الكلية الجامعية للعلوم التطبيقية
University College of Applied Sciences

Programming language 2 Project

(e-Exam System)

Semester II 2022/2023

# Idea

The training and learning process should be validated using varied assessment tools, one of them is an e-Exam system.

# Main actors. *[The main people whose use the system]*

The main persons should affect the system are:

1. Admin
    - U1.1. should be able to add teachers
    - U1.2. should be able to add students
    - U1.3. should be able to view all students
    - U1.4. should be able to view all teachers
2. Teacher
    - U2.1. should be able to add students
    - U2.2. should be able to view all students
    - U2.3. should be able to add questions
    - U2.4. should be able to edit questions
    - U2.5. should be able to view all questions
    - U2.6. should be able to add exam
    - U2.7. should be able to view all exams
    - U2.8. should be able to view exam's results
    - U2.9. should be able to view student's exam result
    - U2.10.(Optional) should be able to export exams to a text file
3. Student
    - U3.1. should be able to solve exam
    - U3.2. should be able to view his/her exam results
    - U3.3. should be able to view his/her exam result details
    - U3.4. should be able to edit his/her information
    - U3.5. (Optional) should be able to export results to a text file
    - U3.6. should be able to view his/her rank in average of exams result
    - U3.7. (Optional) should be able to export an exam to a JSON file

# Main UI

This project includes many user interfaces; you should at least implement the following UIs:

UI.1. Login Screen
UI.2. Admin Main Menu Screen
UI.3. Teacher Main Menu Screen
UI.4. Student Main Menu Screen
UI.5. Add Teacher Screen
UI.6. Add Student Screen
UI.7. Add Question Screen
UI.8. Add Exam Screen
UI.9. View all questions Screen
UI.10. View Exam Results Screen
UI.11. View Student Results Screen
UI.12. View Students Ranks Screen
UI.13. Start Exam Screen

Note: You are free to add any additional screens.

# Functional requirements

Initially, UI.1 will starts and allows user to enter a **username** & **password**, then - and after validation process - the system will dynamically detect the user type, according to the following criteria:

- If user type is an Admin, then the system will start UI.2.
- If user type is a Teacher, then the system will start UI.3.
- If user type is a Student, then the system will start UI.4.
- Otherwise, display an error message.

The UI.2, UI.3 and UI.4 each of these screens should have a content according to the description in #Main actors' part.

Requirements description:

R.1. Add teachers (Belongs U1.1)

When user starts UI.5, user should fill the following teacher's information (username, password, name, birthdate, gender, salary, specialty) and don't forget to validate all fields to be in a good format, then save the data.

R.2. Add students (Belongs U1.2 and U2.1)

When user starts UI.6, user should fill the following student's information (username, password, name, university id, birthdate, gender) and don't forget to validate all fields to be in a good format, then save the data.

R.3. View all students (Belongs U1.3 and U2.2)

When user starts (UI.2 or UI.3), briefly display data for all students in a table (username, name, university id, gender)

R.4. View all teachers (Belongs U1.4)

When user starts UI.2, briefly display data for all teachers in a table (username, name, field, gender)

R.5. View all questions (Belongs U2.5)

When user starts UI.9, briefly display data for all question (id, question text, type, marks)

R.6. Add questions (Belongs U2.3)

To create an exam, we should have predefined questions, so user will use UI.7 for that purpose. All questions have the following properties (id, question text, marks weight, descriptive image [optional]).
However, question has many types:

1. Yes-No question: each question should have an answer which could be true or false.
2. Multiple Choice Question (MCQ): each question should have multiple choices (could be 6 or less, but not less than 2), and the answer is a choice number.
3. Fill the blank question: each question should have a short text as answer
4. (Optional) Order question: this type contains a list of text, each one has its order in the list, you should view these texts in a random order for students, and the true answer is the real order for these texts.
5. (Optional) Matching question: this type contains a grid of texts filled in columns and n of rows, for example: a list of terms and its definitions. You should display these data for each column in a random order for students, and the true answer is the real match between two columns.

R.7. Edit question (Belongs U2.4)

By selecting one of R.9 items, a new UI similar to UI.7, but already filled with selected question data to allow user to edit and save new data.

R.8. Add exam (Belongs U2.6)

When user starts UI.8, and should fill exam information (number of questions, min pass average, allowed question's type). all questions related to exam should be assigned randomly at the moment of creating the exam.

R.9. View all exams (Belongs U2.7)

When user starts (UI.3 and UI.4), it should briefly display data for all exams (id, name, number of questions, min pass average, total marks) that added by logged in teacher only (in UI.3) and you will display all available exams (in UI.4).

R.10. View exam's results (Belongs U2.8)

When user starts UI.10, it should display exam information, all students that already passed the exam and the results for each one.

R.11. View student's exam result (Belongs U2.9 and U3.2)

When user starts UI.11, it should display student information and all related results from exams that already he passed.

R.12. (Optional) Export an exam to a text file (Belongs U2.10)

When user start UI.3, By selecting one of the displayed exams, a click on export button will be available to click, and by clicking to that button, a pop up screen will appear to select a folder to save a file.

R.13. Solve exam (Belongs U3.1)

By starting UI.4, a list of available exams in the system will displayed, and by selecting of them, a UI.13 will start, it will displays the questions sequentially, one in a time, and some of exam's data will be available all the time. After finishing the exam, the result should have displayed immediately.

R.14. View his own result details (Belongs U3.3)

After R11, user can choose one of results, then a UI similar to UI.13 will view the questions sequentially, one in each time, each question viewed should display the student answers, the correct solution and the marks student have.

R.15. Edits his information (Belongs U3.4)

After user starts UI.4, the student should have the ability to starts UI similar to UI.6 to update only the following fields (password, name, birthdate).

R.16. (Optional) export his results in a text file (Belongs U3.5)

After user starts UI.4, the student should have the ability to export all exams' results.

R.17. Explore his rank in average of exams result (Belongs U3.6)

After user starts UI.12, which contains a list of top 5 students in average of exams' result in the system.

R.18. (Optional) Export an exam to a JSON file

When users start UI.3 and selecting one of displayed exams, they can click on export button, which displays a pop up screen to select a folder to save a file.

# Scenarios

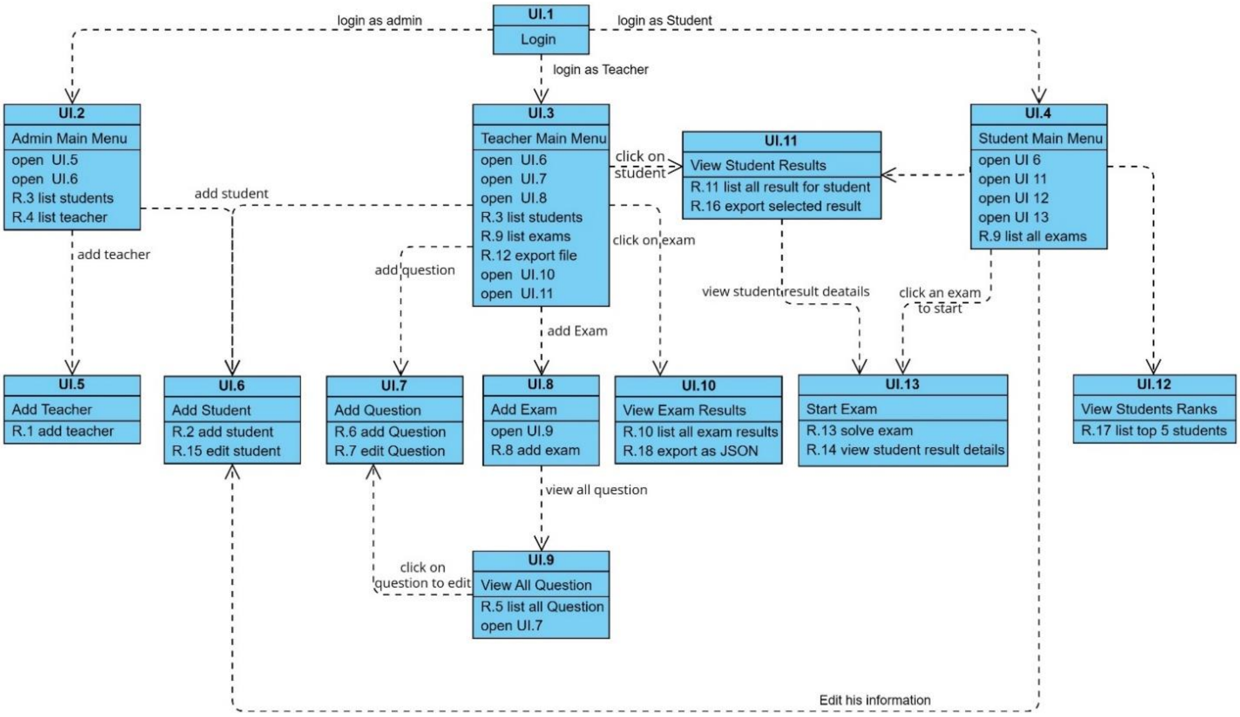
The scenarios depend on actor's data. The available scenarios are:

S1. Admin logged in using UI.1, so UI.2 will starts, which contains many buttons or menus that allow them to add teacher (R.1) or add student (R.2). in addition, the UI contains a list of all students (R.3) or of teacher (R.4) in the system. Finally, admin can logout and go back to login UI.

S2. Teacher logged in using UI.1, so UI.3 will starts, which contains many buttons or menus that allow them to add student (R.2), add questions (R.6), add exams (R.8) or export all created exams to a text file (R.12). in addition, the UI contains a list of all students (R.3) and list of all exams that are created before in (R.9). By clicking on add Exam button, UI.8 will starts to allow user to fill all exam data, also this UI could contain a button, to view all question available in the system (R.5). In UI.9 by selecting any question of the displayed list, user can edit it (R.7). In UI.3 by selecting any question of the displayed list, this will start UI.10, which contains all students' result for this exam (R.10). In UI.3 if user selects a student, this will start UI.11, which contains all result for this student (R.11).

S3. Student login using UI.1, so UI.4 started, which contains many buttons or menus that allow user to view all results (R.11), start a new exam (R.13), edits information (R.15), monitor the top five students (R.17). When UI.11 starts and the student selects a result from the table, then (R.14) will starts to display mode details, UI.11 should contains button that allow the user to export the selected result in a text file (R.16).

# User story flow



# Deployment

This project should be worked by **<u>NetBeans IDE</u>**, in a project named with your **first name** then your ID (e.g. **Heba_220220000**). in the project folder you should create **group.txt** file that contains full name and ID for each student in the group, then you should compress project (e.g. Heba_220220000.rar or Heba_220220000.zip) and deploy the compressed file using class room.

Notes

- ☒ if the group contains two students, one of them should deploy the project file, and other should deploy **group.txt** only
- ☒ You should deploy first phase (OOP classes) in 26/4/2023.
- ☒ Deadline for full project 18/5/2023, and any delayed project will not accepted
- ☒ you must have at least classes name mentioned in description, but you can create more classes as you need.
- ☒ The full mark is depending on some notes:
    - ✓ Uploaded file name matching the structure name (**Name_UniversityID.zip**)
    - ✓ Any Student **copy** code from other, both will get **Zero** mark
    - ✓ Required functions and applied functions.
    - ✓ Final Discussion.
- ☒ The project can be prepared by groups; each group contains only one or two students and each one should have his/her copy in the discussion.

# Test Data

+ Admin could login with username (admin) and password (admin)

+ Students' data

Username	Password	Name	University id	Birthday	Gender
s1	123456	Ahmed Ali	120180001	1/1/2000	Male
s2	123456	Heba Ahmed	120190001	1/1/2001	Female
s3	123456	Dalia Mohammed	120200001	1/1/2002	Female

Students' group should add they data as a test.

+ Teachers' data

Username	Password	Name	Birthday	Gender	Salary	Field
JafarAgha	112233	Jafar Al-Agha	1/1/1997	male	2000	Eng
MohDalo	112233	Mohammed Al-Dalo	1/1/1998	male	1000	Eng
EtafHadda	112233	Etaf Abu Hadda	1/1/1999	female	3000	Eng
HashemSaqqa	112233	Hashem Al-Saqqa	1/1/2000	male	1500	IT

+ Yes-No Questions' data

Id	Question text	Marks	Image	Answer
1	Does Java can build GUI?	2	null	Yes
2	Java doesn't support OOP?	2	image2.jpg	No

+ Multiple Choices Questions' data

Id	Question text	Marks	Image	Choices[]	Answer
1	Which Type is not primitive	2	null	[String, int, double, char]	0
2	Which class is super class for all classes	2	null	[String, Object, number, Exception]	1

+ Fill the blank Questions' data

Id	Question text	Marks	Image	Answer
1	What is the keywords that used to call super class constructor?	2	null	super
2	What is the keywords that used to inherit from a class?	2	null	extends

+ Order Questions' data

Id	Question text	Marks	Image	steps[]	orders[]
1	Order the following access modifiers from most restrictive to low	2	null	[package, public, private, protected]	[4, 2, 1, 3]

+ Matching Questions' data

Id	Question text	Marks	Image	terms[]	definitions[]	Answer matching[]
1	Matching the following terms	5	null	[Class, attribute]	[define a properties in real world, define an entity in real world]	[1, 0]

Good Luck ^_^