

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

SCSE22-0184
Full-Stack Web Development for
Auto-Assessment Platform

Final Report

Author

Liu Wing Lam (U1940771B)

Project Supervisors

Dr Loke Yuan Ren

Dr Ng Keng Meng

Examiner

Assoc Prof Liu Weichen

Submitted in Partial Fulfilment of the Requirements of the Degree of Bachelor of Science in
Mathematical and Computer Sciences (Double Major) of the Nanyang Technological
University

School of Computer Science and Engineering
School of Physical and Mathematical Sciences
2023

Abstract

In the recent decade, prevalence of technology has increased the adoption of online assessment platforms. The popularity of these platforms among students and graders is the result of convenience, consistency, and accuracy. Furthermore, the serious outbreak of the Covid-19 global pandemic forced many institutions to accept modern technology to facilitate learning.

At Nanyang Technological University (NTU), the in-house Auto-Assessment Platform (AASP) was built to serve as an alternative to in-person examinations and tests so that students and invigilators do not have to gather at a physical venue. It had basic functionalities of an online coding platform. However, AASP was insufficient to be an alternative or a substitute to in-person assessments. There was an absence of proctoring to enforce a monitored and strict test environment. Hence, AASP was deemed not ready to be used for actual University assessments.

To make AASP production-ready, proctoring features were designed and implemented. Educators and lab assistants can now monitor assessment sessions through snapshots of students taken at random intervals during their assessment attempt. Students' behaviour and environment throughout the assessment will be recorded and can be reviewed after the attempts are completed. Cheating can also be deterred and prevented by having auto force submission if unusual browser tab switching has been identified. Examinations and tests can now be conducted and proctored remotely without requiring students and invigilators to gather at a physical venue.

To ensure AASP student accounts are more secure, email features were added for the ease of disseminating securely and randomly generated passwords upon student account creation and password reset. Information on published assessment is also conveniently made available to students via email.

Acknowledgment

I would like to express my deepest appreciation and gratitude to my project supervisor, Dr Loke Yuan Ren, whose contribution in guiding me and giving suggestions helped me with the implementations and in writing this report.

Table of Contents

Abstract.....	ii
Acknowledgment.....	iii
Table of Contents	iv
Nomenclature	vi
Table of Figures.....	vii
1 Introduction.....	1
1.1 Background	1
1.2 Prior Work.....	1
1.3 Problem Statement	2
1.4 Objective	2
1.5 Scope	2
2 Literature Review	3
2.1 Related Works	3
2.2 Proctoring Features.....	4
3 Project Plan	7
4 Design Methodology.....	10
4.1 Use Case Diagram	10
4.2 Functional Requirements.....	11
4.2.1 Toggling Proctoring Features	11
4.2.2 Capturing Candidate Snapshots.....	11
4.2.3 Force Submitting Assessment after Multiple Tab Switching.....	12
4.2.4 Email Notifications	13
4.3 Non-functional Requirements	13
4.4 Activity Diagrams	14
4.4.1 Toggling Proctoring Features	14
4.4.2 Capturing Candidate Snapshots.....	14
4.4.3 Force Submitting Assessment after Multiple Tab Switching.....	16
4.4.4 Email Notifications	17

4.5 Database (DB) Design.....	18
4.6 User Interface (UI) Design	19
4.6.1 Assessment Creation and Update	19
4.6.2 Proctoring Features During Assessment	20
4.6.3 Viewing Candidate Snapshots on Assessment Attempt Report	21
5 Implementation	23
5.1 Overview	23
5.2 Proctoring Features.....	25
5.2.1 Capturing Candidate Snapshots.....	25
5.2.2 Force Submitting Assessment after Multiple Tab Switching	29
5.3 Email Notifications.....	31
5.3.1 Student Passwords	31
5.3.2 Published Assessments.....	33
6 Code Refactors & Standardisations	34
6.1 HTTP Response Status Codes	34
6.2 Prints & Logs.....	34
7 Challenges Faced.....	35
7.1 Debugging Existing Bugs.....	35
7.3 Bringing InsightFace Detection Model Offline.....	35
8 Future Works	37
8.1 More Question Types	37
8.2 AutoSave Code.....	37
8.3 Configure Proper Email Host	37
8.4 Get Domain to Host AASP Application.....	37
8.5 Facial Recognition.....	37
9 Conclusion	38
Bibliography	I
Appendix.....	III

Nomenclature

AA	Automatic Assessment
AASP	Auto-Assessment Platform
API	Application Programming Interface
DB	Database
ERD	Entity Relationship Diagram
HTTP	Hypertext Transfer Protocol
LFS	Large File Storage
LMS	Learning Management System
MCQ	Multiple-Choice Questions
NTU	Nanyang Technological University
ORM	Object Relational Mapper
SCSE	School of Computer Science and Engineering
SSL	Secure Sockets Layer
UI	User Interface
URL	Uniform Resource Locator
WBS	Work Breakdown Structure

Table of Figures

Figure 1 HackerRank: Copy/Paste Tracking Toggle Setting	4
Figure 2 HackerRank: Test Browser Prompting to allow Webcam Access	4
Figure 3 HackerRank: Candidate Snapshots on Test Evaluation Report	5
Figure 4 HackerEarth: Prevent Tab Switching	5
Figure 5 ProProfs Quiz Maker: Disable Tab Switching	6
Figure 6 Project Gantt Chart	7
Figure 7 Project Gantt Chart (continued).....	8
Figure 8 Project WBS	9
Figure 9 Use Case Diagram	10
Figure 10 Activity Diagram for Changing Proctoring Settings	14
Figure 11 Activity Diagram for Student on Assessment Landing Page	14
Figure 12 Activity Diagram for System Detecting Faces on Initial Student Photo	15
Figure 13 Activity Diagram for System Capturing Candidate Snapshots During Assessment	15
Figure 14 Activity Diagram for Educator Viewing Snapshots on Attempt Report	15
Figure 15 Activity Diagram for Force Submission of Assessment after Multiple Tab Switching	16
Figure 16 Activity Diagram of Student Account Creation	17
Figure 17 Activity Diagram of Student Reset Password	17
Figure 18 Activity Diagram of Published Assessment	17
Figure 19 ERD of Relevant DB Schemas	18
Figure 20 Wireframe of Educator UI to Create a New Assessment	19
Figure 21 Wireframe of Student UI to Capture Initial Snapshot Before an Assessment	20
Figure 22 Wireframe of Student UI of Tab Switching Warning During an Assessment Attempt	20
Figure 23 Wireframe of Educator UI Viewing Students' Assessment Best Attempt.....	21
Figure 24 Wireframe of Educator UI Viewing a Student's Assessment Attempts.....	21
Figure 25 Wireframe of Educator UI to View Candidate Snapshots in Assessment Attempt Report.....	22
Figure 26 System Architecture	23
Figure 27 System Prompting Student for Browser Webcam Access	25

Figure 28 Student Taking Initial Snapshot with Matriculation Card.....	26
Figure 29 Educator Viewing Assessment Attempts of Students	27
Figure 30 Educator Viewing a Student's Attempts	27
Figure 31 Educator Viewing a Student's Attempt Snapshots.....	28
Figure 32 Tab Switching Warning When Student Leaves Assessment Screen	29
Figure 33 Tab Switching Timer after Student Leaves Screen Twice	30
Figure 34 Email Sent to Student upon Account Creation.....	31
Figure 35 Email Sent to Student Upon Password Reset by Educator or Lab Assistant	32
Figure 36 Student Change Password Interface after Clicking URL in the Email	32
Figure 37 Educator Published an Assessment	33
Figure 38 Email Notification Sent to Student upon Publishing Assessment.....	33
Figure 39 Error Downloading Face Detection Model in AASP Celery Worker Container	35
Figure 40 Accuracy of InsightFace Pretrained Face Detection Models	36

1 Introduction

1.1 Background

With the growing number of students enrolling in programming courses, it is increasingly difficult for manual grading of programming assessments [1]. Manual grading of programming assignments is tedious, time-consuming, and prone to mistakes and bias [2] [3]. Different human graders may give different evaluations to the same solution due to fatigue, inconsistency, etc. The more students there are, the more human graders are required to complete grading in time, resulting in inconsistency in grading [4].

In the recent decade, prevalence of technology has increased the adoption of online-based platforms at several institutions. The popularity of these platforms among students and graders is the result of convenience. Furthermore, assessments and assignments can be graded more accurately and consistently [5]. With the serious outbreak of the global pandemic Covid-19, institutions were made to go from offline mode to online mode of learning, forcing many to accept modern technology. Since then, online education has increased exponentially [6].

At Nanyang Technological University (NTU), Blackboard is used as the main Learning Management System (LMS) to manage courses and administer tests. However, coding question types are not supported on this platform [7]. Hence, HackerEarth, a third-party online coding platform, is used for some of the coding assignments and assessments. Although this platform has a myriad of features for technical skill assessments, insourcing the assessment platform would allow full control over the system. The university would be able to comply with its standards and better meet its needs.

1.2 Prior Work

The first AASP was developed by a student, Soh Yan Quan, Kenneth [8]. This version was then enhanced by another student Yap Guan Sheng [9]. Thereafter, AASP was redone by student Lee Jun Wei so that the platform can be redesigned and reimplemented to increase maintainability [10].

1.3 Problem Statement

AASP uses Judge0 to conduct a dynamic analysis on students' submitted codes, which is useful to grade most of the University's coding assessments. While AASP has the basic functions of an online coding platform, it does not possess any monitoring features for invigilation. This deems AASP insufficient to replace or act as an alternative to in-person University assessments. Proctoring features are necessary to enforce a strict and monitored test environment for actual University assessments.

1.4 Objective

This project's objective is to design and implement test proctoring features to monitor students' assessment session, and to deter and/or prevent cheating. Unusual activity or violations should be flagged out so that educators can review them after the assessment is completed. Having such proctoring features will allow assessments to be administered remotely, without having students and invigilators to gather at a physical location in the University. Furthermore, manpower required to proctor in-person assessments can be reduced or omitted.

1.5 Scope

The scope of the project includes:

- To study and understand the needs of proctoring features in an online assessment platform.
- To analyse the existing online assessment platforms in the market.
- To implement proctoring features to enforce a monitored assessment environment.
- To implement features that will enhance AASP.

2 Literature Review

2.1 Related Works

To improve AASP, strengths and weaknesses of existing products in the market must be studied.

Helmick [11] stated that various approaches have been used in Automatic Assessment (AA), with most systems adopting the approach of analysing the output of programs. However, there seems to be no established consensus on the best way to grade students' codes in AA. Ullah et al [12] and Fonte et al [1] outline three types of AA methodologies – dynamic, static and hybrid analysis. Dynamic analysis refers to assessing programs by executing students' programs through a series of pre-defined test cases and comparing the actual and expected output. Static analysis involves using a fixed set of software metrics to analyse the code. This method grades the assessment without having to execute or compile students' codes. Hybrid analysis is the combination of dynamic and static analysis in the same AA system.

After studying the types of AA methodologies, dynamic analysis is the most suitable for AASP since it has the most use case in the University. Static analysis is not useful for most undergraduate courses.

Other than AA methodologies, products related to online assessment with candidate monitoring features must also be analysed to provide insight to enhance AASP.

One monitoring feature is using browser lockdown. A popular platform that implements this feature is Gradescope [13]. Gradescope utilises LockDown Browser, powered by Respondus [14] in the beta version. Using the browser lockdown feature invigilates tests by only allowing student to open one window of the secured browser, and only one tab of the LMS can be opened. By doing so, students will not be able to refer to online materials on the same machine. However, the test system must be integrated into the LMS.

Another monitoring feature is proctoring. This feature means that tests are invigilated by monitoring candidates' focus on the test screen while attempting tests, to ensure the integrity of the test environment. Any suspicious activity will be identified and flagged out to instructors.

Monitoring features that are implemented in existing platforms include disabling or tracking of copy/paste, tab proctoring and recording of candidate snapshots periodically.

2.2 Proctoring Features

An example of a product with such features is HackerRank. HackerRank is a commercial and competitive programming platform that has an enterprise-side product, HackerRank for Work, which allows companies to screen potential candidates based on their technical skills.

Figure 1, 2, 3 are some screenshots of the features on HackerRank taken from an article on HackerRank's website [15].



Figure 1 HackerRank: Copy/Paste Tracking Toggle Setting

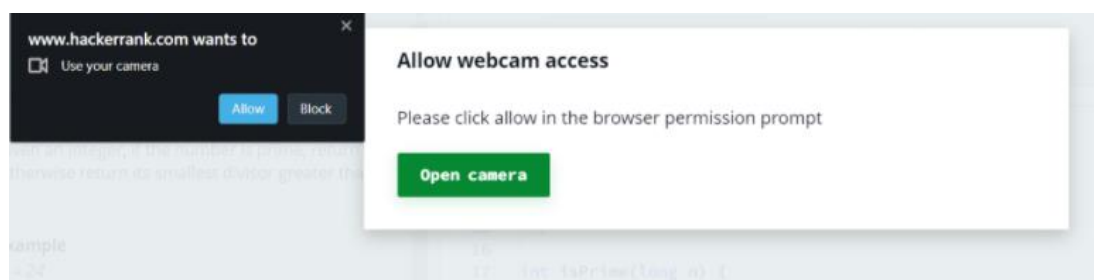


Figure 2 HackerRank: Test Browser Prompting to allow Webcam Access

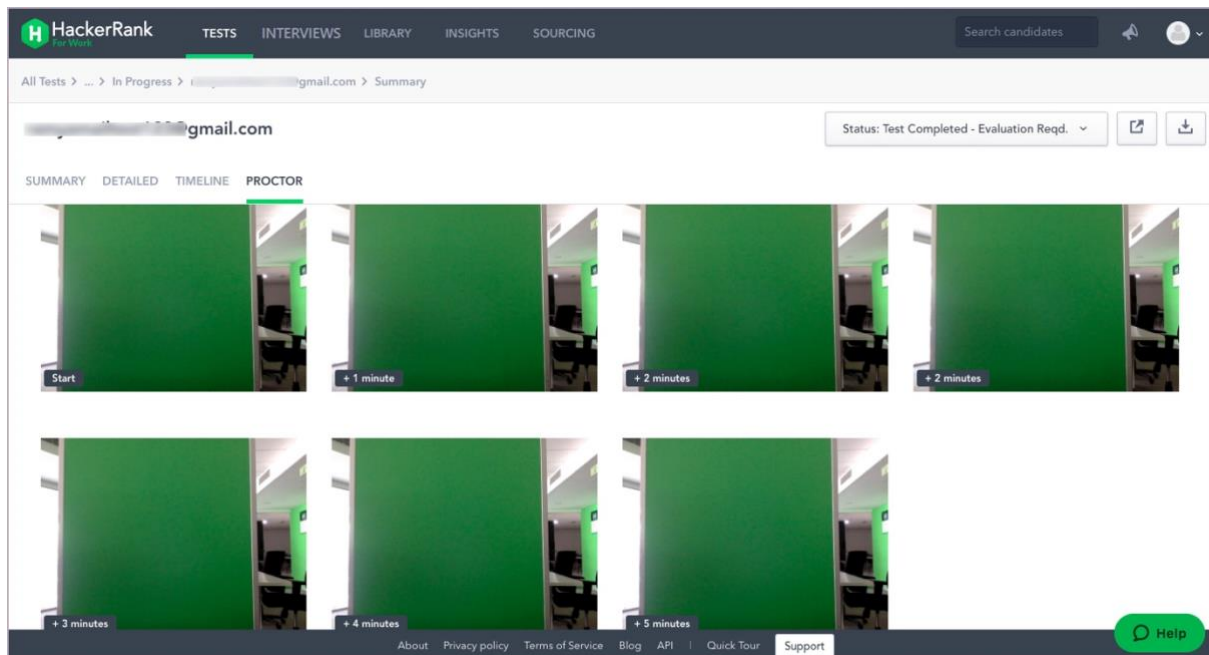


Figure 3 HackerRank: Candidate Snapshots on Test Evaluation Report

Another product with monitoring features is HackerEarth. Like HackerRank for Work, HackerEarth is a software company that provides an enterprise solution to companies with technical hiring.

Figure 4 is a screenshot of the tab monitoring feature implemented, taken from HackerEarth's website [16].

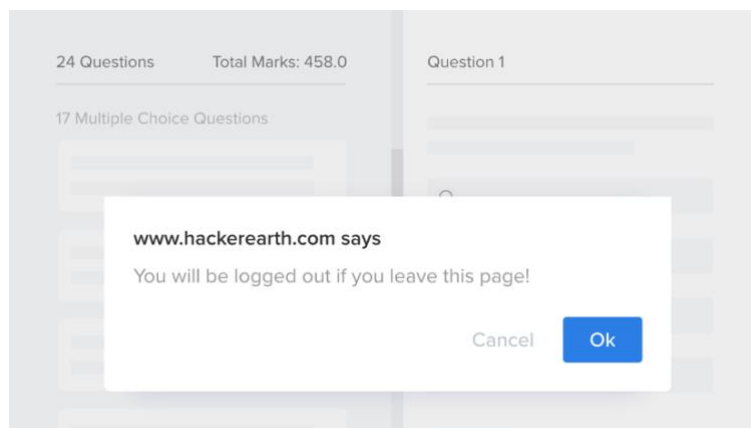


Figure 4 HackerEarth: Prevent Tab Switching

ProProfs is a software company that provide smart tools for professional online training and assessments such as tests and quizzes.

Figure 5 shows how tab switching is disabled by ProProfs in their Quiz Maker product [17].

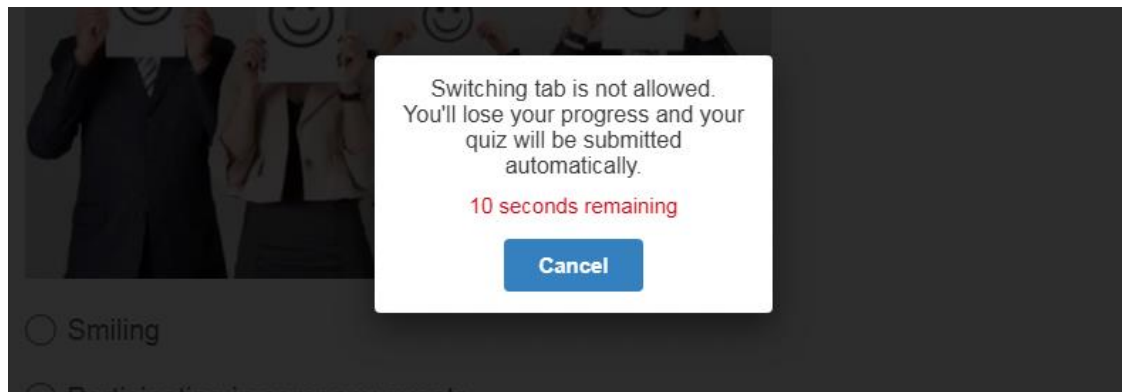


Figure 5 ProProfs Quiz Maker: Disable Tab Switching

3 Project Plan

Figure 6, 7 is the Gantt Chart of this project. The project is split into a few sections – Discussion & Research; Analysis & Planning; Implementation of Main Features; Implementation of Other Enhancements; Miscellaneous Code Refactors/Changes; Documentations. An iterative Agile approach was adopted so that continuous analysis, implementation, and improvements can be made. While implementing the main Proctoring features, more potential enhancements were found, leading to the idea of implementing email notification features.

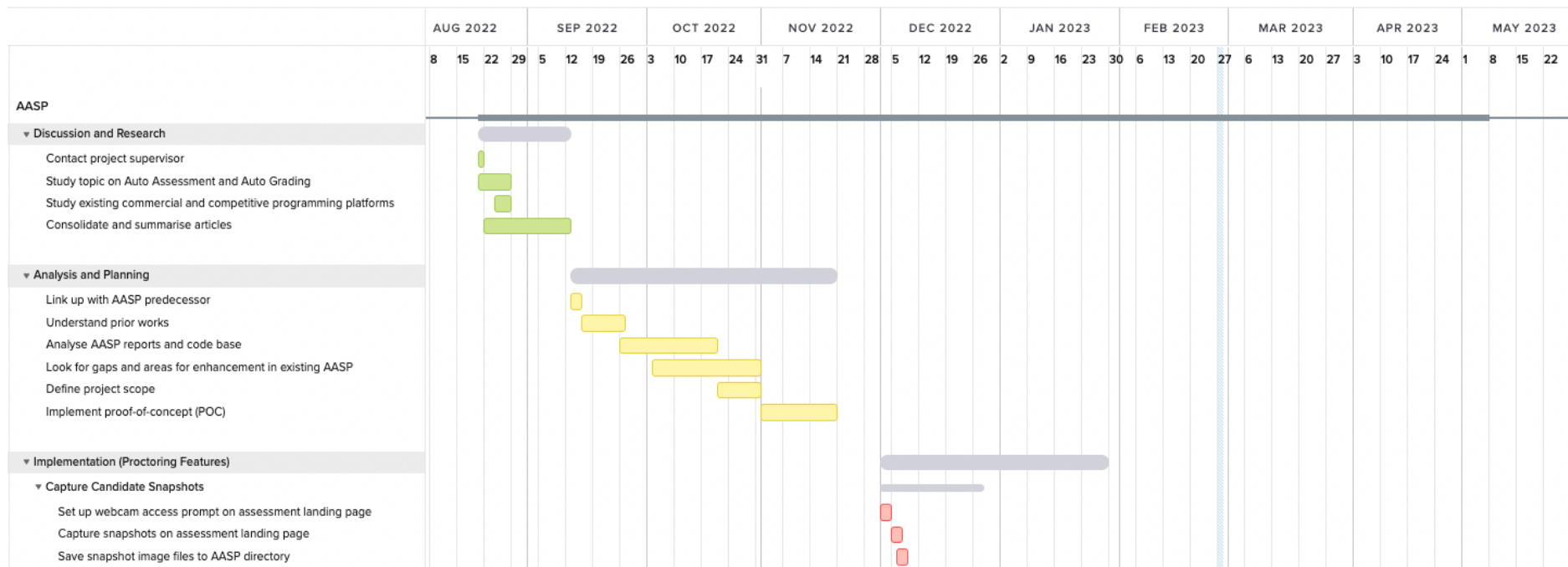


Figure 6 Project Gantt Chart

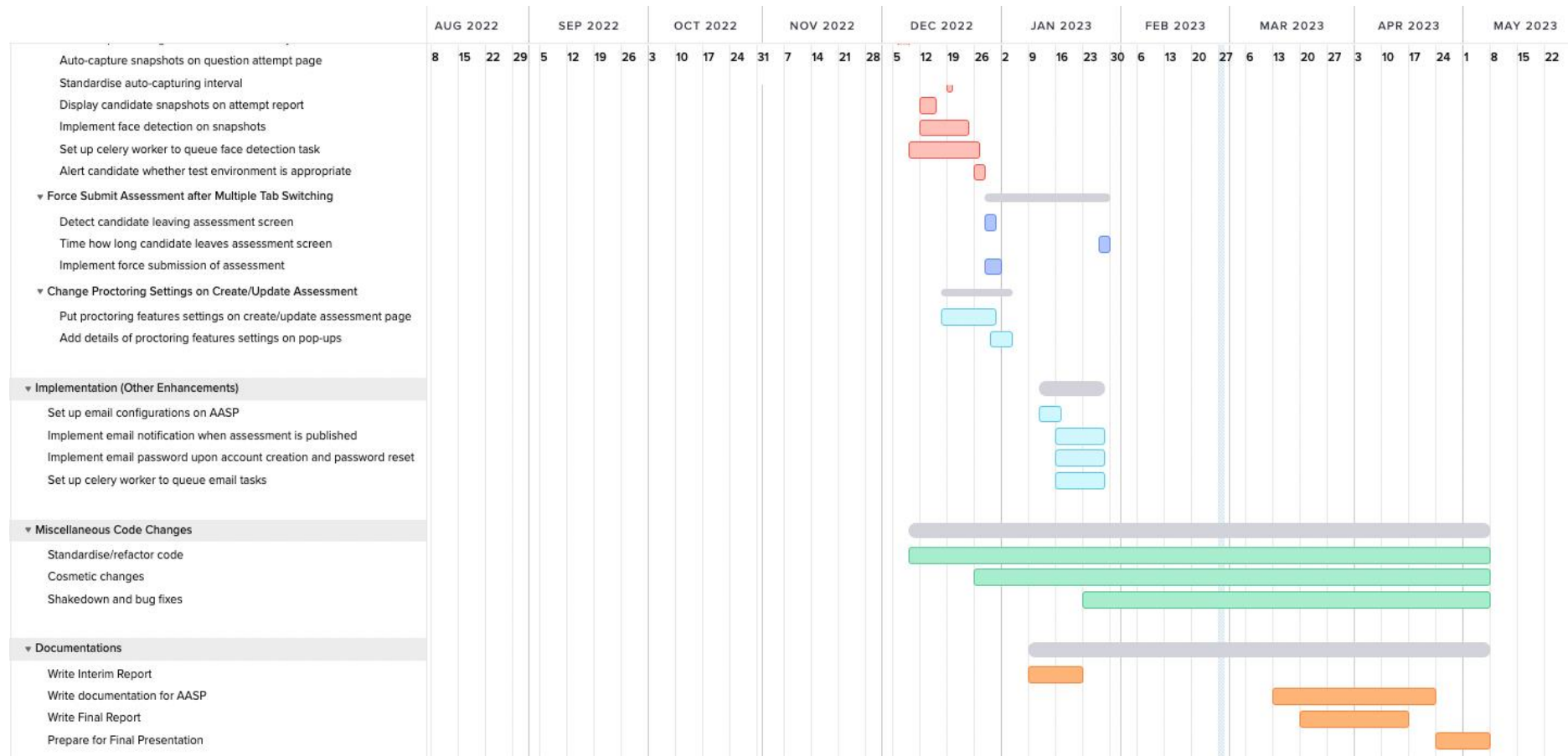


Figure 7 Project Gantt Chart (continued)

Figure 8 shows the Work Breakdown Structure (WBS) of this project.

WBS #	Name / Title	Type	Start Date	End Date
1	AASP	project	19/8/22	5/5/23
1.1	Discussion and Research	group	19/8/22	12/9/22
1.1.1	Contact project supervisor	task	19/8/22	19/8/22
1.1.2	Study topic on Auto Assessment and Auto Grading	task	19/8/22	26/8/22
1.1.3	Study existing commercial and competitive programming platforms	task	24/8/22	26/8/22
1.1.4	Consolidate and summarise articles	task	22/8/22	12/9/22
1.2	Analysis and Planning	group	13/9/22	18/11/22
1.2.1	Link up with AASP predecessor	task	13/9/22	14/9/22
1.2.2	Understand prior works	task	15/9/22	26/9/22
1.2.3	Analyse AASP reports and code base	task	26/9/22	19/10/22
1.2.4	Look for gaps and areas for enhancement in existing AASP	task	4/10/22	31/10/22
1.2.5	Define project scope	task	20/10/22	31/10/22
1.2.6	Implement proof-of-concept (POC)	task	1/11/22	18/11/22
1.3	Implementation (Proctoring Features)	group	1/12/22	27/1/23
1.3.1	Capture Candidate Snapshots	subgroup	1/12/22	27/12/22
1.3.1.1	Set up webcam access prompt on assessment landing page	task	1/12/22	2/12/22
1.3.1.2	Capture snapshots on assessment landing page	task	5/12/22	6/12/22
1.3.1.3	Save snapshot image files to AASP directory	task	6/12/22	7/12/22
1.3.1.4	Auto-capture snapshots on question attempt page	task	8/12/22	9/12/22
1.3.1.5	Standardise auto-capturing interval	task	19/12/22	19/12/22
1.3.1.6	Display candidate snapshots on attempt report	task	12/12/22	14/12/22
1.3.1.7	Implement face detection on snapshots	task	12/12/22	22/12/22
1.3.1.8	Set up celery worker to queue face detection task	task	8/12/22	26/12/22
1.3.1.9	Alert candidate whether test environment is appropriate	task	26/12/22	27/12/22
1.3.2	Force Submit Assessment after Multiple Tab Switching	subgroup	28/12/22	27/1/23
1.3.2.1	Detect candidate leaving assessment screen	task	28/12/22	29/12/22
1.3.2.2	Time how long candidate leaves assessment screen	task	26/1/23	27/1/23
1.3.2.3	Implement force submission of assessment	task	28/12/22	30/12/22
1.3.3	Change Proctoring Settings on Create/Update Assessment	subgroup	16/12/22	3/1/23
1.3.3.1	Put proctoring features settings on create/update assessment page	task	16/12/22	29/12/22
1.3.3.2	Add details of proctoring features settings on pop-ups	task	29/12/22	3/1/23
1.4	Implementation (Other Enhancements)	group	11/1/23	26/1/23
1.4.1	Set up email configurations on AASP	task	11/1/23	16/1/23
1.4.2	Implement email notification when assessment is published	task	16/1/23	26/1/23
1.4.3	Implement email password upon account creation and password reset	task	16/1/23	26/1/23
1.4.4	Set up celery worker to queue email tasks	task	16/1/23	26/1/23
1.5	Miscellaneous Code Changes	group	8/12/22	5/5/23
1.5.1	Standardise/refactor code	task	8/12/22	5/5/23
1.5.2	Cosmetic changes	task	26/12/22	5/5/23
1.5.3	Shakedown and bug fixes	task	23/1/23	5/5/23
1.6	Documentations	group	9/1/23	5/5/23
1.6.1	Write Interim Report	task	9/1/23	20/1/23
1.6.2	Write documentation for AASP	task	13/3/23	21/4/23
1.6.3	Write Final Report	task	20/3/23	14/4/23
1.6.4	Prepare for Final Presentation	task	24/4/23	5/5/23

Figure 8 Project WBS

4 Design Methodology

4.1 Use Case Diagram

Figure 9 is the use case diagram of the added AASP proctoring features, showing the key use cases of the actors involved in the new features.

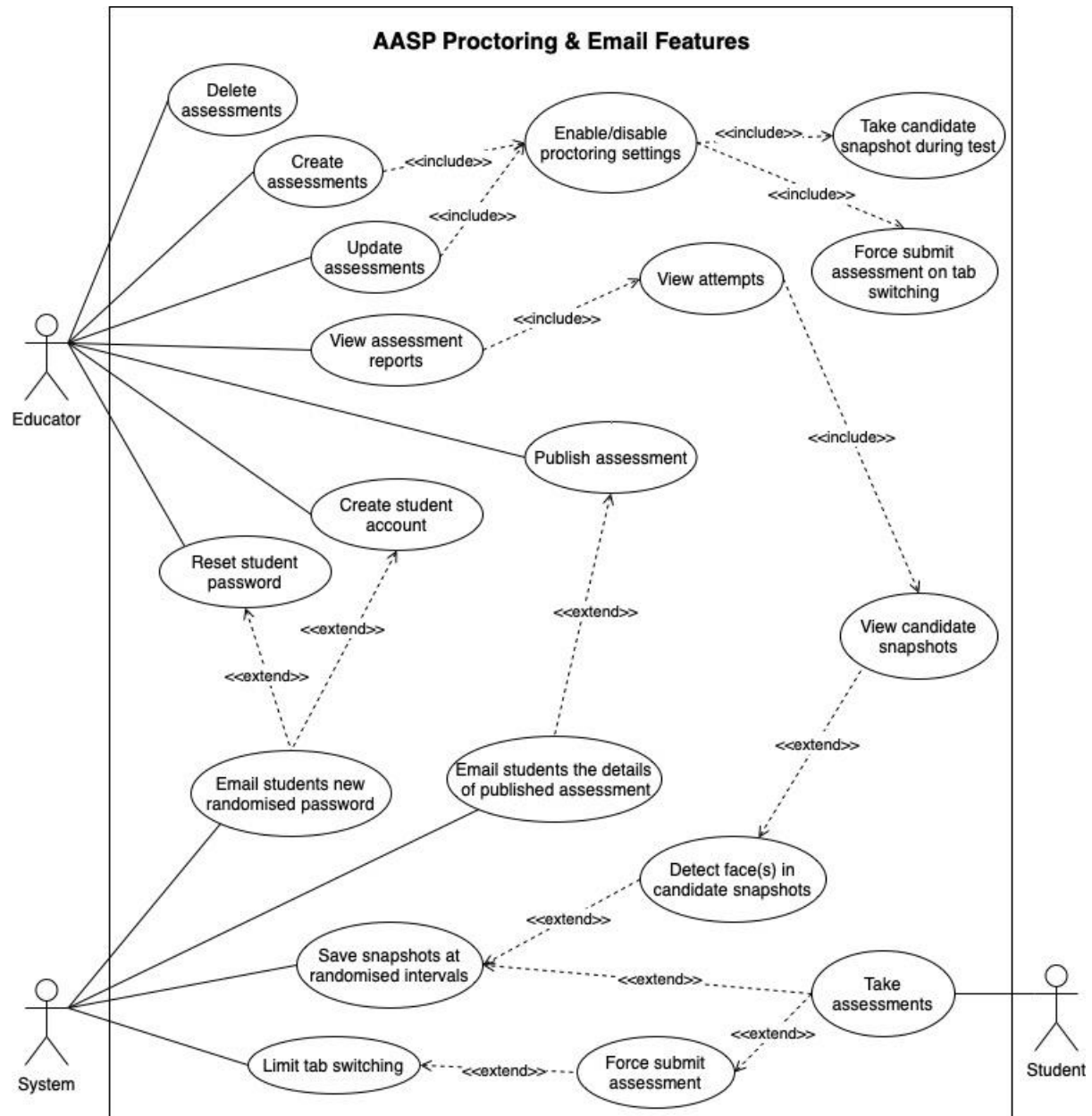


Figure 9 Use Case Diagram

4.2 Functional Requirements

This section will discuss the 2 main proctoring features – capturing candidate snapshots, and forcing submission of assessment after multiple tab switching, as well as email notification features.

4.2.1 Toggling Proctoring Features

1. The educator must be able to change proctoring settings when creating and updating assessments.

1.1. The educator must be able to enable or disable these 2 proctoring settings:

1.1.1. The capturing of candidate snapshots during the assessment.

1.1.1.1. The educator must be informed that enabling this setting will require candidates to have a functional webcam.

1.1.2. The force submission of assessment on tab switching.

1.1.2.1. The educator must be informed that students will be given a warning if they leave the assessment browser tab, and the assessment will be automatically submitted on the third tab switch.

4.2.2 Capturing Candidate Snapshots

1. If the candidate snapshot setting is enabled,

1.1. Before starting the assessment,

1.1.1. The browser must be able to prompt the student to allow camera access to the assessment platform site.

1.1.2. The student must be able to capture an initial photo of himself with his matriculation card.

1.1.2.1. The system must be able to alert the student whether his test environment is appropriate.

1.1.2.1.1. The system must be able to detect the number of faces in the initial photo taken. The number of faces must be exactly 2 for the student to start the assessment.

1.1.2.1.2. The system must not let the student know that it is not able to differentiate between a real face and a photo.

1.2. During the assessment,

- 1.2.1. The system must be able to capture and save snapshots of the candidate.
 - 1.2.1.1. The system must be able to randomise the auto-capturing interval between 1 minute and 10% of the assessment total duration.
 - 1.2.1.2. The system must be able to detect the number of faces present.
- 1.2.2. The educator must be able to view the snapshots on the assessment reports.
 - 1.2.2.1. The system must be able to categorise the snapshots according to number of faces detected.
 - 1.2.2.2. The system must be able to flag out snapshots that do not contain a face and those that contain more than 1 face.
- 2. If the candidate snapshot setting is disabled,
 - 2.1. Before starting the assessment, the browser must not prompt the student to allow camera access to the assessment platform site.
 - 2.2. During the assessment, the system must not capture any snapshots of the student during the assessment.

4.2.3 Force Submitting Assessment after Multiple Tab Switching

- 1. If force submission of assessment on leaving the assessment tab is enabled,
 - 1.1.1. The system must be able to give the student warnings when he leaves the assessment browser tab. The system must be able to give warnings for the first 2 times on leaving the assessment tab.
 - 1.1.2. The system must be able to identify any focus away from the assessment browser screen as leaving the tab. This includes clicking on windows, pop-ups and tabs that are not the assessment screen.
 - 1.2. The system must be able to force submit the student's assessment on his third time leaving the assessment tab.
 - 1.2.1. The system must not prompt the confirmation modal for leaving the assessment tab on force submission.
- 2. If force submission on leaving the assessment tab is not enabled,
 - 2.1. The system must not give any warnings to the student when he leaves the assessment tab.
 - 2.2. The system must not force submit the student's assessment on his third time leaving the assessment tab.

4.2.4 Email Notifications

1. The system must be able to send emails containing students' randomised password.
 - 1.1. This email must be sent when the educator or lab assistant creates an account for a student.
 - 1.2. This email must be sent when the educator or lab assistant resets a student's password.
2. The system must be able to send emails containing details of a published assessment.
 - 2.1. This email must be sent to students enrolled in the assessment course when the educator publishes an assessment.
 - 2.2. If there are existing published assessments, this email must be sent to newly enrolled students when the educator or lab assistant enrolls them to the course.

4.3 Non-functional Requirements

1. The system shall be able to support at least 600 students concurrently.
2. The system shall be compatible with most of the common web browsers.
 - 2.1. Web browsers include Google Chrome, Firefox, Safari, and Microsoft Edge.
3. The system shall be deployable on any Linux machine.
 - 3.1. The system shall be containerised and isolated from its environment setup.
4. The system shall have an average Application Programming Interface (API) response time of no more than 2 seconds.

4.4 Activity Diagrams

The activity diagrams below describe the steps in the use case diagram and are visual representations of the flow of functional requirements.

4.4.1 Toggling Proctoring Features

Educators can enable or disable the 2 proctoring features of an assessment during creation or update (Figure 10).

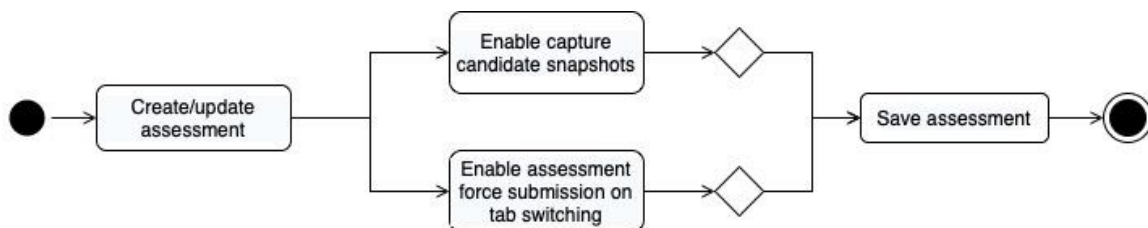


Figure 10 Activity Diagram for Changing Proctoring Settings

4.4.2 Capturing Candidate Snapshots

Activity diagrams for this feature will be split into 3 parts – student, system, and educator, to show how processes flow for each actor.

Student

Before the start of an assessment, the student views information about the assessment. If the capturing of candidate snapshots is enabled, the student will be prompted to take an initial photo with his matriculation card before entering the assessment (Figure 11).

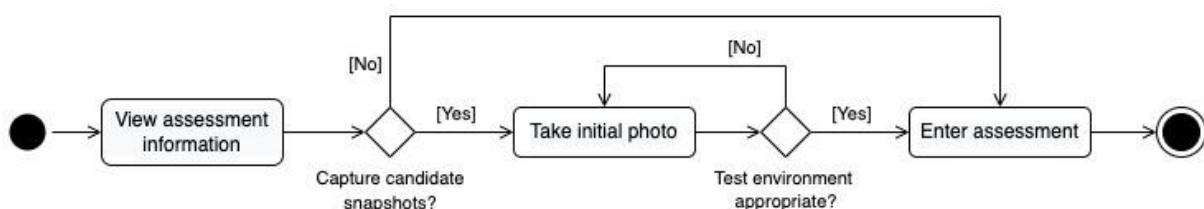


Figure 11 Activity Diagram for Student on Assessment Landing Page

System

Before the start of an assessment, if the monitoring feature is enabled, the system will detect the number of faces in the student's initial photo (Figure 12).

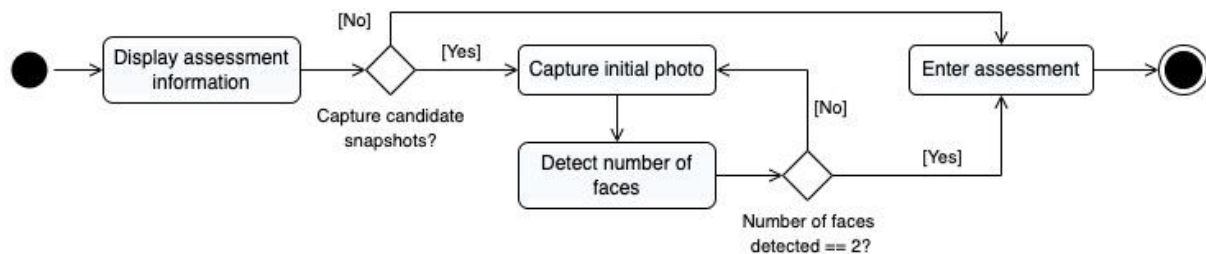


Figure 12 Activity Diagram for System Detecting Faces on Initial Student Photo

During the assessment, if the monitoring feature is enabled, the system will capture and process snapshots of the students through a flow of actions (Figure 13).

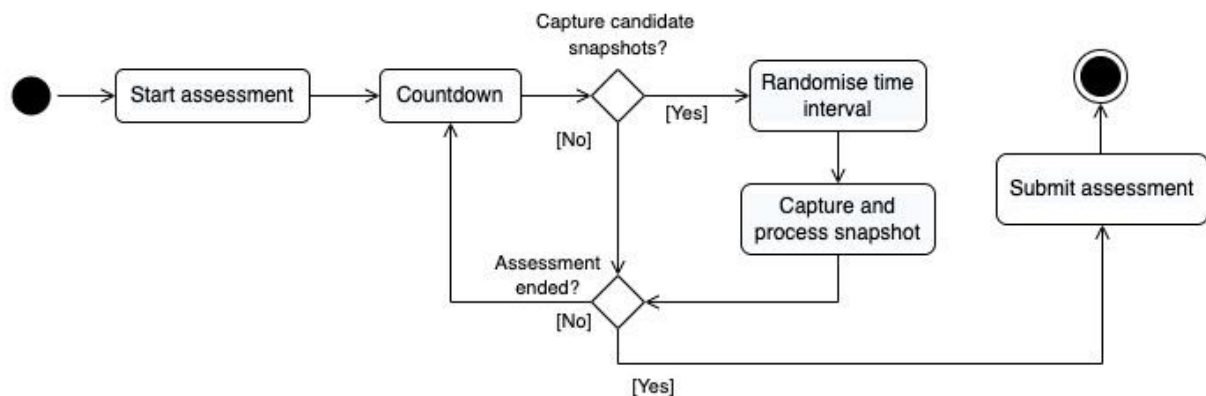


Figure 13 Activity Diagram for System Capturing Candidate Snapshots During Assessment

Educator

The educator can view captured snapshots on the students' assessment attempt report, where the snapshots are categorised based on number of faces detected (Figure 14).

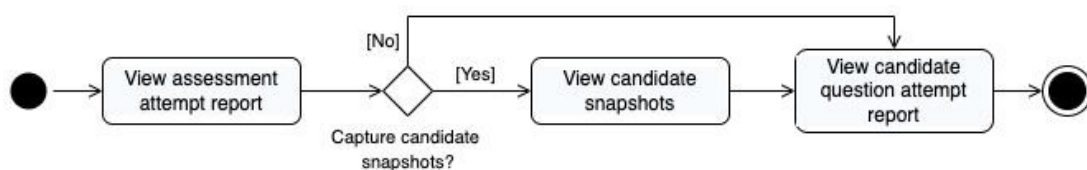


Figure 14 Activity Diagram for Educator Viewing Snapshots on Attempt Report

4.4.3 Force Submitting Assessment after Multiple Tab Switching

If force assessment submission is enabled, if the student leaves the assessment screen for more than 2 times, a countdown timer will be started. When the time is up, his assessment will be forcefully submitted (Figure 15).

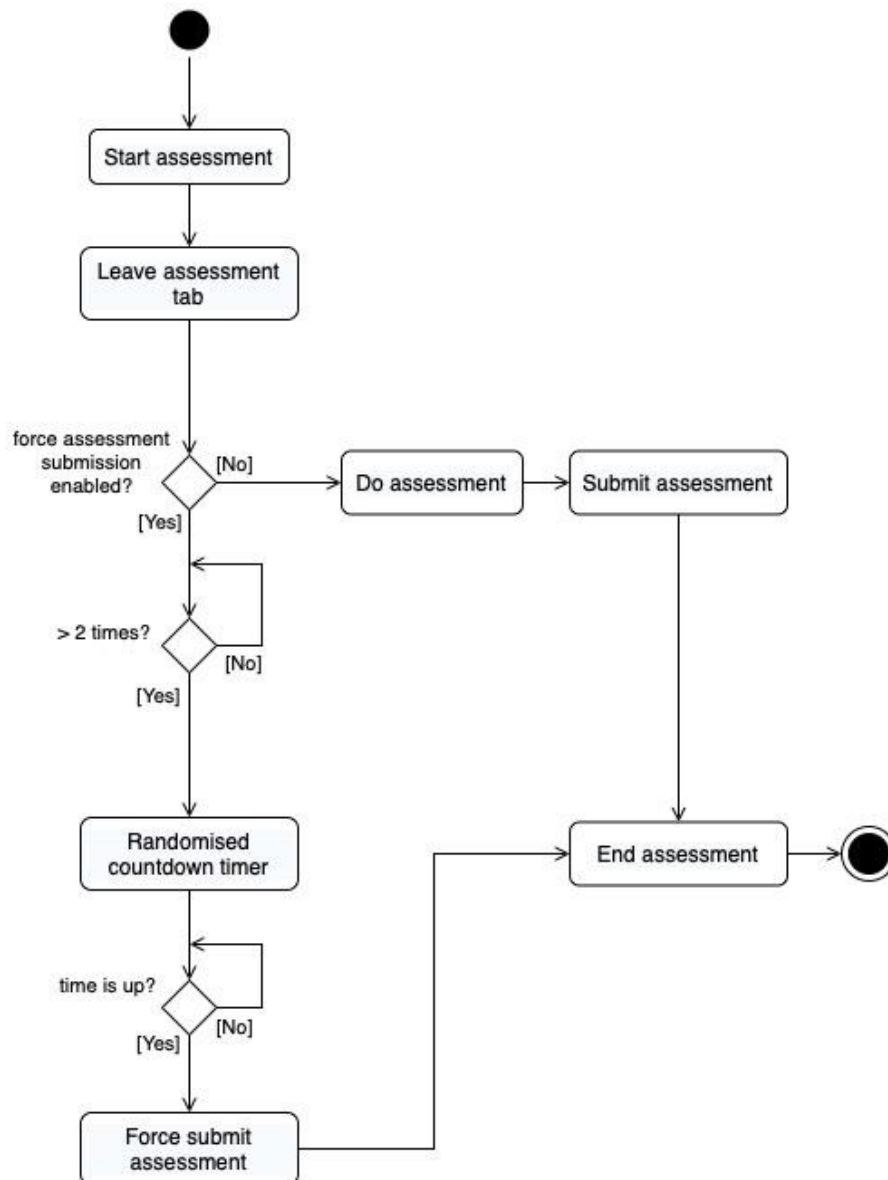


Figure 15 Activity Diagram for Force Submission of Assessment after Multiple Tab Switching

4.4.4 Email Notifications

Upon student course enrolment, if the student account does not exist, the account is created. Initial passwords are randomised and sent to each student via email. If the course has any existing published assessment, an email notification will be sent to the newly enrolled student (Figure 16).

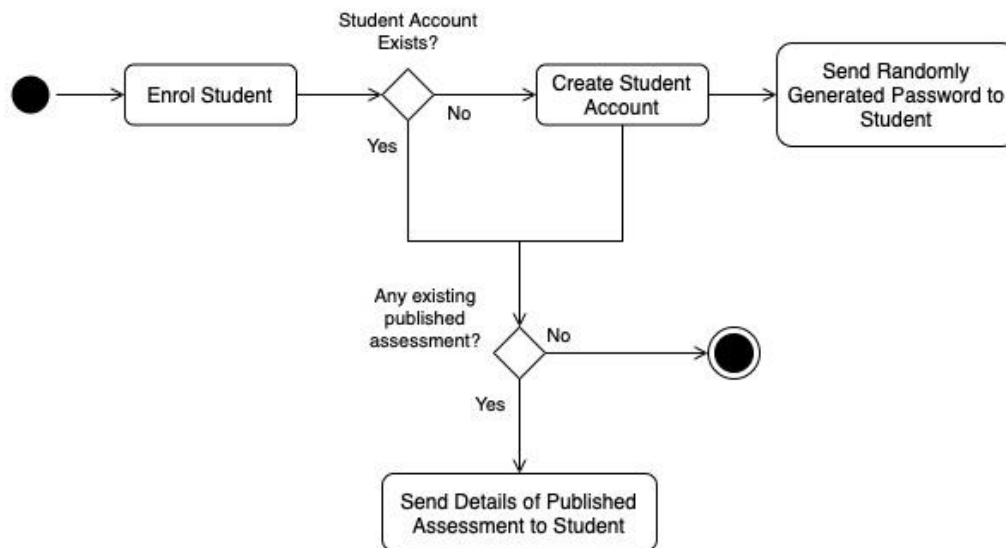


Figure 16 Activity Diagram of Student Account Creation

After an educator or lab assistant resets a student's password, the new password will be sent to the student (Figure 17).

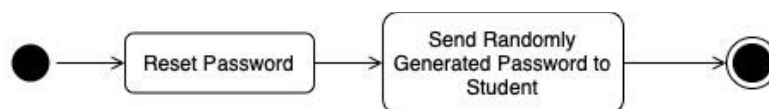


Figure 17 Activity Diagram of Student Reset Password

When an educator publishes an assessment, students enrolled in the course will be notified via email (Figure 18).

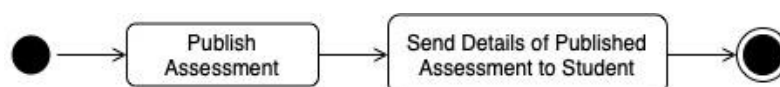


Figure 18 Activity Diagram of Published Assessment

4.5 Database (DB) Design

Since Django framework is shipped with its own Object Relational Mapper (ORM), the model classes translate into DB schemas. Hence, class diagram maps to Entity Relationship Diagram (ERD). The figure below shows the ERD of DB schemas that are relevant to the new AASP proctoring schemas (Figure 19). *CandidateSnapshot* is a newly added table and new fields have been added to *Assessment* table.

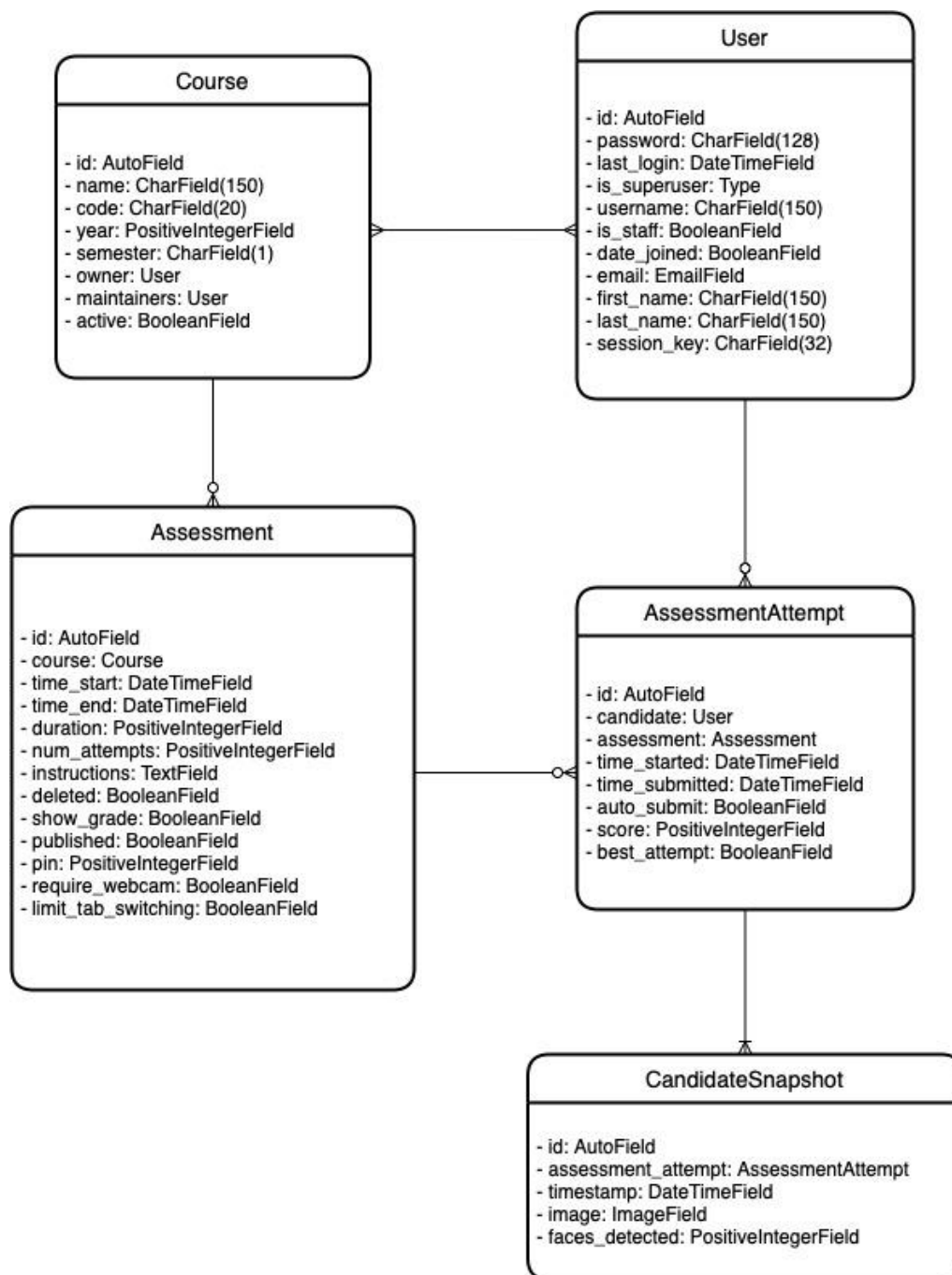


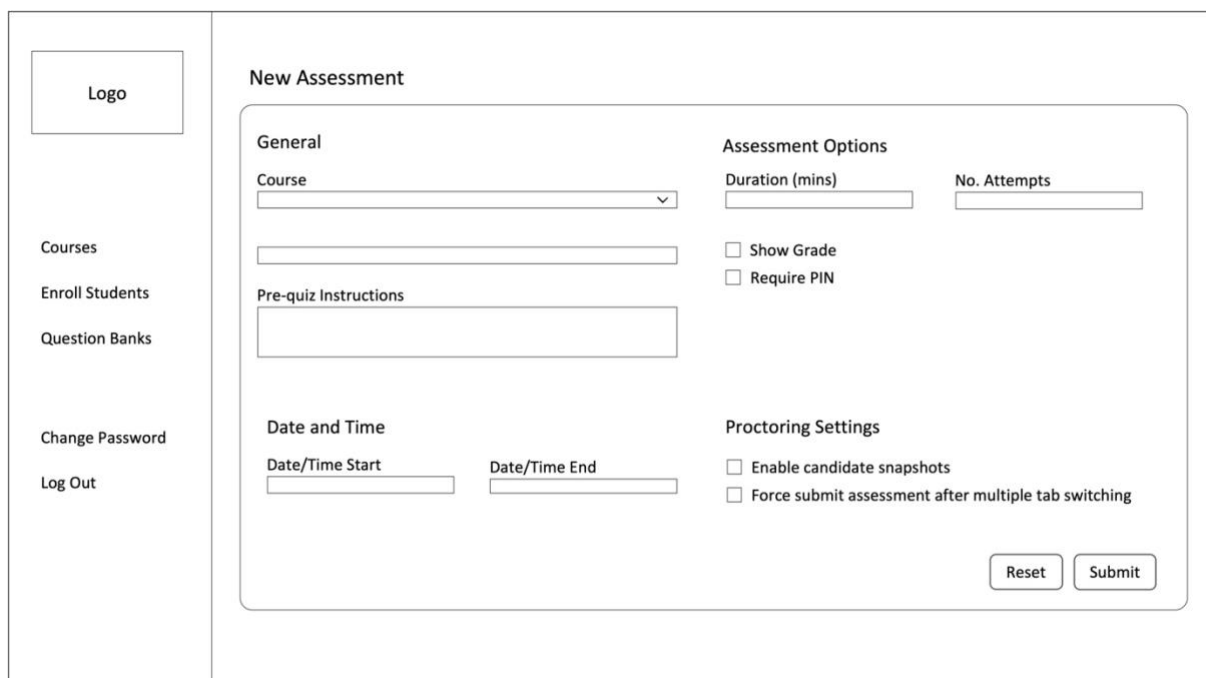
Figure 19 ERD of Relevant DB Schemas

4.6 User Interface (UI) Design

The initial UI design of the AASP proctoring features is displayed below. The wireframes show key AASP pages that contain the new proctoring features.

4.6.1 Assessment Creation and Update

When creating a new assessment, the educator can enable or disable the 2 proctoring features. This can be found at the bottom of the creation form, labelled “Proctoring Settings” (Figure 20). The proctoring settings will be saved and applied to all students’ assessment attempt.



The wireframe shows a user interface for creating a new assessment. It features a sidebar on the left with navigation links: Logo, Courses, Enroll Students, Question Banks, Change Password, and Log Out. The main content area is titled "New Assessment" and contains several sections: "General" with a Course dropdown and a text input; "Pre-quiz Instructions" with a text area; "Date and Time" with Date/Time Start and Date/Time End inputs; "Assessment Options" with Duration (mins) and No. Attempts inputs, and checkboxes for Show Grade and Require PIN; and "Proctoring Settings" with checkboxes for Enable candidate snapshots and Force submit assessment after multiple tab switching. At the bottom right are Reset and Submit buttons.

New Assessment	
General	Assessment Options
Course <input type="text"/>	Duration (mins) <input type="text"/> No. Attempts <input type="text"/>
<input type="text"/>	<input type="checkbox"/> Show Grade
Pre-quiz Instructions	<input type="checkbox"/> Require PIN
<input type="text"/>	
Date and Time	Proctoring Settings
Date/Time Start <input type="text"/>	<input type="checkbox"/> Enable candidate snapshots
Date/Time End <input type="text"/>	<input type="checkbox"/> Force submit assessment after multiple tab switching
	<input type="button" value="Reset"/> <input type="button" value="Submit"/>

Figure 20 Wireframe of Educator UI to Create a New Assessment

4.6.2 Proctoring Features During Assessment

If capturing of candidate snapshots is enabled, the student will be prompted to take an initial snapshot before the start of the assessment (Figure 21). The snapshot must contain the student's face and his matriculation card.

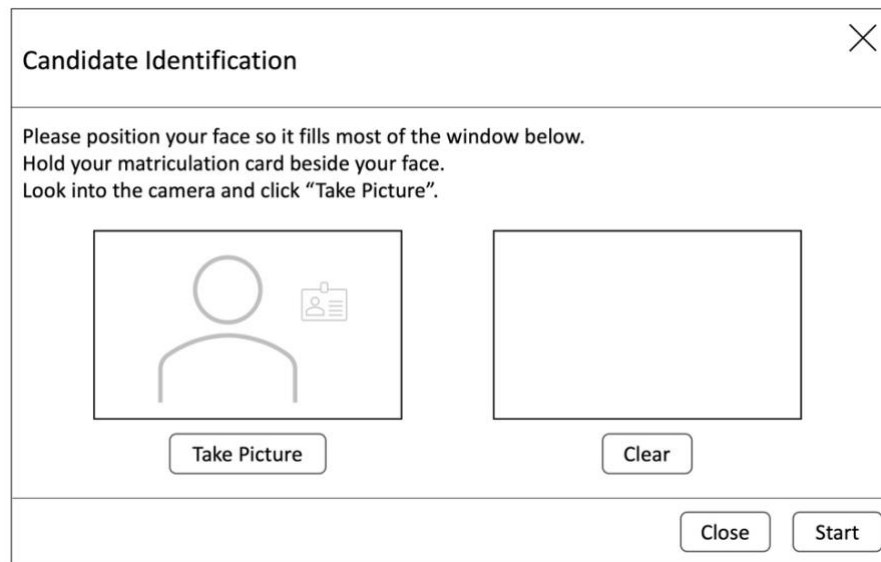


Figure 21 Wireframe of Student UI to Capture Initial Snapshot Before an Assessment

If force assessment submission is enabled, the student will be warned via a pop-up modal if he leaves the assessment screen (Figure 22).

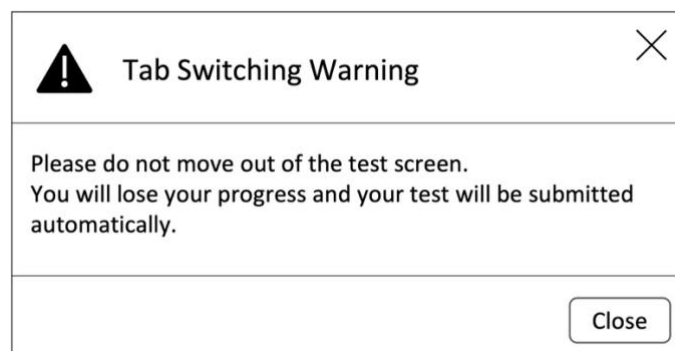


Figure 22 Wireframe of Student UI of Tab Switching Warning During an Assessment Attempt

4.6.3 Viewing Candidate Snapshots on Assessment Attempt Report

After students' submission, the educator can view their assessment attempts via the assessment report. If capturing candidate snapshots is enabled, presence of multiple faces or missing candidate will be flagged out in the table showing students' best attempts (Figure 23).

Logo

Courses

Enroll Students

Question Banks

Change Password

Log Out

Assessment Report

Completed Attempts

#	Candidate	Score	Duration	Total Attempts	Multiple Faces Detected	Missing Candidate	Actions
1	Student1	5	33 min	2	Yes	No	View
2	Student2	0	20 min	2	No	No	View
3	Student3	4	40 min	1	No	No	View

Ongoing & Processing Attempts

#	Candidate	Score	Duration	Total Attempts	Multiple Faces Detected	Missing Candidate	Actions
No data available in table							

Figure 23 Wireframe of Educator UI Viewing Students' Assessment Best Attempt

The educator can view all attempts of a student (Figure 24).

Assessment Attempts by Student1							
#	Time Started	Time Submitted	Score	Best	Multiple Faces Detected	Missing Candidate	Actions
1	16/01/2023 12:05 PM	16/01/2023 12:35 PM	5	Best Score	Yes	No	View
2	16/01/2023 12:40 PM	16/01/2023 12:44 PM	1		No	No	View
Close							

Figure 24 Wireframe of Educator UI Viewing a Student's Assessment Attempts

Students' snapshots will be available on the assessment attempt report after they submit their assessment attempt (Figure 25). The snapshots will be categorised into 3 collapsible sections – Multiple Faces Detected, Missing Candidate, and All Snapshots. Snapshots will have the timestamp as caption.

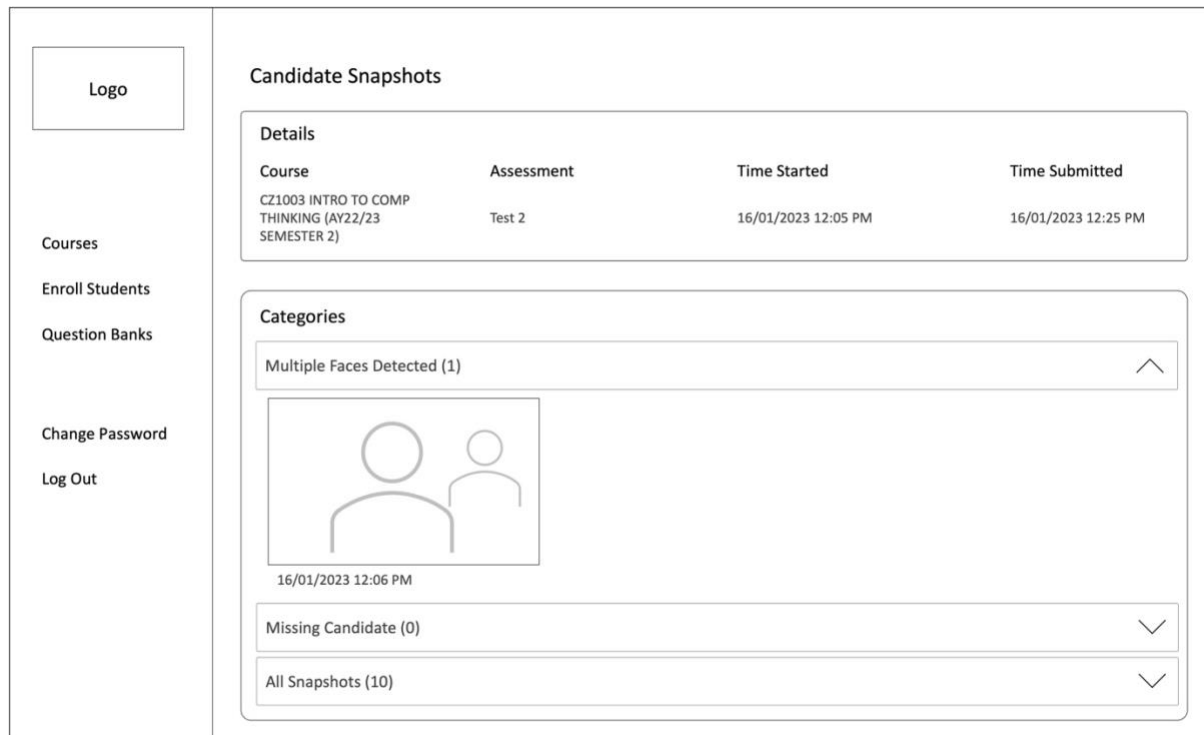


Figure 25 Wireframe of Educator UI to View Candidate Snapshots in Assessment Attempt Report

5 Implementation

5.1 Overview

Figure 26¹ shows parts of the system architecture involving the proctoring and email features.

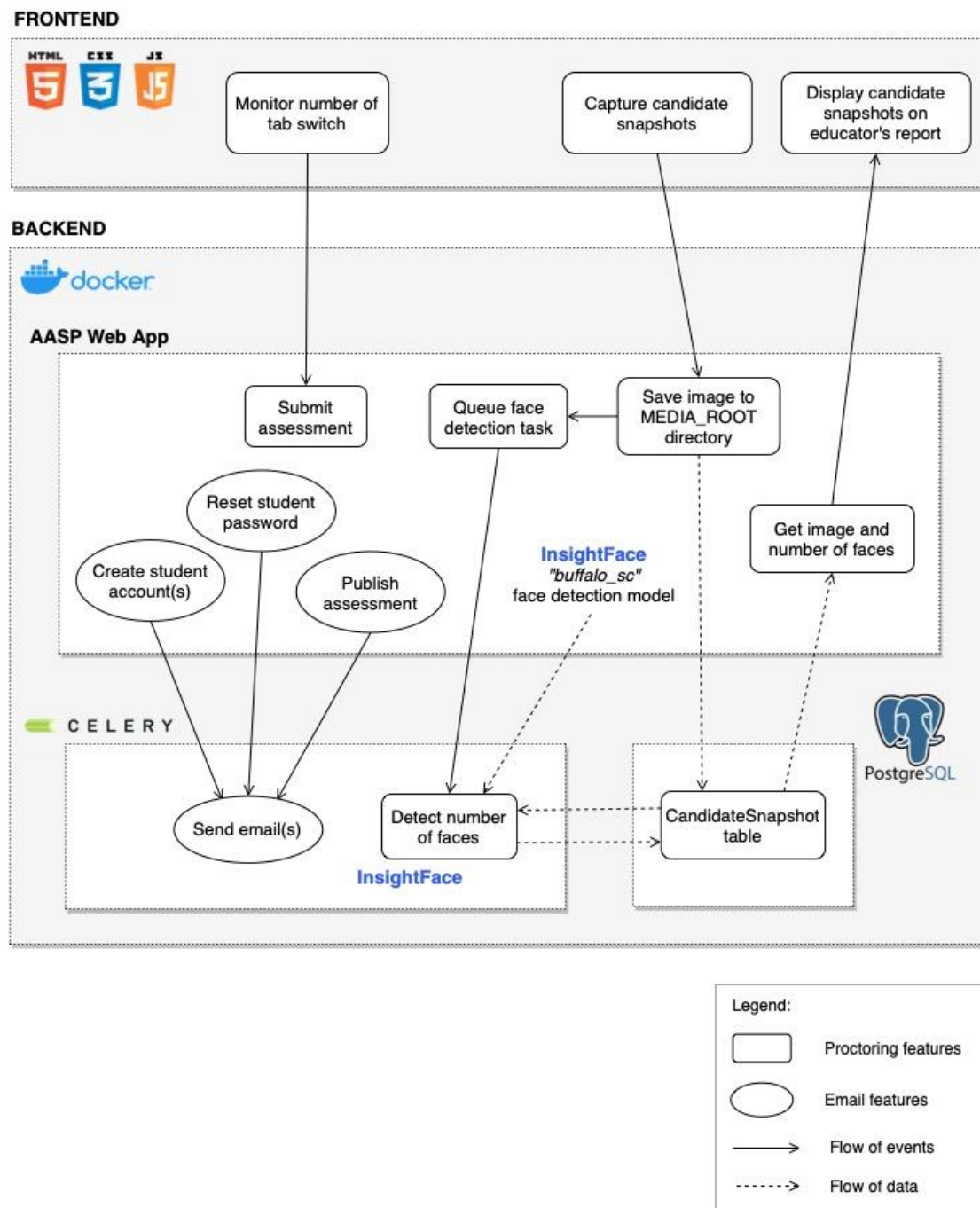


Figure 26 System Architecture

¹ Refer to Appendix for the referenced diagram.

On AASP web container creation, a small face detection model (~16.1 MB) by InsightFace [18] is copied from the repository to the host machine `/root/.insightface/models/buffalo_sc` via instructions stated in the Dockerfile. This allows the face detection to be done offline, without having to download any detection model directly from online sources.

To enforce a strict and monitored assessment environment, candidates' surrounding and their behaviour are monitored by capturing their snapshots are captured at randomised intervals throughout the assessment. The intervals are randomised between 1 minute and 10% of the total duration of the assessment. Each candidate snapshot is captured on the frontend browser using the computer's webcam. The image is sent to the backend via a HTTP POST request and saved to the AASP web app `MEDIA_ROOT` directory. The image file path is then saved to the DB *CandidateSnapshot* table. Thereafter, face detection is done on the image and the number of faces detected is saved to the same *CandidateSnapshot* table.

To efficiently disseminate information from AASP, emails are broadcasted to students when their accounts are created, when their passwords are reset, and when an educator publishes an assessment for courses that they are enrolled in. URLs to AASP are provided in the emails to give students easy access to the platform to either change their password or view details of the published assessment. The email host account and AASP base URL, which is dependent on the host machine IP address, are easily configurable in the `.env` file.

Both the face detection and sending of email functions are asynchronous tasks that are queued using the celery worker. They are non-blocking tasks that run in the background, allowing main operations to continue processing. Performing these celery tasks do not cause significant delays in HTTP requests and responses. Having celery tasks is especially useful when there are many students taking tests simultaneously, generating large amounts of images to be processed. It is also useful for bulk creation of student accounts since many emails are crafted and sent at once.

5.2 Proctoring Features

As the main objective of this project, proctoring features are added to enforce a strict and monitored assessment environment.

5.2.1 Capturing Candidate Snapshots

Before the start of an assessment, the student will be prompted to grant AASP access to the browser's camera (Figure 27).

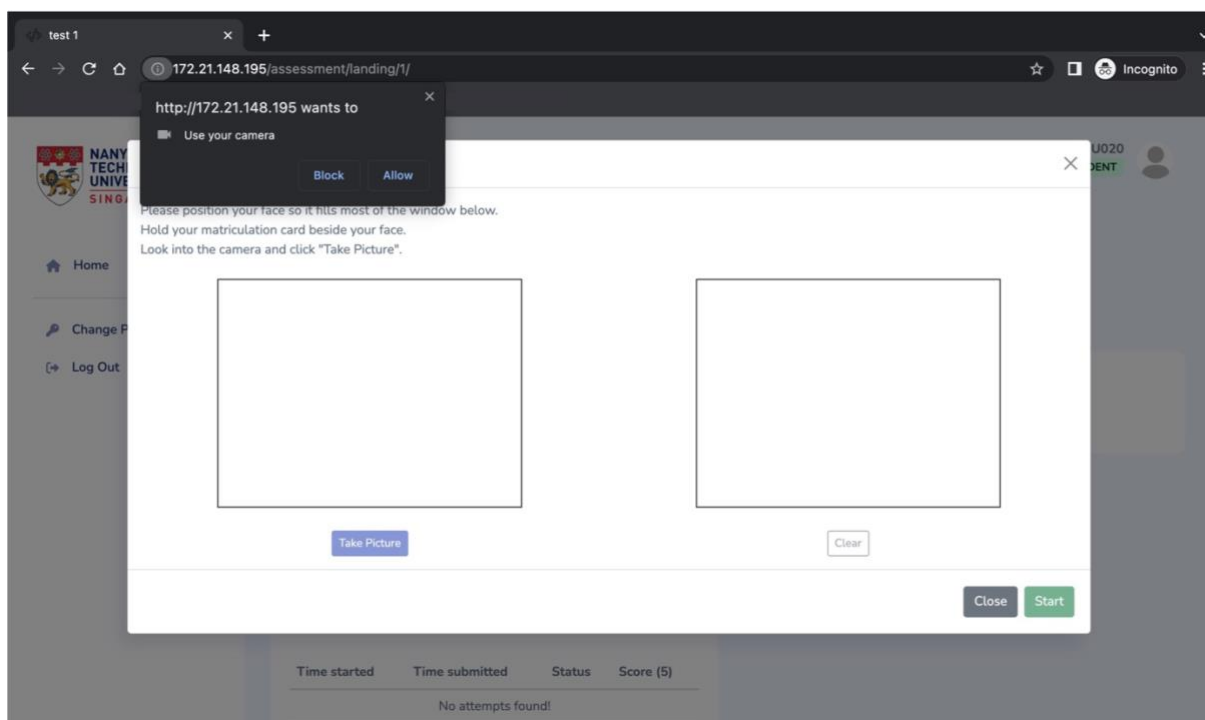


Figure 27 System Prompting Student for Browser Webcam Access

After allowing access to the webcam, the student will be prompted to take an initial snapshot of himself, together with his matriculation card (Figure 28).

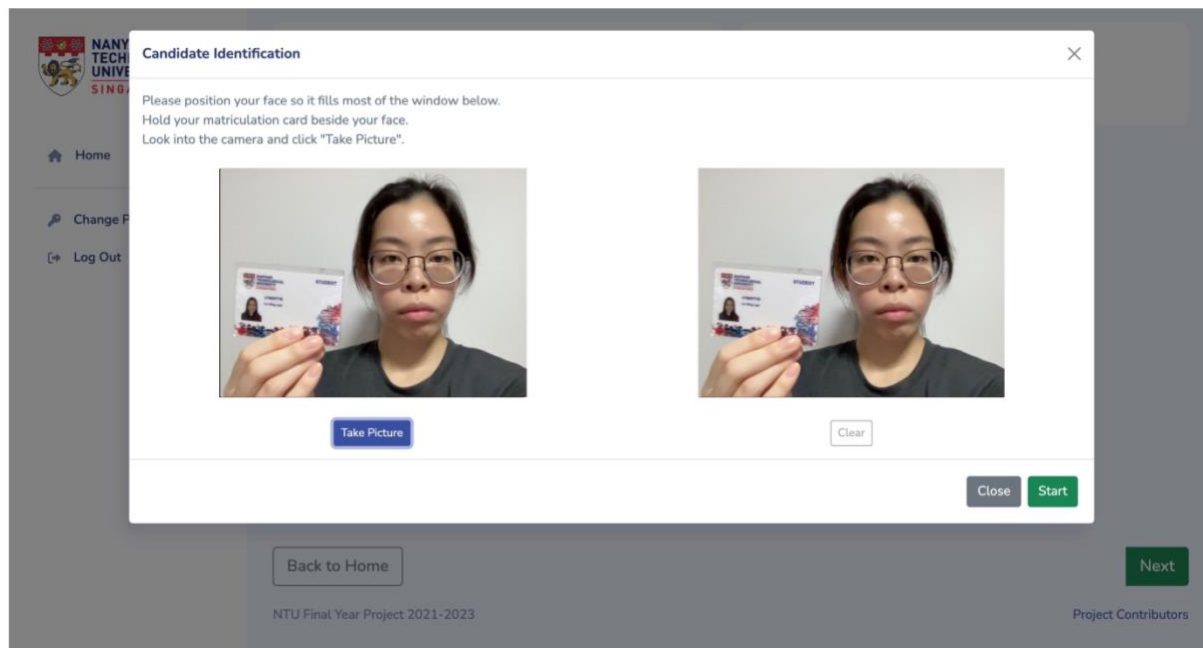


Figure 28 Student Taking Initial Snapshot with Matriculation Card

During the assessment, snapshots of the student will be captured periodically and processed by the AASP system. Number of faces detected will be recorded.

The educator can view all students' best attempts and if any snapshot contains multiple faces or does not contain a face, the system will flag it out by displaying a “Yes” or “No” in the report table (Figure 29).

The screenshot shows the 'Assessment Report' for 'test 1' in the 'Active' state. The left sidebar contains navigation links: Home, Courses, Enrol Students, Question Banks, Change Password, and Log Out. The main content area displays 'Completed Attempts' with a sub-header 'The best completed attempts of each candidate.' and a 'CSV' download button. A table lists attempts with columns: #, Candidate, Score (5), Duration, Total Attempts, Multiple Faces Detected, Missing Candidate, and Actions. One entry is shown for candidate WLIU020 with a score of 0, duration of 11 sec, and 2 total attempts. The 'Multiple Faces Detected' column shows 'Yes'. Below the table, it says 'Showing 1 to 1 of 1 entries' with 'Previous' and 'Next' navigation buttons. Below this, there is a section for 'Ongoing & Processing Attempts' with a similar search and display controls.

#	Candidate	Score (5)	Duration	Total Attempts	Multiple Faces Detected	Missing Candidate	Actions
1	WLIU020	0	11 sec	2	Yes	No	View

Figure 29 Educator Viewing Assessment Attempts of Students

Upon viewing a particular student's assessment records, all his attempts will be displayed in a pop-up modal (Figure 30).

The screenshot shows the same 'Assessment Report' interface, but with a pop-up modal titled 'Assessment attempts by WLIU020' open. The modal contains a table with columns: #, Time Started, Time Submitted, Score (5), Best, Multiple Faces Detected, Missing Candidate, and Actions. Two attempts are listed for candidate WLIU020. The first attempt has a score of 0 and is marked as the 'Best Score'. The second attempt also has a score of 0. Both attempts show 'Yes' for 'Multiple Faces Detected' and 'No' for 'Missing Candidate'. The modal has a 'Close' button at the bottom right.

#	Time Started	Time Submitted	Score (5)	Best	Multiple Faces Detected	Missing Candidate	Actions
1	16/01/2023 6:16 PM	16/01/2023 6:17 PM	0	Best Score	Yes	No	View
2	16/01/2023 6:21 PM	16/01/2023 6:27 PM	0		No	No	View

Figure 30 Educator Viewing a Student's Attempts

The educator can view a student's snapshots that have been categorised into “Multiple Faces Detected”, “Missing Candidate” and “All Snapshots” (Figure 31). These categories can be expanded or collapsed to view the relevant images.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

YRLOKE EDUCATOR

Candidate Snapshots
WLIU020

Course	Assessment	Time Started	Time Submitted
CZ1003 INTRO TO COMP THINKING (AY22/23 SEMESTER 2)	test 1	16/01/2023 06:21 PM	16/01/2023 06:27 PM

Categories

- Multiple Faces Detected (1)**
 - 16/01/2023 06:24 PM
- Missing Candidate (0)
- All Snapshots (7)

Figure 31 Educator Viewing a Student's Attempt Snapshots

If more than one face is detected in a snapshot, it will be categorised into the “Multiple Faces Detected” category. If face is absent from a snapshot, it will be categorised into the “Missing Candidate” category. The initial snapshot of the candidate with his matriculation card is found in the “All Snapshots” category, with the photo caption “Initial snapshot: <timestamp>”.

5.2.2 Force Submitting Assessment after Multiple Tab Switching

During the assessment, if the student leaves the assessment screen, he will be prompted a warning (Figure 32). There are several actions that are considered as leaving the assessment screen – switching tab, switching window, clicking on a system-generated pop-up, etc.

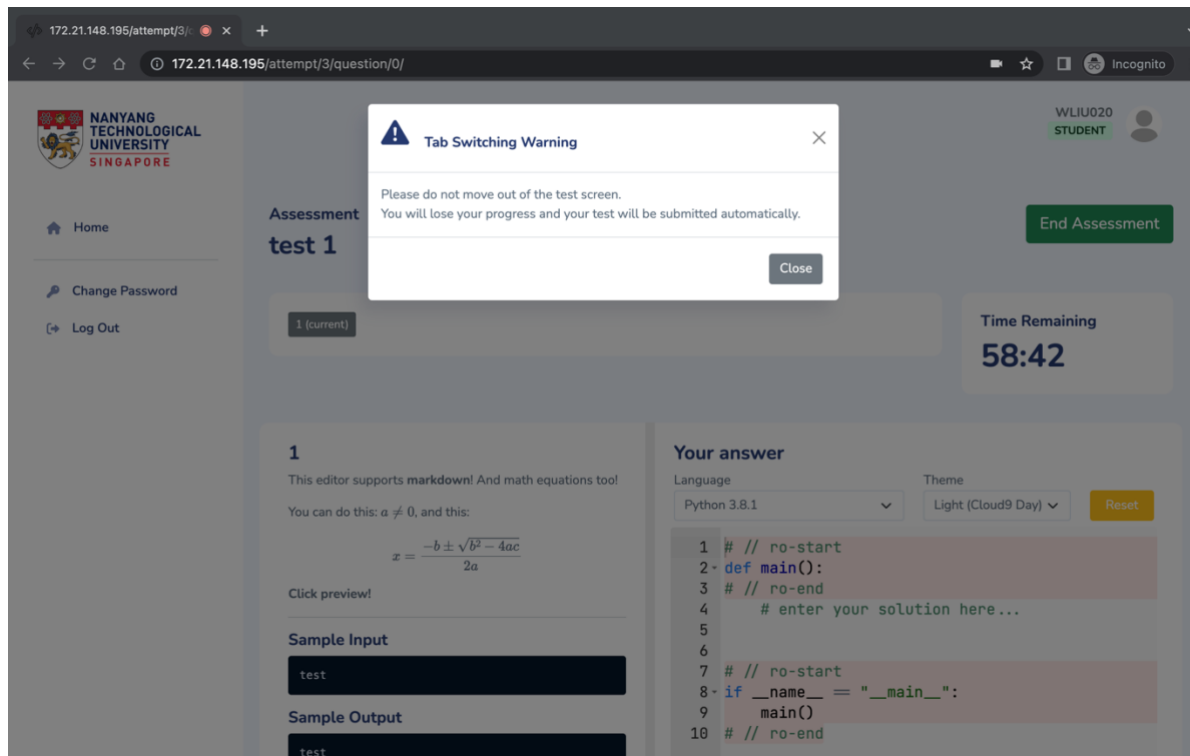


Figure 32 Tab Switching Warning When Student Leaves Assessment Screen

After the student's second time leaving the screen, a countdown timer will be started. The timer randomises the countdown time between 3 and 10 seconds. The student must close the tab switching warning modal to stop the timer (Figure 33). Else when the time is up, his assessment will be forcefully submitted automatically.

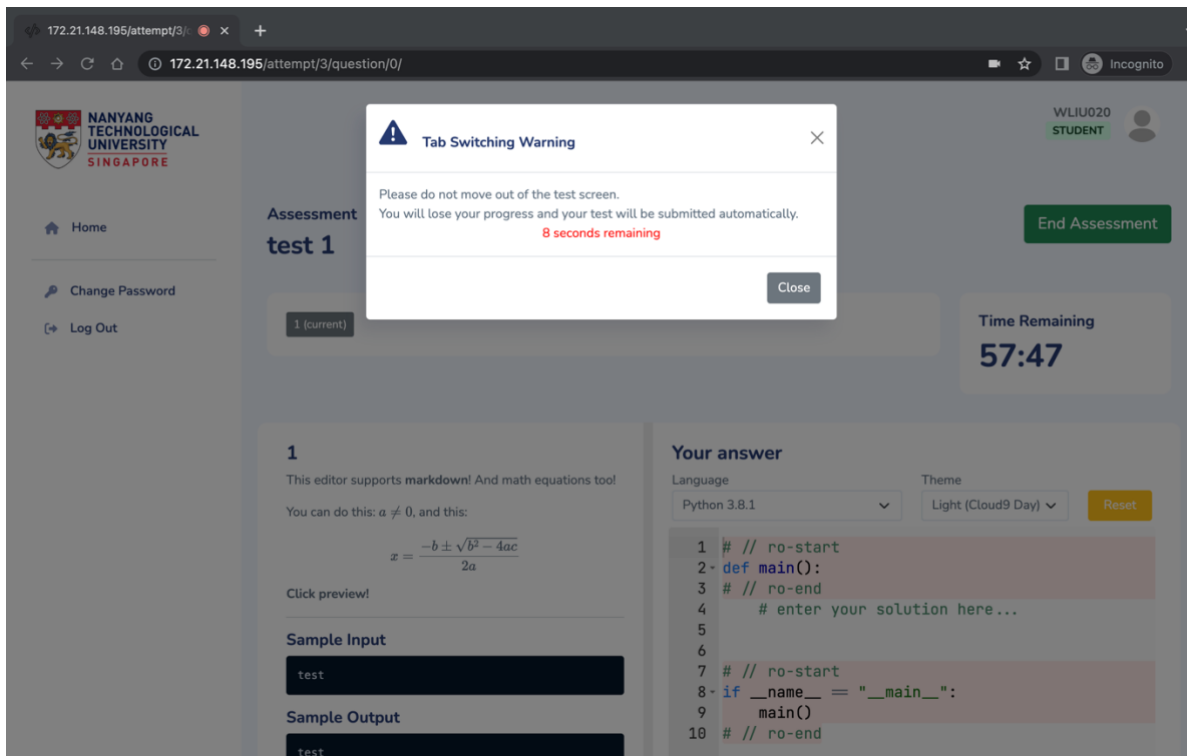


Figure 33 Tab Switching Timer after Student Leaves Screen Twice

The countdown will start each time he leaves the assessment tab from the third time onwards. This timer is there to give students a chance to react when he leaves the tab by accident, for example when the computer prompts a system update alert.

5.3 Email Notifications

The use of email is integrated for 2 purposes.

1. Disseminate securely and randomly generated alphanumeric passwords to students upon account creation and/or password reset.
2. Notify students of a published assessment for courses that they are enrolled in.

5.3.1 Student Passwords

Currently, AASP sets all initial student passwords as the default “password123”. This potentially causes a problem if students do not change their passwords promptly after account creation. Students can log in to other students’ accounts without anyone knowing. This may create an opportunity of students sabotaging one another’s.

In this new feature, all student passwords are generated using Django’s crypto module which uses the secrets module. This ensures all passwords are securely and randomly (not pseudo-randomly) generated, achieving higher security of the students’ accounts.

Passwords will be known to students once their accounts are created without having the educator or lab assistant communicate with the students on their own. Figure 34 shows an email that was sent to the student upon his account creation by an educator.

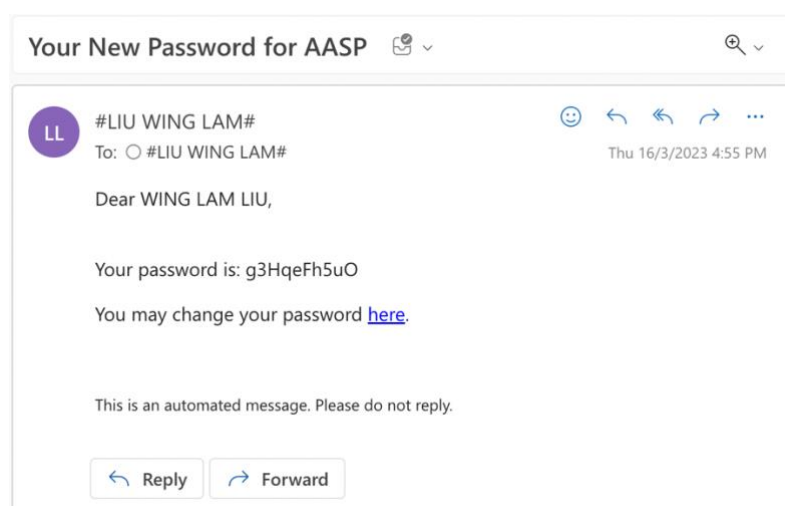


Figure 34 Email Sent to Student upon Account Creation

Similarly, the securely and randomly generated passwords will be made known to students via email once the password reset has been done by an educator or lab assistant. Figure 35 shows an email that was sent to the student a password reset was done by an educator.

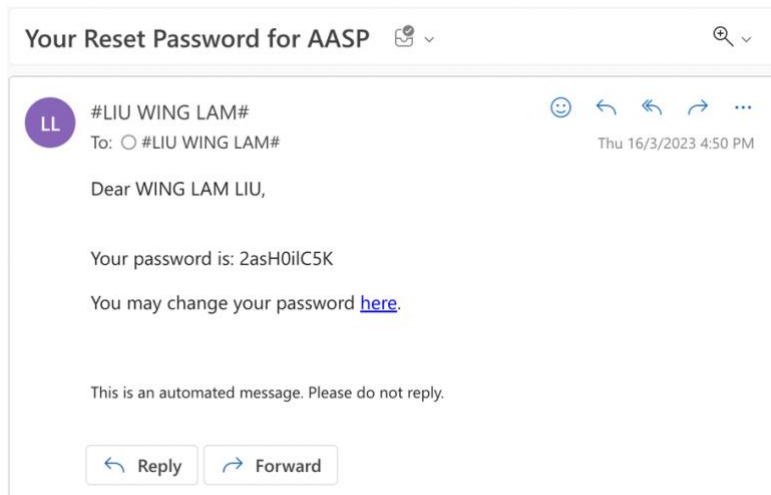


Figure 35 Email Sent to Student Upon Password Reset by Educator or Lab Assistant

Upon clicking the hyperlink in the email, the student will be prompted to change their password as shown in Figure 36.

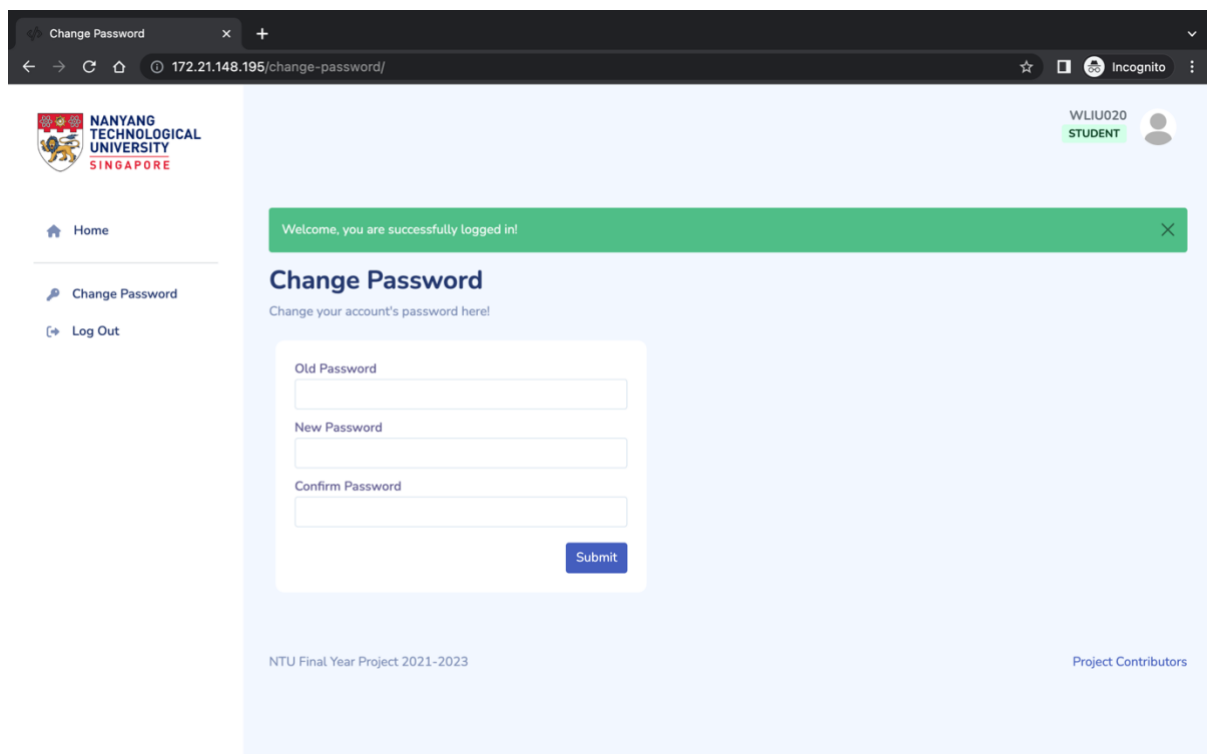


Figure 36 Student Change Password Interface after Clicking URL in the Email

5.3.2 Published Assessments

Immediately after the educator publishes an assessment, he will be informed that students have been notified of the assessment via email (Figure 37).

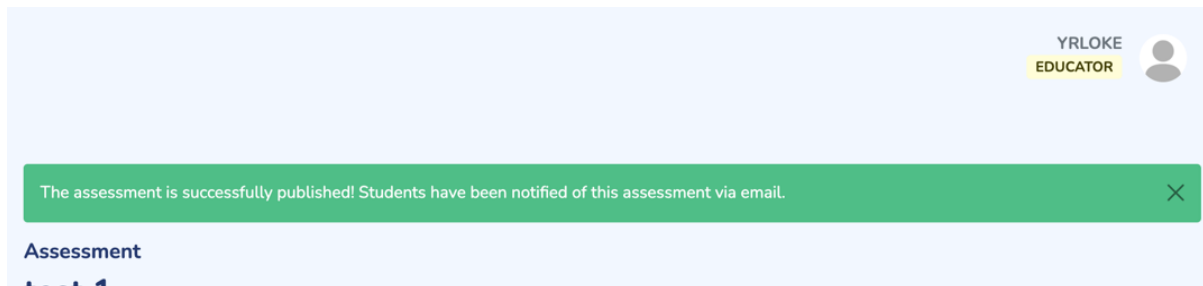


Figure 37 Educator Published an Assessment

Figure 38 shows an email sent to the student that was enrolled in the published assessment's course. Clicking on the hyperlink will bring students to the assessment information page where more information and instructions for candidates are available.

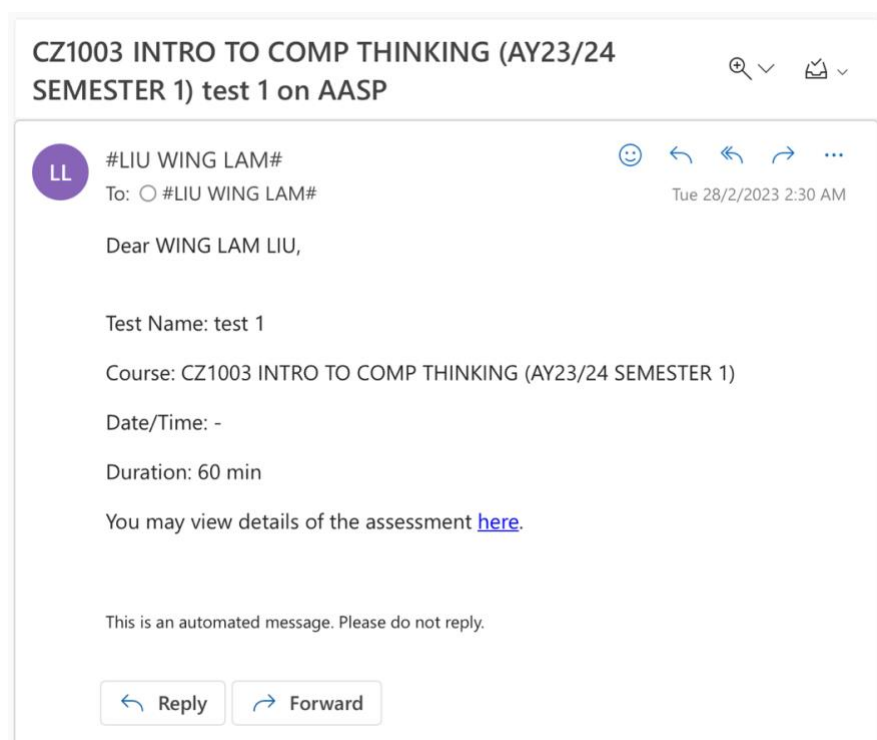


Figure 38 Email Notification Sent to Student upon Publishing Assessment

6 Code Refactors & Standardisations

Other than implementing useful features, some code refactors and standardisations were done.

6.1 HTTP Response Status Codes

HTTP response codes and messages were better defined to provide a more meaningful description of each response. In the long term, this would be useful for application monitoring and maintainability, as well as future debugging should users face any errors.

Currently, AASP backend always returns a successful HTTP 200 OK status code regardless of whether errors were encountered. Error HTTP status codes are only returned when the Judge0 APIs give an error response. To make the responses more informative, HTTP status codes are categorised according to conventions specified in the RFC 9110 [19]. The 2xx Successful class of status codes is used when the frontend's request has been successfully received and processed. The 4xx Client Error status codes are used when the frontend's request has been received, but the backend cannot process it. Lastly, 5xx Server Error status codes are used when the Judge0 server gives an error response, causing AASP backend not being able to process the frontend's request.

To send the appropriate status codes, Django's REST framework was used [20].

Having this standardised set of HTTP status codes lets developers know whether a request has been successful. The better-defined status codes also allow the client (frontend) to detect the success or failure of ajax calls and handle appropriately.

6.2 Prints & Logs

Unnecessary prints and console logs were removed as it may also expose data that should not be known to users. It is also unprofessional to keep them around in a working application in production.

7 Challenges Faced

7.1 Debugging Existing Bugs

While familiarising myself with AASP, I was unsure in determining whether certain behaviours and output encountered are bugs due to 2 reasons:

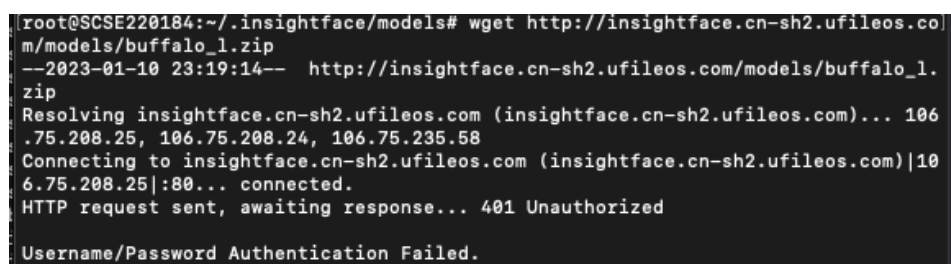
1. Unfamiliarity with features and functionalities.
2. Uniform 2xx HTTP status codes from AASP backend regardless of whether errors were encountered when requests were processed.

As such, each time I encountered behaviours that seemed abnormal, I traced the executed code in debug mode to check the values of variables in the backend. After determining whether the code is working as expected, I standardised the set of HTTP status codes used in AASP to facilitate future debugging as discussed in Chapter 6.1 HTTP Response Status Codes.

7.3 Bringing InsightFace Detection Model Offline

One of the main features in this project is capturing candidate snapshots and performing face detection on the images using InsightFace. However, unexpected issues came up a few weeks after implementing this feature.

Firstly, AASP was unable to download the face detection model on AASP start-up as it showed the error below (Figure 39).



```
root@SCSE220184:~/insightface/models# wget http://insightface.cn-sh2.ufileos.com/models/buffalo_1.zip
--2023-01-10 23:19:14-- http://insightface.cn-sh2.ufileos.com/models/buffalo_1.zip
Resolving insightface.cn-sh2.ufileos.com (insightface.cn-sh2.ufileos.com)... 106.75.208.25, 106.75.208.24, 106.75.235.58
Connecting to insightface.cn-sh2.ufileos.com (insightface.cn-sh2.ufileos.com)|106.75.208.25|:80... connected.
HTTP request sent, awaiting response... 401 Unauthorized
Username/Password Authentication Failed.
```

Figure 39 Error Downloading Face Detection Model in AASP Celery Worker Container

Upon contacting the developers of InsightFace, I found that they updated their model downloading provider after I installed and started using their package. After updating to the latest version, the model downloaded successfully. However, I decided it would be better to

bring the model offline so that AASP does not have to download it from the online source on first start-up. Instructions were stated in the Dockerfile to copy relevant files to the host machine's root address.

Just as I thought the face detection should work again, the model contained files that were too large (each file >100 MB) to push to GitHub repository. I attempted to use Git Large File Storage (LFS), but AASP was unable to load the model, SCRFD_10G, due to the altered filetype and file content. This error `[ONNXRuntimeError] : 7 : INVALID_PROTOBUF : Protobuf parsing failed` was encountered. To resolve the error, the host machine must have Git LFS installed but not all servers will have this package installed. This prompted me to use a smaller pretrained model, SCRFD_500M, that can be pushed to GitHub without changing the filetype. Despite having lower accuracy (Figure 40) [21], tests were done using images taken under different lightings with different angles. The tests showed that the model is sufficient to detect a presence or lack of a human face in front of the webcam during an assessment attempt.

Pretrained-Models						
Name	Easy	Medium	Hard	FLOPs	Params(M)	Infer(ms)
SCRFD_500M	90.57	88.12	68.51	500M	0.57	3.6
SCRFD_1G	92.38	90.57	74.80	1G	0.64	4.1
SCRFD_2.5G	93.78	92.16	77.87	2.5G	0.67	4.2
SCRFD_10G	95.16	93.87	83.05	10G	3.86	4.9

Figure 40 Accuracy of InsightFace Pretrained Face Detection Models

8 Future Works

8.1 More Question Types

AASP currently only supports programming/coding question type. To make AASP more applicable to other courses in the University, more question types can be added. These include Multiple-Choice Questions (MCQ), short/long answer and questions that require handwritten solutions such as mathematical equations.

8.2 AutoSave Code

Autosaving students' code while they are attempting questions will allow them to revisit them.

8.3 Configure Proper Email Host

The email set up in AASP is configurable. To make it production ready, an SCSE do-not-reply alias email should be set up.

8.4 Get Domain to Host AASP Application

SSL digital certificate should be used to authenticate AASP. Thereafter, nginx can be configured to use SSL to encrypt HTTP requests so that browsers will treat AASP as a secure site and not block vulnerable features.

8.5 Facial Recognition

On top of face detection, having facial recognition would better enforce a strict assessment environment. However, this would be challenging since facial recognition is dependent on the computer webcam's definition, and clarity and accuracy of the photo on the matriculation card.

9 Conclusion

This project started with the study of the topic on Online AA and the research on existing products. The 3 types of AA methodologies – dynamic, static, and hybrid analyses, were studied but it was concluded that dynamic analysis was the most useful for AASP. Next, gaps were found as the lack of test proctoring meant AASP was not a good alternative or substitute for in-person assessments. As such, proctoring was established as the project focus. Using insights from studying existing products, monitoring features were adapted and designed for AASP.

The main objective of this project was to enforce a monitored and strict assessment environment so that AASP can be used for the University assessments. This was achieved through implementing proctoring features to monitor student behaviour and environment throughout the assessment, to deter and prevent cheating.

The UI was carefully designed based on a few considerations – the educator’s ease of disabling and enabling proctoring features for each assessment, and the visibility of these features to students. The snapshot capturing and force submission options must be clear and concise to educators. On the other hand, there must be a certain level of abstraction to students. They must know that they are monitored through the webcam but must not know how and when the snapshots are timed and processed. They must also know that leaving the assessment tab is not allowed but must not know exactly when the assessment will be forcefully submitted.

To minimise any lag in response time, any image processing such as face detection is done asynchronously in a Celery queue system. This would allow AASP’s normal flow to continue executing while the image is processing in the background.

Since the iterative Agile approach was adopted, more gaps were found subsequently, and email features were implemented as further enhancements. Students’ initial and reset passwords are now securely and randomly generated, and only made known to the specific individual via email. Students enrolled in courses will also be notified of published assessments of the courses. Like the image processing, all emails are sent via Celery.

Through this project, proctoring and email features have been integrated into AASP, making it a more well-rounded online assessment platform. With the proctoring features especially, the University can use AASP easily to conduct monitored assessments. However, future works such as those highlighted in Chapter 8 Future Works can be considered to make AASP more robust.

In conclusion, having only one developer to manage the frontend, backend, and deployment of AASP has proven to be challenging. When problems arose, overcoming them alone was a learning experience. Overall, it has been a fruitful and rewarding journey integrating new features into the existing AASP to make it more functional and ready to conduct online assessments.

Bibliography

- [1] D. Fonte, D. d. Cruz, A. L. Gançarski and P. R. Henriques, “A Flexible Dynamic System for Automatic Grading of Programming Exercises,” in *2nd Symposium on Languages, Applications and Technologies*, 2013.
- [2] K. Zen, D. A. Iskandar and O. Linang, “Using Latent Semantic Analysis for Automated Grading Programming Assignments,” in *2011 International Conference on Semantic Technology and Information Retrieval*, Putrajaya, Malaysia, 2011.
- [3] A. Gordillo, “Effect of an Instructor-Centered Tool for Automatic Assessment of Programming Assignments on Students’ Perceptions and Performance,” *Technology-Enhanced Learning: Applications, Architectures and Challenges*, vol. 11, no. 20, 2019.
- [4] A. Singh, S. Karayev, K. Gutowski and P. Abbeel, “Gradescope: A Fast, Flexible, and Fair System for Scalable Assessment of Handwritten Work,” in *L@S 2017: Fourth (2017) ACM Conference on Learning @ Scale*, Cambridge Massachusetts USA, 2017.
- [5] Y. Cao, P. Lee, S. N. Liao and R. Ord, “Paper or Online?: A Comparison of Exam Grading Techniques,” in *2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*, New York, NY, USA, 2019.
- [6] S. Dhawan, “Online Learning: A Panacea in the Time of COVID-19 Crisis,” *Journal of Educational Technology Systems*, vol. 49, no. 1, pp. 5-22, 2020.
- [7] Blackboard Inc., “Create Tests,” [Online]. Available: https://help.blackboard.com/Learn/Instructor/Ultra/Tests_Pools_Surveys/Create_Tests_and_Surveys. [Accessed 13 September 2022].
- [8] Y. Q. Soh, “Automated Assessment Platform (AASP),” Nanyang Technological University, Singapore, 2021.
- [9] G. S. Yap, “Development of Auto-Assessment System,” Nanyang Technological University, Singapore, 2021.
- [10] J. W. Lee, “Full-stack Web Development for Auto-Assessment Platform,” Nanyang Technological University, Singapore, 2022.

- [11 M. T. Helmick, "Interface-based programming assignments and automatic grading of java programs," in *ITiCSE '07: Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*, Dundee Scotland, 2007.
- [12 Z. Ullah, A. Lajis, M. Jamjoom, A. Altalhi, A. Al-Ghamdi and F. Saleem, "The effect of automatic assessment on novice programming: Strengths and limitations of existing systems," *Computer Applications in Engineering Education*, vol. 26, no. 6, pp. 2328-2341, November 2018.
- [13 Turnitin, LLC., "Can Gradescope help prevent cheating on remote assessments?," May 2022. [Online]. Available: <https://help.gradescope.com/article/y9ik9y70u1-remote-faq-prevent-cheating>. [Accessed 13 September 2022].
- [14 Respondus, Inc., "LockDown Browser," [Online]. Available: <https://web.respondus.com/he/lockdownbrowser/>. [Accessed 13 September 2022].
- [15 HackerRank, Inc., "Proctoring HackerRank Tests," September 2022. [Online]. Available: <https://support.hackerrank.com/hc/en-us/articles/360011479133-Proctoring-HackerRank-Tests>. [Accessed 13 September 2022].
- [16 HackerEarth, "Drive quality assessments using proctoring," [Online]. Available: <https://www.hackerearth.com/recruit/features/proctoring/>. [Accessed 13 January 2023].
- [17 ProProfs Quiz Maker, "How to Disable Tab Switching for Quiz Takers," ProProfs, 2005 - 2023. [Online]. Available: <https://quiz.proprofs.com/how-to-disable-tab-switching-for-test-takers>. [Accessed 1 March 2023].
- [18 InsightFace, "State of the art deep face analysis library," 2019. [Online]. Available: <https://insightface.ai>. [Accessed December 2022].
- [19 R. Fielding, M. Nottingham and J. Reschke, "'HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110," June 2022. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc9110>. [Accessed 7 March 2023].
- [20 Encode OSS Ltd., "Web APIs for Django," 2023. [Online]. Available: <https://github.com/encode/django-rest-framework>. [Accessed 7 March 2023].
- [21 J. Guo, J. Deng, A. Lattas and S. Zafeiriou, "Sample and Computation Redistribution for Efficient Face Detection," Cornell University, Ithaca, New York, 2021.

Appendix

System architecture diagram is taken from Lee [10].

