
Implementation Document

for

Ai Invigilation System

Version 1.0

Prepared by Zicheng Guo, Xing Li, Yongxin Zhao

Department of Computing and Software, McMaster University

Supervisor: Dr. Rong Zheng

Table of Contents

1. System Architecture	3
1.1 Overview	3
1.2 WI System	3
1.3 MM System	4
1.4 BE System	5
2. Implementation Details	5
2.1 WI System	5
2.2 MM System	5
2.3 BE System	6
3. Interface Descriptions	6
3.1 WI System & MM System	7
3.2 MM System and & BE System	7
4. Naming Conventions	7
4.1 WI System	7
4.2 MM System	11
4.3 BE System	14
5. GitLab Repo	16
5.1 URL	16
5.2 Commits	16

It contains several pages for different features, handling two types of users (supervisors and technical staff) authentication and distributing corresponding rights to each type of user.

The main page of the system allows users to identify themselves as a supervisor or staff member. Then the react app will redirect the user to one of the authentication pages. In terms of the user authentication pages, the two types of users share a similar interface but the paths are different. Once the user is authenticated, the app will redirect to the home page. For supervisors, the homepage contains the full features of the system (including setting triggers, creating exams, checking exam history, set up cameras and logout) while the staff member's home page only contains setup cameras and logout.

The structure of the homepage consists of a sidebar and a content section. The sidebar can easily navigate the users to each feature of the system.

To build the react app, users have to download Node.js and run the following commands in the ai-invigilation-system folder:

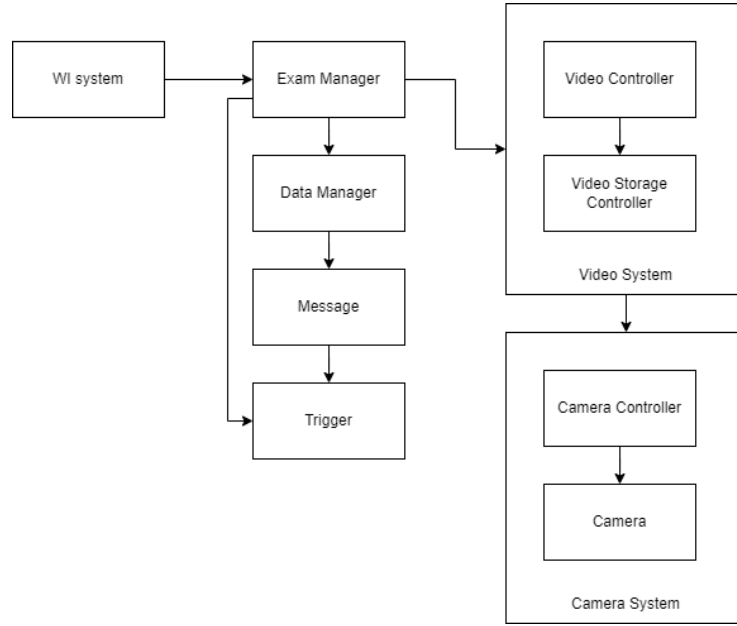
- `npm i`
- `npm start`

Then, the WI react app will be launched in the browser with the URL `localhost:3000`.

1.3 MM System

The main machine system handles the user input data sent from the web interface, creates corresponding objects and returns them to users. At the same time, it operates with back-end modules to analyze frames, generate messages and support the major features of the entire system.

The structure of classes and their relationships can be demonstrated by the following diagram:



From the hardware perspective, this system contains a GUI connecting the hardware (cameras) and the system, then receives camera feeds and sends them to the back end.

1.4 BE System

This system is the back-end server running the AI models analyzing captured camera videos. It connects with WI (for streaming camera feeds and output data) and MM (for storing frames and videos) as described in the overview structure diagram.

2. Implementation Details

2.1 WI System

The WI system is implemented as a react app, so the languages used are HTML5 for content, CSS for decoration and JavaScript for the interactions. In the react app, each feature is embedded in a webpage while links and routers are used to navigate among these pages.

Each page in the web interface is assigned a hierarchy path name (detailed naming conventions are explained in section 4 of the document) in order to track and identify pages. Buttons are implemented as React Link that supports the navigation among pages, it also ensures the corresponding admin rights are controlled for different types of users.

As for the information passing among pages inside the react app, it is handled by the react app. To be specific, the exam information page contains a form receiving user input with the submit parameter set to be sending the information to a new display info webpage.

Besides, an API (especially the Python flask module) is used to connect the front-end and back-end of the system. It transmits data between WI and MM/BE via GET and POST requests, and the requests are handled by the react app itself.

The user authentication feature is implemented by the form, and once the form is submitted, the input data will be sent to both the display info page (via data passing inside the react app) and the POST request to the MM/BE system.

2.2 MM System

The MM system in one sentence, handles the interaction between the back-end server and the WI system. We use python to implement our MM system. We have basically three big modules for our MM system, exam manager, video system and camera system, as the figure shown above.

The Exam_manager module is focusing on helper functions for two other modules, it creates a basic foundation. The video system and camera system module have their own functions and derive some of the functions from exam_manager.

For now, the communication between the MM system with the BE system and the WI system is left unimplemented, we are not yet at the stage to connect them all.

Some of the inputs of the functions are received from the WI system and some are from the BE system, MM system is like a transition for WI and BE, providing feedback for each other.

2.3 BE System

The BE system is implemented as a Python CLI app, where pytorch and dlib is used as a neural network engine. CUDA GPU is needed for the best performance, for details about the environment, please check the README file in the back_end directory.

Currently, for debugging and development purposes, the system is using the local camera to fetch webcam pictures instead of receiving frames from MM. In addition, the above `local` mode will be

left in the final version and can be selected with a command line argument for technicians to adjust the system.

At the current stage for BE, networking is left unimplemented(i.e. Communicating with MM and WI). We focus on the detection part of the BE system.

The BE will run using CLI arguments and once it is running it will start fetching local webcam images using CV2 package, then the image is passed to the detection algorithm `AlgorithmControler` and show the result in a window. In the window of the result, landmark and detection feature will be labelled on the original image, currently, 2 major features have been added: mouse action estimation and head yaw estimation. If it has passed the preset threshold, the line will be marked red instead of green.

In local mode, for each 10 frames, 1 will be saved to the preset directory(set by CLI argument) for backup purposes, and for online mode in the future, every frame it received will be saved.

3. Interface Descriptions

3.1 WI System & MM System

As mentioned above, the interfaces used between the WI system and the MM system are proxy (for the WI system) and Python flask module (for the MM system).

The react app is built on the localhost:3000 by default, and we add a proxy with path localhost:5000 in the configuration file of the WI system so that the commands (in this case, the POST and GET requests) that cannot be identified by the react app (localhost:3000) will be sent to the proxy (localhost:5000). Then, the requests will be handled by the back-end python code. For example, on the user authentication page, users need to input their institution id, then the WI system will send a POST request to the back-end python code, the python file that contains the flask module will handle the request and send generated hashcode to the user's institution email address. Another case would be the create exam page. On this page, the supervisors need to input the exam name and time. Once the form is submitted, the input data will be sent to both another page (to display the exam info) and the MM system via POST requests. Then, the MM system will create the corresponding exam objects and do other operations.

For the triggers and cameras, the WI interface provides checkboxes for users to enable or disable any of them. The status of triggers and cameras are also sent to the MM system to control the involved triggers for the AI modules and hardware. Once the status is changed, a request will be sent to update the system.

3.2 MM System and & BE System

The connection between MM and BE will be TCP connection, where BE works as TCP server listens to packages and MM will send packages. Packages contain information like each frame, trigger status, command and others. For each type of command, a specific header will be sent together for BE to distinguish from each other.

4. Naming Conventions

4.1 WI System

System main page

Path: /

File:

- ai-invigilation-system/src/components/Auth.js
- ai-invigilation-system/src/components/Auth.css

Component: MainPage

Description: This page is the main page of the system, it provides two links to the separate authentication page for supervisors and staff members.

Supervisor authentication page

Path: /auth-supervisor

File:

- ai-invigilation-system/src/components/AuthUser.js

Component: AuthSupervisor

Description: This page is the authentication page for supervisors, it handles user input (institution id and access code) and authenticates users. Once the supervisor is authenticated, it redirects to the supervisor's home page.

Staff authentication page

Path: /auth-supervisor

File:

- ai-invigilation-system/src/components/AuthUser.js

Component: AuthStaff

Description: This page is the authentication page for staff members, it handles user input (institution id and access code) and authenticates users. Once the staff member is authenticated, it redirects to the staff home page.

Supervisor home page

Path: /supervisor-home

File:

- ai-invigilation-system/src/components/UserHome.js
- ai-invigilation-system/src/components/UserHome.css

Component: SupervisorHome

Description: This page is the main page for supervisors, it contains a sidebar on the left of the page and a content section. The sidebar can navigate users among available features, and the default page is set triggers.

Staff home page

Path: /staff-home

File:

- ai-invigilation-system/src/components/UserHome.js
- ai-invigilation-system/src/components/UserHome.css

Component: StaffHome

Description: This page is the main page for staff members, it contains a sidebar on the left of the page and a content section. The sidebar can navigate users among available features, and the default page is set up cameras.

System features (inside user home page)

1. Set triggers

Path: /supervisor-home/

File: ai-invigilation-system/src/components/user/SetTriggers.js

Component: SetTriggers

Description: This page is only available for supervisors and is the default page in supervisor home page. It contains a table displaying each trigger, users can enable/disable any triggers by checking/unchecking the checkbox in the status column.

2. Create exam

Path: /supervisor-home/create-exam

File: ai-invigilation-system/src/components/user/CreateExam.js

Component: CreateExam

Description: This page is only available for supervisors. It contains a form that receives user input and will pop up a new window displaying the exam info once the user saves the exam (submit the form).

3. Exam history

Path: /supervisor-home/exam-history

File: ai-invigilation-system/src/components/user/ExamHistory.js

Component: ExamHistory

Description: This page is only available for supervisors. It contains a table displaying each exam and links to the recording and reports of the exam stored in the local machine.

4. Setup cameras

Path:

- (for supervisors) /supervisor-home/setup-cams
- (for staffs) /staff-home/setup-cams

File: ai-invigilation-system/src/components/user/SetupCams.js

Component: SetupCams

Description: This page is available for both supervisors staffs while the two different paths redirect to the same page. It contains a table displaying each camera, users can enable/disable any cameras by checking/unchecking the checkbox in the status column.

5. Log out

Path:

- (for supervisors) /supervisor-home/log-out
- (for staffs) /staff-home/log-out

File: ai-invigilation-system/src/components/user/LogOut.js

Component: LogOut

Description: This page is available for both supervisors staffs while the two different paths redirect to the same page. It contains a link that logs out the user and redirects to the main page of the system.

Exam info page

Path: /exam-info

File: ai-invigilation-system/src/components/UserHome.js

Component: ExamInfo

Description: This page will be created once the user creates the exam (submit the create exam form), and it displays the information and a countdown of the exam remaining time.

4.2 MM System

Exam_Manager

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/exam_manager.py

Component: Exam Manager

Description: This module will be helper functions for all other modules.

- **generate_report()** will compose to **end_exam()** and **end_exam()** will be composed to another function in another module as one component. This function outputs the suspicious clips timeframe out as a txt file.
- **set_info ()** and **display_info()** will be components for **start_exam()**. These two functions set up the basic information of the exam and display it.
- **start_exam()** and **end_exam()** will be helper functions for module video_controller.

Student

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/student.py

Component: student

Description: This module defines the class student, with id(int) and suspicious(boolean) as features.

system

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/system.py

Component: Exam Manager, system

Description: This module does generate hashcode for users and authentication.

- **generate_supervisor()** function does generate a hashcode for users that are supervisors of the exam.
- **generate_staff()** function does generate a hashcode for users that are normal staff of the exam.
- **authenticate_supervisor()** function authenticate the hashcode of supervisors.
- **Authenticate_staff()** function authenticate the hashcode for staff.

Message

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/system.py

Component: student, message

Description: This module imports the student as an object.

- **show_message()** does interact with the WI system, when you click on the button of the flagged person, it will show the possible reasons that trigger the flag.
- **hide_message()** does interact with the WI system, clicking on the button with a hide label on it will conceal the message of the trigger.
- **dismiss_message()** does interact with the WI system, when a possible examinee keeps triggering the flag.

trigger

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/trigger.py

Component: trigger

Description: This module creates class trigger

- **active_trigger()** does interact with the WI system and BE system, when you click on the box of the certain trigger, it will activate the trigger you selected and pass the information through the MM system back to the BE system.
- **cancel_trigger()** does the same thing but totally reverse than the active_trigger(), you can unbox the certain trigger, it will deactivate the trigger you selected and let the BE system know though the MM system.
- **set_trigger() ???**

video_controller

File: ai-invigilation-system/AI_Invigilation/Exam_Manager/video_controller.py

Component: exam_manager, video_controller

Description: This module controls the class video_controller.

- **start_recording()** will start recording the footage of the camera and start the exam. It composed the function from exam_manager, including start_exam(), set_info(). The function will ask several inputs from the WI interface.
- **end_recording()** will end recording and use the function from exam_manager end_exam() to end the exam, which will save the crucial time points into a txt file to the local machine.
- **send_Next_detection_frame()** will send the camera image back to the BE system from the camera, and send back the result obtained from the BE though MM system back to the WI system.

4.3 BE System

AlgorithmControler

File: AI-invigilation-system\back_end\src\AlgorithmControler.py

Component: AlgorithmControler

Description: This module controls and manipulates all detection algorithms.

- **inference()** will take in a PIL image and run the detection algorithm and prase the raw result into our detection features.
 - Tricks: Original plan is to use multiple computer vision models to detect different features, but since we do not require very accurate results, some of them can be replaced by others. E.g. for head yaw degree estimation, instead of having a separate model to estimate it. We can use the result from landmark(i.e. The distance between left boundary and right boundary to the nose to estimate the head position). Similar to head pitch estimation

CommunicationManager

File: AI-invigilation-system\back_end\src\CommunicationManager.py

Component: CommunicationManager

Description: As we state above, this part is left unimplemented and focuses on the algorithm first.

StorageManager

File: AI-invigilation-system\back_end\src\StorageManager.py

Component: StorageManager

Description: This module controls and writes images to file.

- **saveFrame()** will take in a cv2 image and save it to present directory as a backup.

TriggerManager

File: AI-invigilation-system\back_end\src\TriggerManager.py

Component: TriggerManager

Description: This module is a singleton class that stores and manipulates triggers.

- **set_trigger()** will set the trigger to some threshold. 0 means the trigger is off
- **get_trigger()** will get the threshold for the trigger.

utli

File: AI-invigilation-system\back_end\src\utli.py

Component: cv2_2_pil pil_2_cv2 bb_2_rect dist_2d mid_point_2d sigmoid

Description: utility functions for convience.

- **cv2_2_pil()** will convert a cv2 image class to a PIL image class.

- **pil_2_cv2()** will convert a PIL image class to a cv2 image class.
- **bb_2_rect()** will convert a boundingbox (i.e. a numpy/python list of length 4) to a dlib rectangle class.
- **dist_2d()** will get the euclidean distance of 2 points.
- **mid_point_2d()** will get the middle point of 2 points.
- **sigmoid()** standard sigmoid function.

torch_mtcnn_custom

File: all file under torch_mtcnn_custom folder

Description: a custom version of <https://github.com/TropComplique/mtcnn-pytorch>























































- This is a custom version of python package mtcnn-pytorch, adjustment has been make to suit our need.
- The interface is remain original
- Some change been made: changed some removed function for pytorch, changed CUDA setting. Some numpy torch tensor transform is added to suit newer version.































5. GitLab Repo

5.1 URL







<https://gitlab.cas.mcmaster.ca/zhaoy243/AI-invigilation-system>

5.2 Commits

31 Jan, 2023 2 commits		
	fix the last commit error theFSO authored 14 hours ago	c3f2bdd2  
	merge to this repository theFSO authored 14 hours ago	28ffbac6  
30 Jan, 2023 1 commit		
	fixed some errors Xing Li authored 1 day ago	3c6addaf  
29 Jan, 2023 5 commits		
	Merge branch 'main' of https://github.com/yukizyx/AI-invigilation-system Xing Li authored 1 day ago	5f0cd697  
	update video controller Xing Li authored 1 day ago	0eefd424  
	Implement create exam page (still need router) yukizyx authored 1 day ago	52eb2e28  
	Implement create exam router yukizyx authored 2 days ago	2699777e  
	Implement exam history page yukizyx authored 2 days ago	15580d35  
25 Jan, 2023 4 commits		
	Implement set triggers + cams yukizyx authored 5 days ago	a8b7f539  
	Implement routers and links yukizyx authored 5 days ago	69e9b4d8  
	Implement staff homepage yukizyx authored 5 days ago	d2dd50e6  
	trigger & message python file created Xing Li authored 5 days ago	4a9784a8  
24 Jan, 2023 3 commits		
	video controller class created Xing Li authored 6 days ago	d4df4925  
	Update General Functions Xing Li authored 6 days ago	694e4504  
	update general function - set trigger Xing Li authored 6 days ago	11be6a4f  
23 Jan, 2023 1 commit		
	Implement log out + add api modules yukizyx authored 1 week ago	0cd15b8d  
17 Jan, 2023 2 commits		
	Fix sidebar yukizyx authored 2 weeks ago	0bc46fb4  
	Build main structure of user homepage yukizyx authored 2 weeks ago	3643e821  

16 Jan, 2023 10 commits		
	Update workspace.xml Xing Li authored 2 weeks ago	abe54766  
	MM code updated Xing Li authored 2 weeks ago	b8ff162d  
	MM code updated Xing Li authored 2 weeks ago	98036673  
	MM code updated Xing Li authored 2 weeks ago	da87b7bd  
	? Xing Li authored 2 weeks ago	b9aa8dce  
	Ready to add links + routers yukizyx authored 2 weeks ago	3e5ee22d  
	Implement main page yukizyx authored 2 weeks ago	0c48f558  
	Initialize react app yukizyx authored 2 weeks ago	74a4165e  
	Update README.md Yuki Zhao authored 2 weeks ago	Unverified f374d5fd  
	Initial commit Yuki Zhao authored 2 weeks ago	Unverified 1afbea7d  

Below is commit for back_end (BE), was in separate repository

Commits on Jan 30, 2023	
	add triggers theFSO committed yesterday
	add trigger theFSO committed yesterday
	more stage 1 stuff theFSO committed yesterday
	stage 1 finished theFSO committed yesterday
	add gitignore theFSO committed yesterday
Commits on Jan 18, 2023	
	add backend, face detection algorithm is done theFSO committed 2 weeks ago