School of Computer Science

University of Nottingham Ningbo China (UNNC)

**COMP2043. GRP Interim Group Report**

Smart Class Attendance Taking System

Group 17

Supervisor: Zheng LU

**Name Username**

Boya Wang slybw2

Mingchen Li scyml3

Yiming LI scyyl3

Guohao Yu scygy1

Hongyi Zhu scyhz1

2019.12.0

**1. Updated and extended the description of the problem**

There are two main problems in this project to be solved, which is face recognition and fully functional system implementation such as management.

**Face Recognition implementation:**

The face recognition function can be called from the third-party API of Python libraries as described in the requirement, finding an existing model which works well for Asian face is the highest priority problem.

**Face Recognition function:**

For face recognition, the core of this problem is to recognize and identify the faces of the students enrolled in a specific module, recording arrival time of teaching sessions of the students given the time schedule once the face of the student is identified. Statistical data such as how many students have attended for the current teaching session should be shown as a bar of percentages and the name of the recognized student should be shown on the face recognition UI.

The process of face recognition can be stopped by teachers at any time.

After stopping face recognition, the percentage of attendance and who are missing should be shown as the result and the software should send this information to the remote database.

There are some additional subproblems in face recognition problem as well.

The process of face recognition can be paused and continue manually by buttons on face recognition UI.

While doing face recognition, the function may be able to identify the faces concurrently when several students appear in the recognition area to improve the efficiency. Since the face recognition implementation is called from third-party API, this problem can be solved initially.

**Fully Functional System implementation:**

The whole system should have a well-organized GUI in front end to simplify the human-computer interaction, and a remote database to store all the relative information, including information of students, information of teachers and attendance information.

The system has two kinds of access, one is for teacher, the other is for admin. Both of them need to login in the system with their unique username and password.

1. For admin, he has permission to manage the whole system. The core work for admin is to manage the remote database.

He can easily access each level in management system from a drop-down box in GUI, which contains several levels such as faculties, majors, modules, teaching sessions, teachers and students. Some of them may be at the same level. In each level, admin can update or delete the existing information, and add or import the extract information. In student level and module level, if some specific students missed classes more than 3 times, there will be a warning, so that the admin can quickly alert the relative staffs.

1. For teachers, they have permission to use the system. They can start the face recognition when it's about time. There are several modules shown like the dashboard in moodle.

The teacher can access their teaching modules and their teaching sessions shown as a list. If the teaching session has not to be finished, the teacher can start the face recognition for this specific teaching session. After the teaching session has done, the teacher can review the attendance result of that teaching session.

The teacher can search for the specific student and specific teaching session as well to quickly check the attendance of students and teaching sessions.

Both teachers and admin can export what they what, including attendance list for specific students, modules or teaching sessions.

1. **Survey of any existing systems that address similar problems**

**Comparison with existing software**

There are significant number of facial recognition attendance taking system used in companies and universities nowadays. Following are two examples of existing systems.

1. **DingTalk:**

DingTalk, a professional office application for small and medium-sized enterprises (SMES), is an integration of multiple functionalities. Facial recognition attendance system is applied on DingTalk whose version is over 4.0 by using Ding Talk M2 Smart Receptionist and DingTalk C1 Smart Communication Center, which can identify whether picture or an alive person and recognize it even under extreme lighting condition. The whole system is able to be set up in 4 steps, downloading DingTalk, connecting facility by blue tooth, connecting WIFI and bonding with specific group, which is extremely user friendly. The system can identify multiple people quickly when they expose to the camera. Once the person passes the facial recognition, the attendance sheet will be updated by the system. However, facial recognition attendance system is only a small part of the whole DingTalk.

By 2018, DingTalk had have more than 100 million users and became one of the world’s largest professional communication and management application in China.

1. **Jibble:**

Jibble is an application mainly for record the attendance with sign in time and sign out time if passed the facial recognition. Then the system offers timesheet views and report to the administrator. By using this application, the administrator can have direct access to employee statistics and improve the productivity. Jibble system is based on mobile devices like iPad, so it can only recognize one person at one time using the front camera.

However, Jibble doesn’t have multiple functionalities like DingTalk. Since the system works on individual device, cannot identify multiple person does not affect a lot.

**Market research and interview**

According to our market research, our teacher in university are willing to use a software which can take the place of attendance sheet. Problems of current attendance sheet, what feature is expected to be applied in the software and the time requirement is showed below.

1. Problems of current attendance sheet:
2. Inconvenient for teacher
3. Need to print out the attendance sheet and store the attendance sheet into computer manually
4. Student can sign or each other, since there is no verification
5. Expected features:
6. GUI should be user friendly.
7. Easy and convenient to do the attendance recording.
8. Offer the list of students who has be absent for more than three times
9. Searching by lecture name and show details
10. Comparation of attendance rate of two lectures in same module
11. Comparation of overall attendance rate of two modules
12. Make sure the safety of data
13. Students can only sign in for themselves
14. Time requirement:
15. If record the attendance after class, should make sure it will finish in ten minutes
16. The system should be stable and accurate to record as soon as possible

In conclusion, the system should be easy to use for teachers and reliable enough to reduce the time of recording attendance. In general, by using this system, student cannot sign for each other anymore and the attendance sheet can be stored automatically.

**Technical research**

1. Platforms:
2. Anaconda is used to finish our project since we were asked to finish it by Python. The version of anaconda is over 3 and it contains 3 bask components, which are Anaconda Navigator, Jupyter Notebook and Spyder.
3. OpenCV packages are imported for picture processing.
4. There are 3 suitable packages for UI design. However, the UI made by tkinter is too simple. Pyqt is faster and easier than kivy. As a result, pyqt is used to design UI.
5. Tools

Camera is the solely tools that the project requires.

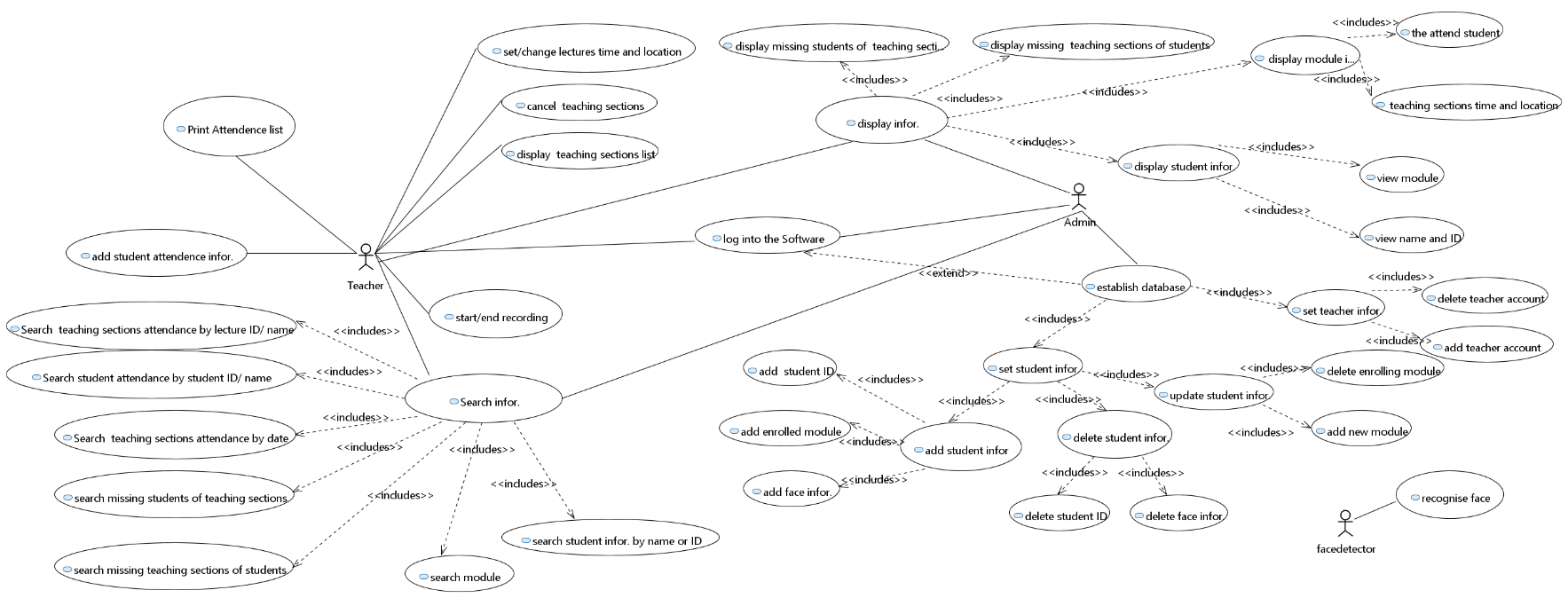
1. Technologies

Face recognition system and school management system is needed.

1. Algorithms

Machine learning algorithms are used. The program should learn from the existing student pictures stores in database and recognize whether the student is in the relative department.

**3. SRS**

**Use case diagram**

**User requirement:**

1. **User characteristics:**

The main users include database administer, teachers, students and a face detector.

1. **Functional requirement:**

* The users can interact with the system through a graphic user interface.
* Common functions:

1. User login
2. User logout

* For face detector

1. Recognize student face image
2. Identified student face image
3. Signal identification success or failure

* For administer

1. Add/ update/ delete teacher accounts in user database
2. Import/ update/ delete student information in database
3. Import/ update/ delete information of teaching session in database
4. View database

* For teacher

1. View his own profile
2. Set/update name and time of teaching session
3. Start/continue/end class attendance recording
4. Search information based on teaching session/student name
5. View searched information
6. View attendance sheet of a module/teaching session/student
7. View immediate attendance condition for a class
8. Edit attendance information and add remark of the edit
9. **Un-functional requirement:**
10. Availability: the administer shall check and maintain the database or server, set software maintenance time and make sure the system can run during the work period.
11. Reliability: the system shall recognize and identify students correctly and prompt notice if the student cannot be identified.
12. Safety: the system shall prompt the name of identified student in case of mis-recognizing the student.
13. Security and Privacy: only users holding the role "admin" can access database and server.
14. User friendliness: the system shall provide user-friendly GUI, user-friendly operation button, immediate relative notice to the users if their operation is wrong.
15. Maintainability: make use of the structural software design method, OOP and quality management.
16. **User interface requirement:**
17. Develop Graphical User Interface
18. The UI should be able to support all operations noted in functional requirement
19. The UI should be clear to understand, and easy to use
20. The UI should be responsive in short time
21. The UI should be consistent on all interfacing screens
22. The UI should have a proper appearance which suits its users

**System Requirement:**

1. **Functional requirement:**

* **Common functions**

1. **User login**

Pre-condition: user logged out

Operation: enters user school ID and password, click confirm button.

Post-condition:

* If information is correct, change user state to log in. Show the main page.
* If not: stay user state, stay in the current page. Prompt a warning window asking the user to enter input again.

1. **User forget password**

Pre-condition: user logged out

Operation: click *forget password* link

Post-condition: Show help page with instruction text on it

1. **User logout**

Pre-condition: user logged in

Operation: click *teacher Info* link, and click logout option

Post-condition: change user state to logout, and show log-in page

* **For face detector**

1. **Recognize student face image**

Pre-condition: face detector is on and the user clicked start recording button

Operation: locate a face, read in and store a shot-cut of the face

Post-condition: a frame of face recognition stream is loaded in and stored in local file

1. **Identify student face image**

Pre-condition: a face has been recognized

Operation: compare face data and identify the corresponding student

Post-condition:

* If a student is identified, add his attendance information
* If the face matches no student, return false value

1. **Signal identification success or failure**

Pre-condition: a face has been processed with identification operation

Operation:

* If recorded successfully, show name and ID of identified student on page
* If recording failed, show error message on page

Post-condition: the user receive feedback message of operation

* **For teacher**

1. **View his own profile**

Precondition: the teacher account must be logged in

Operation: click on his own profile link

Postcondition: show the view of current teacher account’s information.

1. **Set/update lecture name and time**

Precondition: the teacher account must be logged in

Operation:

* choose a specific lecture to edit
* Input lecture name or lecture time
* click confirm button to save changes

Postcondition: lecture name and time is modified.

1. **Start and end class attendance recording**

Precondition: the teacher account must be logged in

Operation:

* choose a specific lecture to record attendance
* click the start button to start recording
* can be stop by clicking the stop button
* can continue recording by clicking the continue button after stop
* click end recording button to finish recording

Postcondition: show the view of attendance information of this lecture and the attendance information will be stored in the database.

1. **Search information based on lecture/student name**

Precondition: the teacher account must be logged in

Operation:

* type in the keyword as lecture or student name
* click search button to start searching

Postcondition: show the view of lecture or student list including the keyword that has been entered.

1. **View searched information**

Precondition: the teacher account must be logged in and has searched for a keyword of lecture/student name

Operation: click on a specific lecture

Postcondition: show the view of attendance information of this lecture.

1. **View attendance sheet of a module/teaching session/student**

Precondition: the teacher account must be logged in

Operation: click on the specific module/teaching session/student

Postcondition: show the view of attendance information of this a module/teaching session/student

1. **View immediate attendance condition for his class**

Precondition: the teacher account must be logged in and has started recording

Operation: /

Postcondition: show the percentage of attendance of current lecture on the right bottom of recording windows.

1. **Edit attendance information and add remark of students**

Precondition: the teacher account must be logged in

Operation:

* choose a specific student
* choose a specific lecture to edit this student’s attendance status
* change the status of current student’s attendance
* add notes to this change
* click confirm button to save changes

Postcondition: The student’s attendance status of this lecture has been changed.

1. **Un-functional requirements:**
2. Availability: the administer shall check and maintain the database or server. When the database or the server loses connection, the system must have a warning alert and try to reconnect.
3. Reliability: The expected number of users that the system can support is around 30,000. To achieve this goal, the database should be large enough to accommodate at least 30,000 students, teachers and lectures records. Furthermore, the system must ensure that it does not crash when using. Also, the system shall prompt notice if the student cannot be identified and it shall allow teachers to edit attendance information manually. When this is done, the reliability of the system can be guaranteed.
4. Safety: the system shall prompt the name of identified student in case of mis-recognizing the student. The system shall recognize and identify students correctly. A confirmation message should be alerted to let students to confirm their information.
5. Security and Privacy: All users’ information should be stored in a separated database on the cloud. Documents stored in a separate database can improve the security of the data, only users holding the role "admin" can access database and server and the privacy of students’ and teachers’ information can also be guaranteed.
6. User friendliness: the system shall provide user-friendly GUI, user-friendly searching operation, immediate alert windows will be notified to the users if their operation is wrong.
7. Maintainability: make use of the structural software design method, OOP and quality management. The administer should be able to add, modify and delete student or teacher or module or lecture information separately or all at once by reading from files. The system must allow administer to update the face identification and recognition algorithm.

**User Interface requirement:**

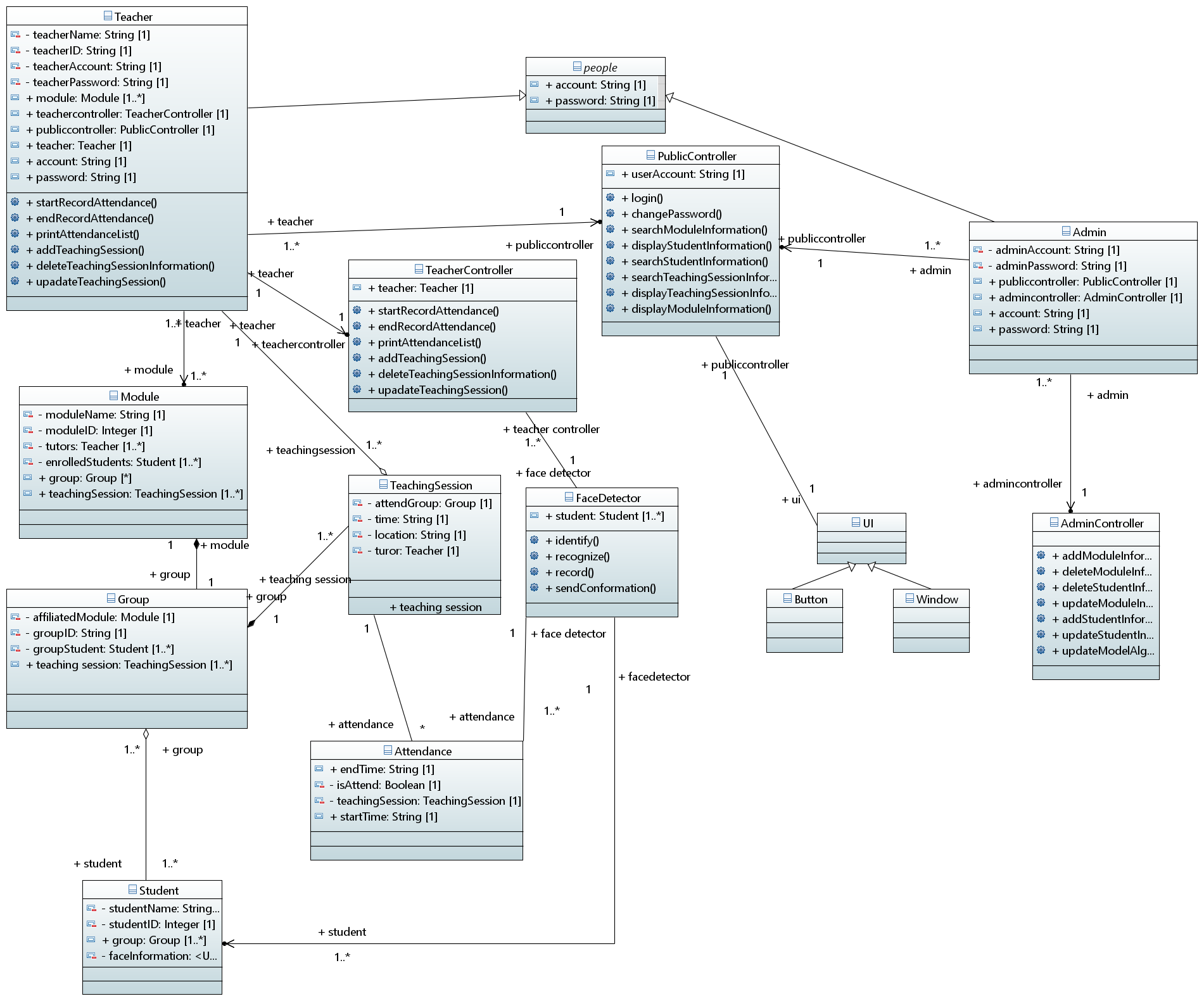
* **Basic design principle**

1. The sequence of user operations and composing of icons of UI should be consistent in view of the logic of user action.
2. The function composing of UI should be organized. For example, it should be delivered with rules of hierarchy and composition.
3. Implement UI with Qt and PyQT5 for better coupling of system functions and portability.

**Method for the system**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method name | Input | Type | Output | Type |
| login | Username (University ID) | String | checkMessage | Bool |
| password | String |
| addModuleInformation | ModuleName | String | checkMessage | Bool |
| TeacherID | String |
| StudentID | Integer |
| ModuleID | String |
| GroupID | Integer |
| deleteModuleInformation | OriginalModuleName | String | checkMessage | Bool |
| changePassword | oldPassword | String | newPassword | String |
| deleteStudentInformation |  | void | checkMessage | Bool |
| updateModuleInformation | OriginalModuleName | String | ModuleMessage | String |
| OriginalTeacherInformation | String |
| NewTeacherName | String |
| NewTeacherID | Integer |
| addStudentInformation | studentName | String | checkMessage | Bool |
| studentID | String |
| studentPhoto | String |
| attendModule | String |
| attendLectureTime | Integer |
| updateStudentInformation | OriginalStudentInformation | String | checkMessage | Bool |
| StudentID | Integer |
| StudentName | String |
| ModuleInformation | String |
| SearchModuleInformation | ModuleName | String | specificModule | String |
| ModuleID | String |
| displayStudentInformation |  | void | SpecificStudentInformation | String |
| SearchStudentInformation | StudentName | String | specificStudent | String |
| StudentID | String |
| SearchTeachingSessionInformation | 6-digit date | Integer | specificLecture | String |
| LectureName | String |
| deleteTeachingSessionInformation | OriginalLecture | String | allLecture | String |
| displayTeachingSessionInformation |  | void | specificLectureInfor. | String |
| StartRecordAttendance |  |  |  | Null |
| EndRecordAttendance |  |  | checkMessage | String |
| PrintAttendenceList | AttendenceOfLecture | String | AttendenceList | String |
| displayModuleInformation |  | Null |  | Null |
| updateTeachingSession | OriginalLecture | String | LectureMessage | String |
| Location | String |
| time | Integer |
| 6-digit date | Integer |
| addTeachingSession | LectureName | String | LectureMessage | String |
| Location | String |
| time | Integer |
| 6-digit date | Integer |
| UpdateModelAlgorithm | OriginalAlgorithm | String | Confirmation\_message | String |
| NewAlgorithm | String |

**Class diagram**

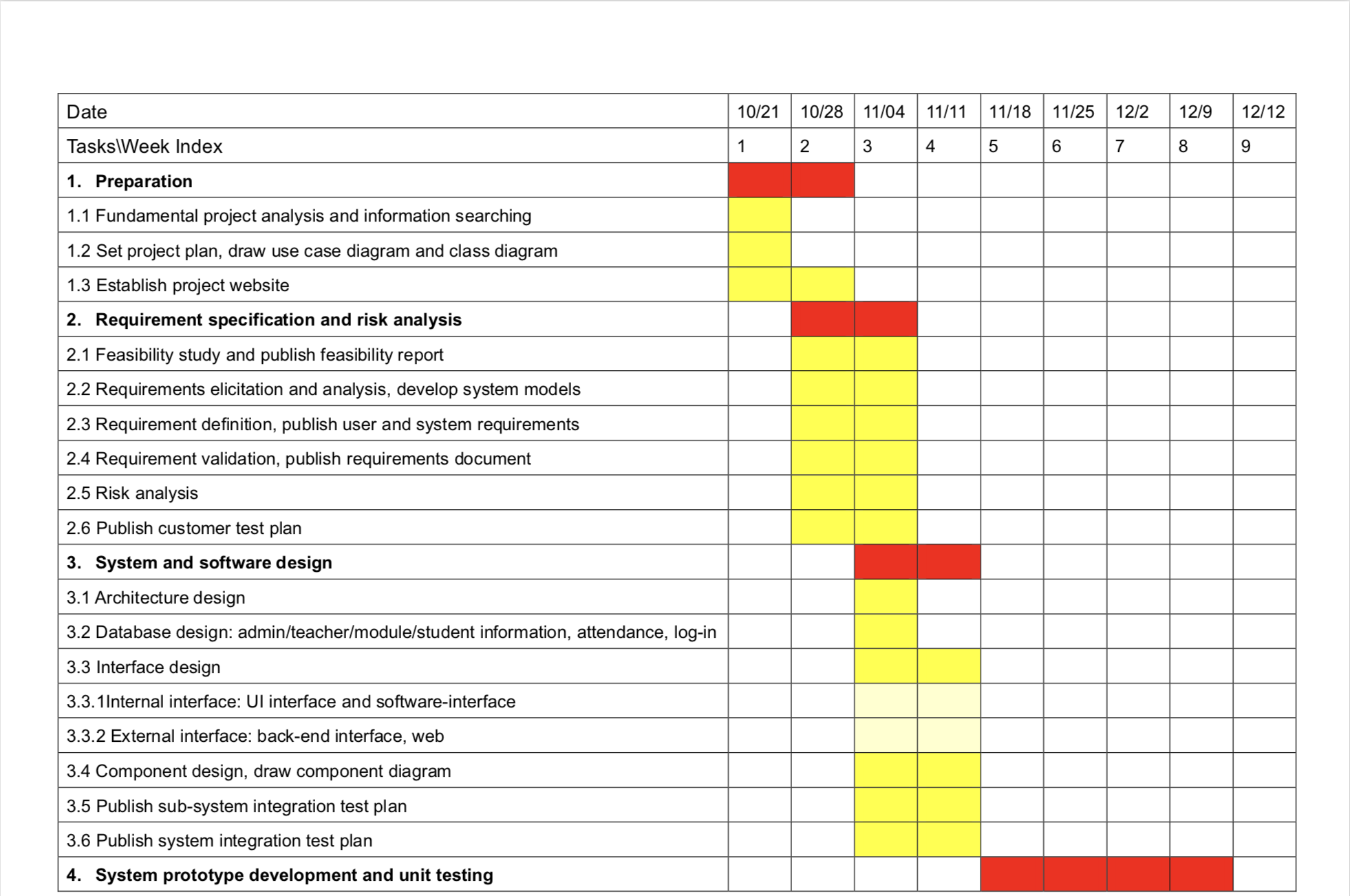
****

**4. Initial Design and Interface**

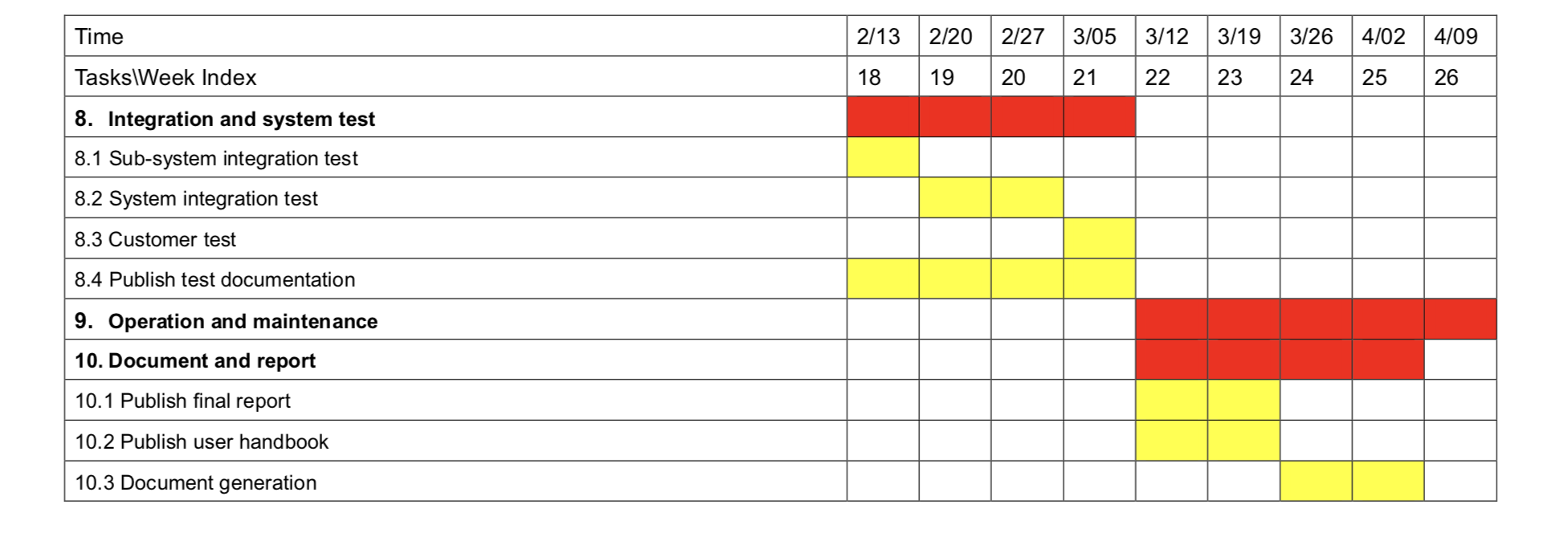
**5. Result of any initial implementation**

**6. Discussion of any problems encountered**

**7. Time plan**

****

****

****

**Appendix I – UI interface**

**Appendix II – formal minutes**

**Appendix III – informal minutes**