which we have learned and used in our previous classes. Besides this, we will not be able to get the actual data and database format from the current AISIS, instead, we will be creating dummy data to help in the demonstration based on the available resources in AISIS.

2.6 Assumptions and Dependencies

For the majority of the development of the product, the group will be assuming that the enlistment process of AISIS will remain the same; thus, the process that will be done in CrAISIS Averted will be the same as the existing one. For the dependencies, as mentioned the dummy data will be based on the data found in AISIS.

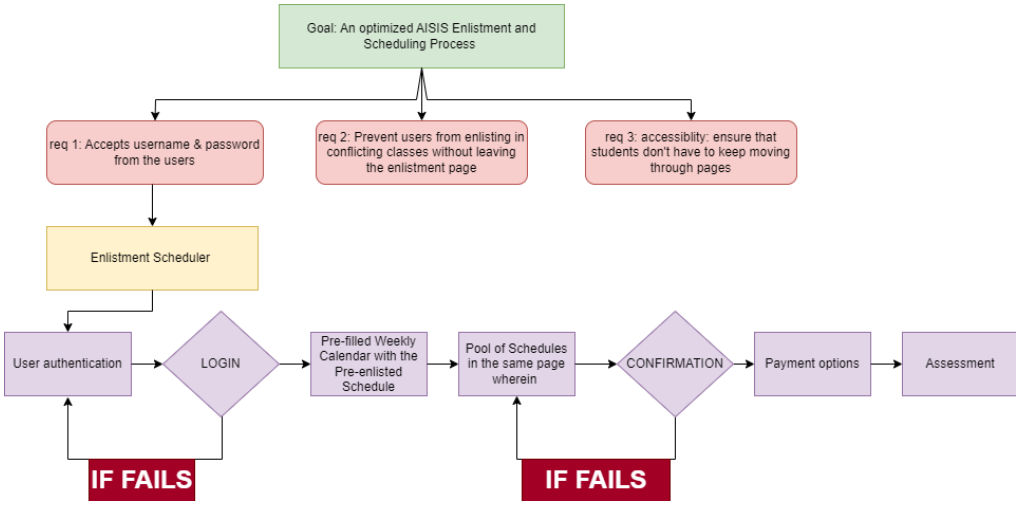
3. SYSTEM ARCHITECTURE

3.1 Architectural Design

The structure of this system would be primarily divided into 2 distinct categories namely the front end(browser based) and the back end(server). The backend, which will make use of the Django framework as its backbone, will be the one responsible for processing the main functionalities of CrAISIS Averted such as authentication, processing of user's input, and connecting the frontend to the database. The front end, on the other hand, is the one that would be responsible for the UI and UX, their interactions with the web pages, what they see, click, etc. Specifically, for this software, we will be using SQLite to create and maintain the database server.

The front end and the back end will be communicating using javascript object notation through the use of the REST Framework. The use of Django, JSON, and the Rest Framework was decided based on previous experience that the group has on them. Moreover, since the project is simple enough, there was no need to consider using more advanced applications.

3.2 Decomposition Description



3.3 Design Rationale

The decisions for coming up with the design are based on the developer's familiarity and experience with it. Given the time constraints in developing this application, the developers found it more suitable to focus on enhancing what they already know rather than having to learn a completely new process.

4. DATA DESIGN

4.1 Data Description

As mentioned previously, we will not be able to get actual data from the supposed offices so we will be creating dummy data based on our own IPSs. We can then store this dummy data we have through Django's Panel, or more specifically, SQLite3.

4.2 Data Dictionary

Entity Name	student
Entity Description	The student refers to an individual enrolled in ADMU. They will be the main user for this project as the view is being created for them.

Attribute Name	Description	Primary Key?	Default Value	Type of Values	Can be null?
student_id	The unique and assigned student number given to the student once they enroll.	Y	0	int	N
first_name	The legal first name of the student.	N	" "	string	N
last_name	The legal family name of the student.	N	" "	string	N
course_code	The course code of the student which is an acronym of their	N	" "	string	N

	full course.				
age	The age of the students is based on their year of birth.	N	0	int	N
regular?	This dictates whether or not a student has a regular or irregular IPS.	N	Y	boolean	N

Entity Name	Class
Entity Description	This refers to the class in which the student enlists in. A student may enlist in more than 1 class to fill out their load for the semester.

Attribute Name	Description	Primary Key?	Default Value	Type of Values	Can be null?
class_id	This is the unique identifier for a particular class.	Y	0	int	N
class_code	This is the code which is an acronym for the full class name.	N	“ ”	string	N
class_section	This is the section code of the class.	N	“ ”	string	N
schedule	This refers to each individual schedule	N	CURRENT_TIMESTAMP	timestamp	N

	for each student.				
max_slots	This refers to the max number of slots available for a particular class.	N	0	int	N
available_slots	This refers to the number of slots that are still available for enlistment within a particular class.	N	0	int	N
instructor_id	This refers to the instructor assigned to a particular class.	N	0	int	N
department_id	This refers to the department that a particular class belongs to.	N	0	int	N

Entity Name	Professor
Entity Description	This refers to the Instructor that is handling the class which the students enlist for.

Attribute Name	Description	Primary Key?	Default Value	Type of Values	Can be null?

company_id	This is the unique identifier for each Instructor.	Y	0	int	N
first_name	This is the first name of the Instructor.	N	“ ”	string	N
last_name	This is the last name of the Instructor.	N	“ ”	string	N
department_id	This is the department which the Instructor falls under.	N	0	int	N

Entity Name	Department
Entity Description	This refers to the group in which classes relevant to each other and fall under are classified in and as well as the Instructors associated with them.

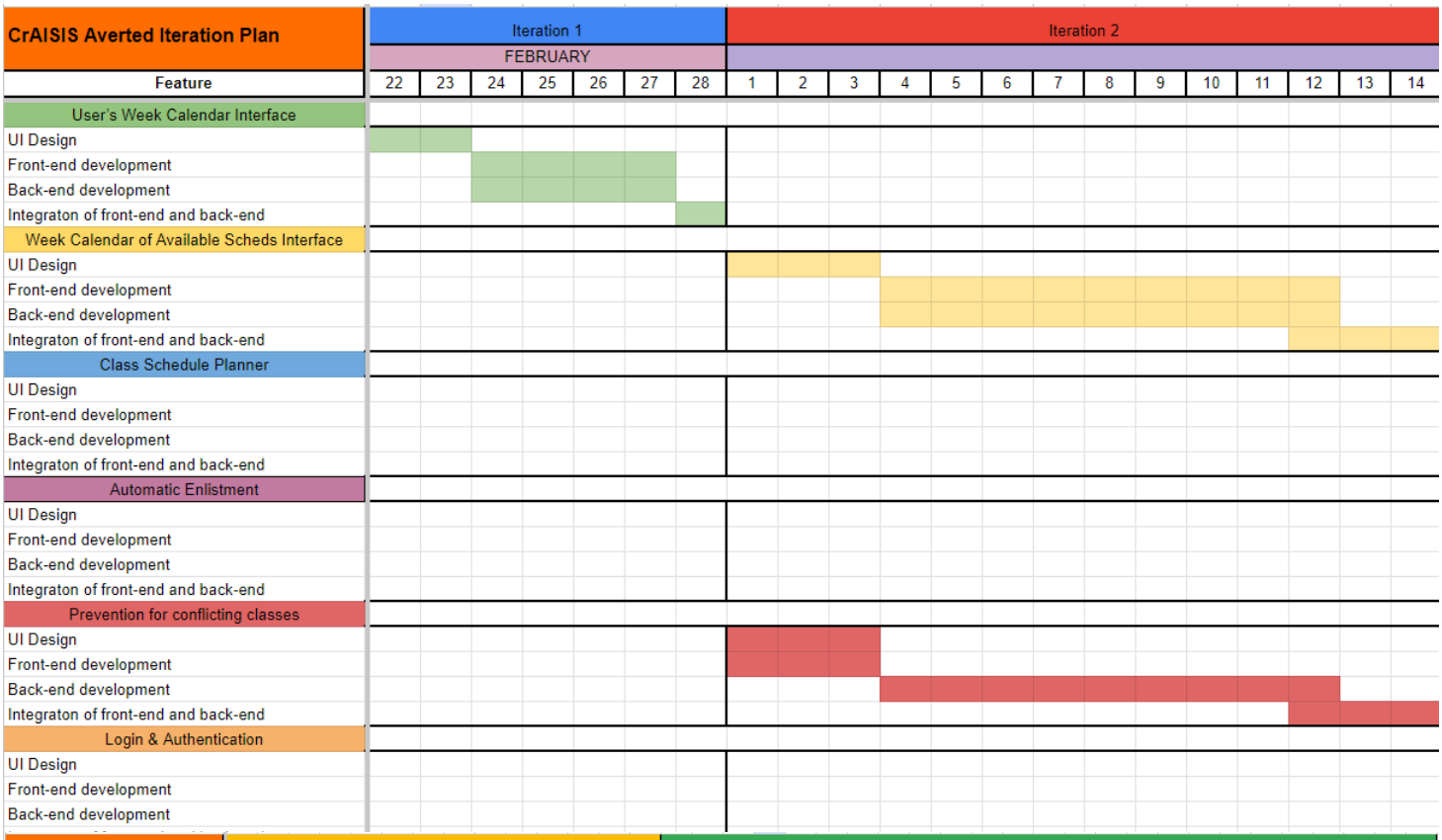
Attribute Name	Description	Primary Key?	Default Value	Type of Values	Can be null?
department_id	This is the unique identifier for the ID of the department.	Y	0	int	N
department_name	This is the name of the department.	N	“ ”	string	N

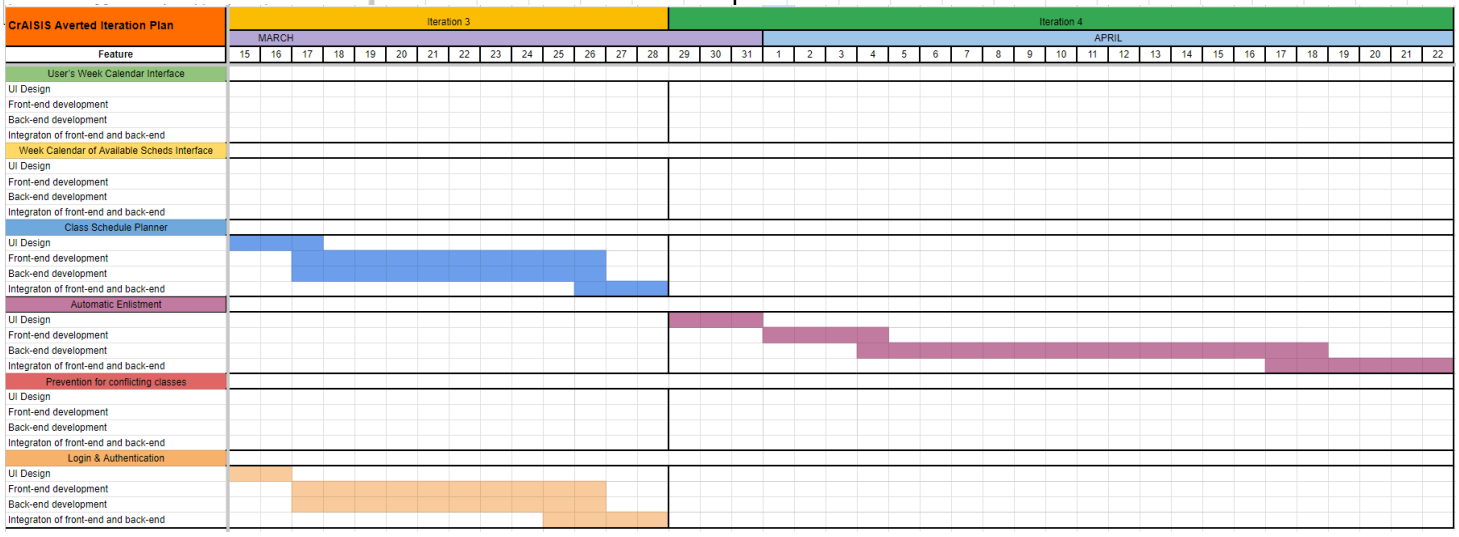
5. ITERATION PLAN

5.1 Work Item Assignment Features

Name / Description	Priority	Iteration	Assigned To	Effort estimate (hours)
User's Week Calendar Interface	High Priority	1st	Mikael Giannes Bernardino	18
Week Calendar of Available Scheds Interface	High Priority	2nd	Elmo Lenard Gumapos	36
Prevention for conflicting classes	High Priority	2nd	Mario Franco Deuna	36
Class Schedule Planner	Medium Priority	3rd	Joseph Izon	36
Automatic Enlistment	Low Priority	4th	Luis Mikhael Gonzalez	54
Login & Authentication	Low Priority	3rd	Chino Tesoro	36

5.2 GANTT Chart





6. HUMAN INTERFACE DESIGN

The screenshots below are the tentative design of the currently being developed software:

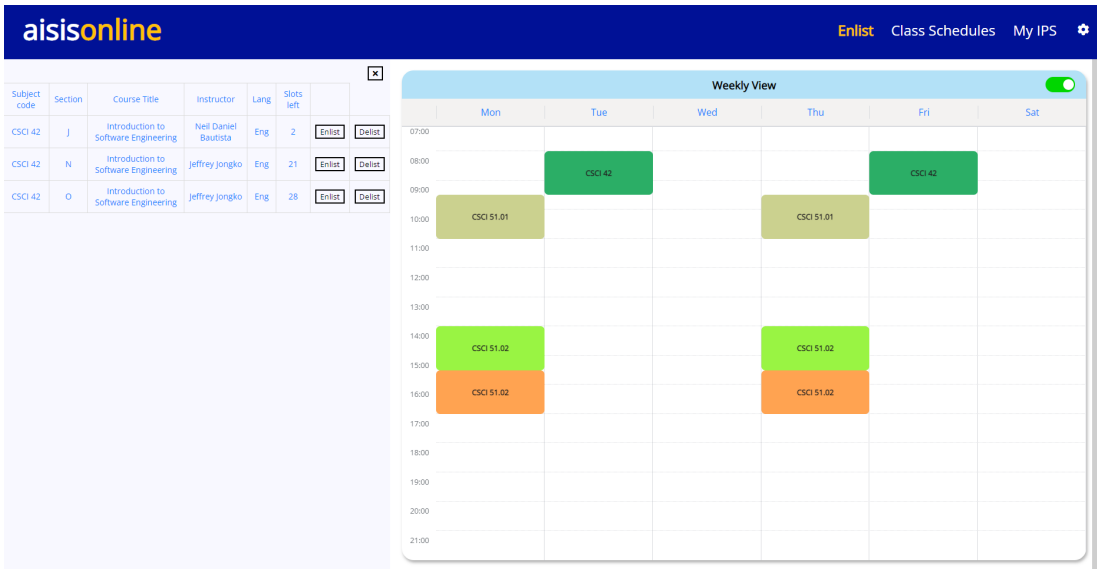
6.1 Overview of User Interface

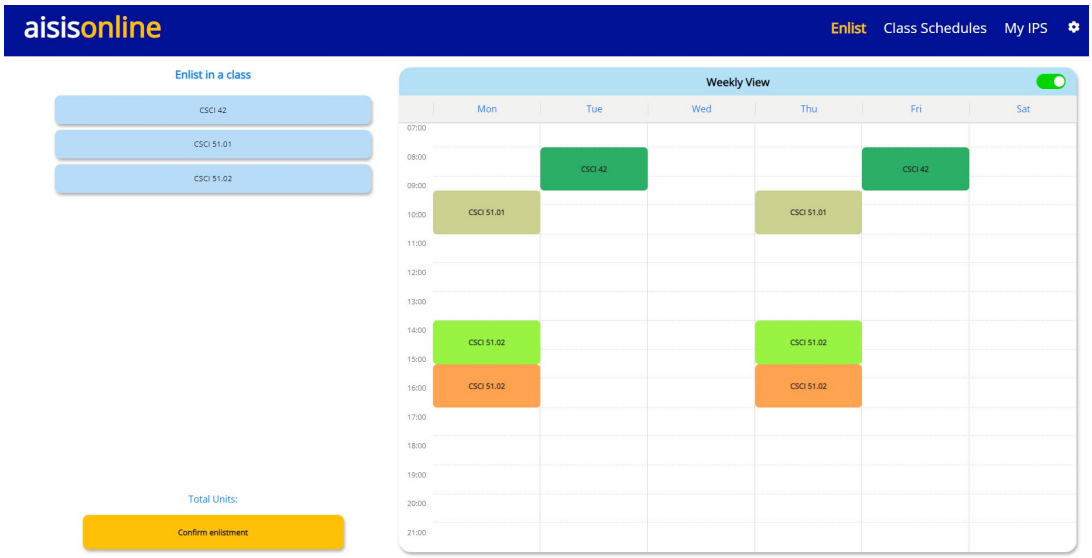
TBD.

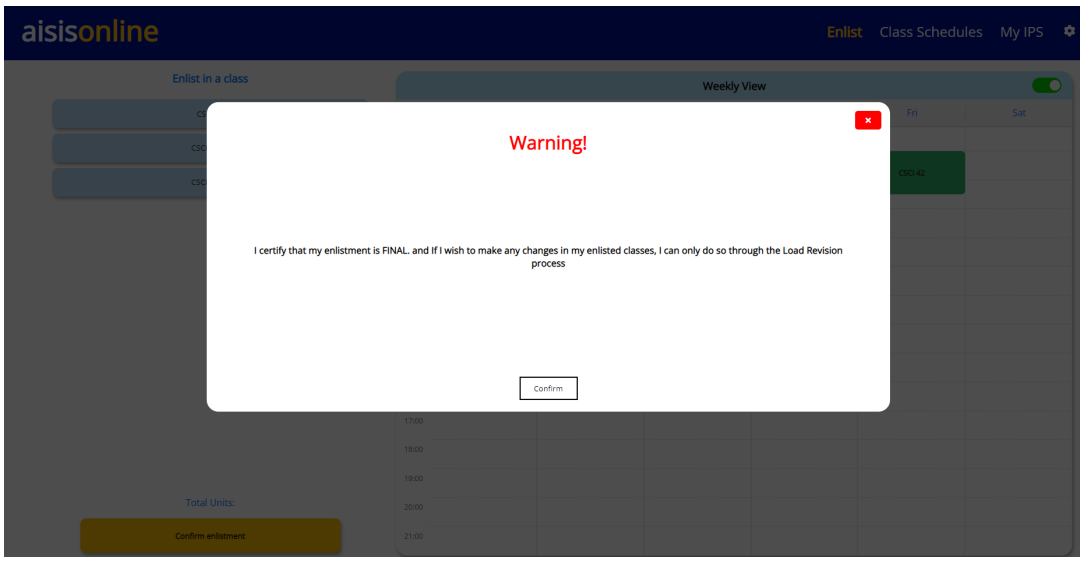
6.2 Screen Images

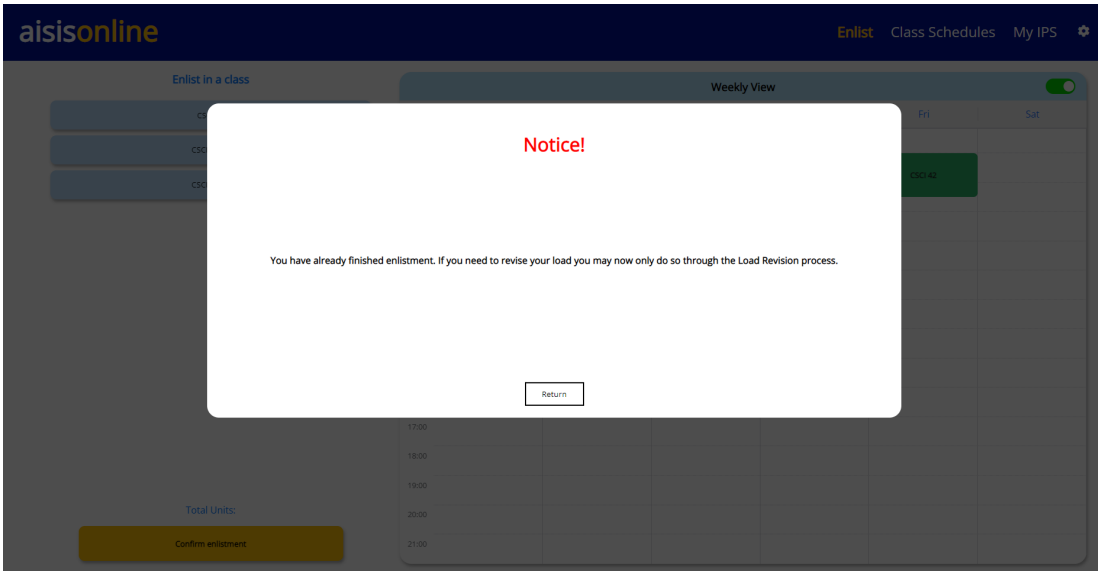
Login Page

Enlistment Page

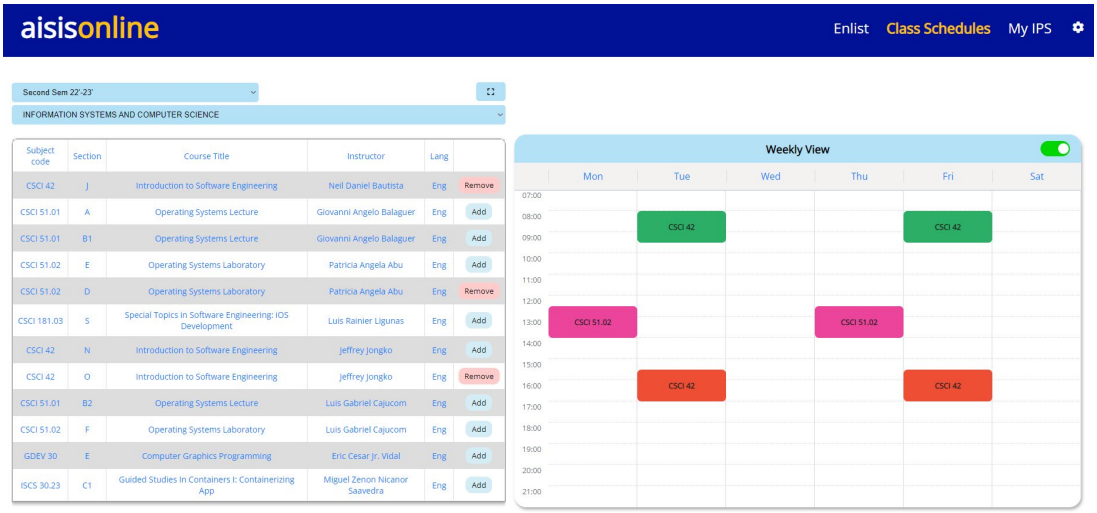






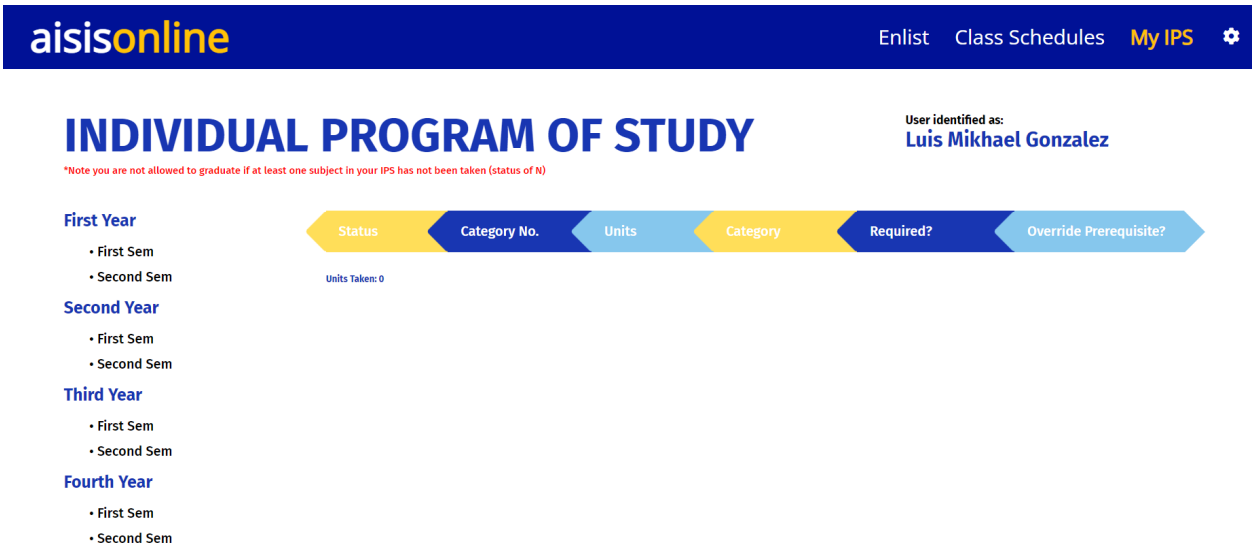


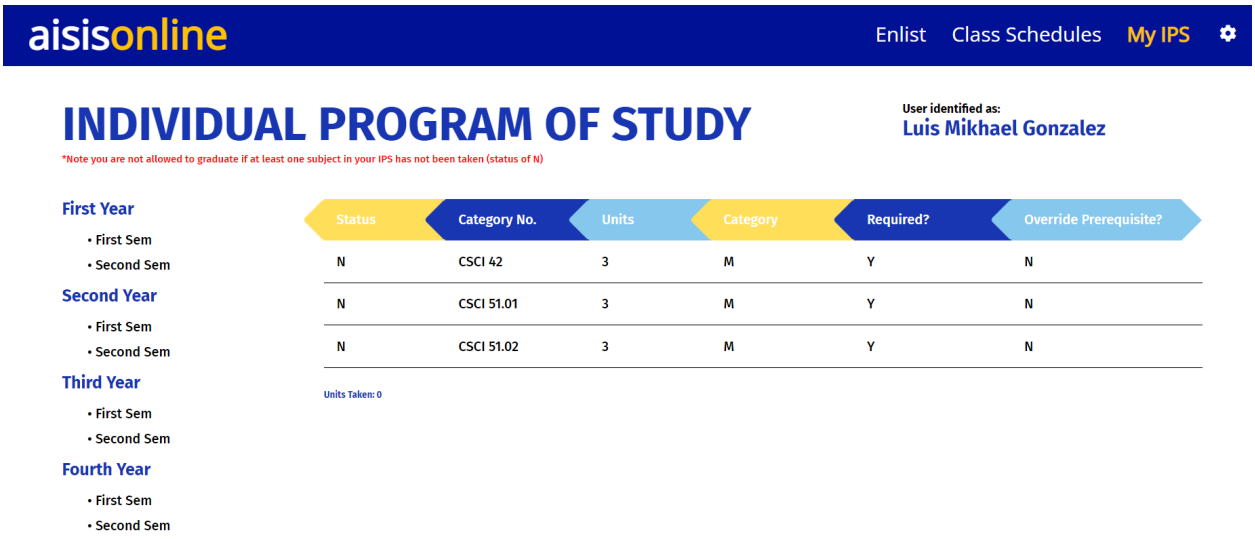
Class Schedule Planner Page



Subject code	Section	Course Title	Instructor		Room	Time	Units	Max Slots
CSCI 42	J	Introduction to Software Engineering	Neil Daniel Bautista	Eng	F-204	TF 08:00	3	25
CSCI 51.01	A	Operating Systems Lecture	Giovanni Angelo Balaguer	Eng	F-227	MTh 08:00	3	25
CSCI 51.01	B1	Operating Systems Lecture	Giovanni Angelo Balaguer	Eng	F-227	MTh 09:30	3	25
CSCI 51.02	E	Operating Systems Laboratory	Patricia Angela Abu	Eng	F-227	MTh 14:00	3	25
CSCI 51.02	D	Operating Systems Laboratory	Patricia Angela Abu	Eng	F-227	MTh 12:30	3	1
CSCI 181.03	S	Special Topics in Software Engineering: IOS Development	Luis Rainier Ligunas	Eng	CTC 201B	W 09:00	3	25
CSCI 42	N	Introduction to Software Engineering	Jeffrey Jongko	Eng	F-204	TF 14:00	3	30
CSCI 42	O	Introduction to Software Engineering	Jeffrey Jongko	Eng	F-204	TF 15:30	3	30
CSCI 51.01	B2	Operating Systems Lecture	Luis Gabriel Cajucom	Eng	CTC 214	MTh 09:30	3	30
CSCI 51.02	F	Operating Systems Laboratory	Luis Gabriel Cajucom	Eng	F-227	MTh 15:30	3	30
GDEV 30	E	Computer Graphics Programming	Eric Cesar Jr. Vidal	Eng	CTC 112	MTh 14:00	3	35
ISCS 30.23	C1	Guided Studies in Containers I: Containerizing App	Miguel Zenon Nicanor Saavedra	Eng	TBA	MTh 11:00	1	30

Individual Program of Study Page





6.3 Screen Objects and Actions

TBD.

7. REQUIREMENTS MATRIX

System Features

FEATURE 1: User's Week Calendar Interface	
Description and Priority	When the User starts the enlistment process, they will see a page with a Week Calendar from Mon-Sat, that has fillable slots within the hours of 8:00 AM-8:00 PM. This will show the User a better visualization of how their Weekly schedule could look like, instead of just a table of text that the current AISIS implements. This is a high priority.
Functional Requirements	**Non-Functional Requirements**
- Must be able to hold instances of various schedules of different classes - User must be able to interact with it - The schedule must reflect after the instance of enlistment	- Must be comprehensive and easily understood
FEATURE 2: Week Calendar of Available Scheds Interface	
Description and Priority	A similar week calendar that can be viewed on the same page as the user's week calendar. Each subject in the user's IPS has a weekly calendar of its own, that contains all available slots in their corresponding schedule. The user will be able to take schedules from this calendar, to transfer to their own schedule. This is a high priority.
Functional Requirements	**Non-Functional Requirements**
- Must have different instances for each subject in the student's IPS - Each instance holds the available schedules of the subject in the week that the student can choose from. - Must be able to load in the correct classes in which the student must	- Must appear on one page during the enlistment process without looking cramped - Must be smooth in transitioning into the screen - Must be readable despite the numerous amounts of classes to choose from

enlist in	
FEATURE 3: Class Schedule Planner	
Description and Priority	The user can view all the class schedules available in the system and they may freely add or remove it to their planner in order to create their desired class schedule in time for enlistment season. This feature will be of medium priority.
Functional Requirements	**Non-Functional Requirements**
- Add or remove a specific class into the weekly planner - See all information regarding the specific class one is interested in (e.g. max slots, professor, room) - See the schedule reflected on the weekly planner	- Must be responsive when interacted by the User - Must be able to visualize the class schedule clearly through the weekly calendar view
FEATURE 4: Automatic Enlistment	
Description and Priority	If the user is in a rush to complete their enlistment, there is an option for an automatic enlistment. It will automatically fill up the user's weekly calendar according to their IPS, with random schedules so long as they don't conflict. This is of low priority.
Functional Requirements	**Non-Functional Requirements**
- Randomly searches for available schedules according to the student's IPS, and essentially build the student's schedule for themselves	- Must be fast and automatic when assigning classes in order to fulfill its purpose of random hassle-free enlistment - Must clearly display results after use - Must be efficient and easy to use
FEATURE 5: Prevention for conflicting classes	
Description and Priority	When there are schedules that conflict with already chosen slots, or those schedules have run out of slots, they will be grayed out and the

	user will not be able to freely transfer them to their calendar. This is a medium priority.
Functional Requirements	**Non-Functional Requirements**
- Takes notes of schedules that are not available to the student, (conflicting classes, no slots left, etc.) and disables their movement. - Takes into consideration the already enlisted classes to avoid conflicting schedules. - Notifies the user if the class being attempted to enroll conflicts with an existing schedule.	- Must be easily distinguishable in order to prevent confusion and potential panic. - Notification of conflict must be urgently made known to the user.
FEATURE 6: Login & Authentication	
Description and Priority	A basic page that asks for the User's login credentials that once placed, allows the user to enlist. This is a low priority.
Functional Requirements	**Non-Functional Requirements**
- Takes in user credentials before providing access to the enlistment	- Must be specific and strict with inputs as it takes credentials that users should keep secure. - Does not display the password as is.

8. APPENDICES

8.1 Appendix A: Glossary

TBD.

8.2 Appendix B: Analysis Models

TBD.

8.3 Appendix D: Repository

- GitLab: https://gitlab.discs.ateneo.edu/Mikael/craisis-averted