LAB 3: "VIRTUAL UML SPEED REFACTOR/EXTEND CHALLENGE"

Aims:

- Practice your object-oriented analysis and design/maintenance skills
- Practice working in small design teams
- Consider how to design software with lower maintenance effort in the future

PREPARATION AND POST PROCESSING

This lab session is a **group exercise,** and you are asked to work with your **COMP2002 group project team members** on this! The way you do it, is up to you. You can go for face-2-face meetings in A32, any other location, or use Teams and do it online (using your group project Teams site). In the latter case you should use a virtual whiteboard, so that you can draw diagrams together. If you have questions, ask the lab helpers if you are in Lab A32 or ask us on the Teams Questions channel, if you are elsewhere.

To set the scene, imagine you are competing in a "**Virtual UML Speed Refactor/Extend Challenge**". Here are the rules: Overall you have **120 minutes to fulfil the given task**. It is advised that you split up your precious time into two chunks: Use 90 minutes for the design and 30 minutes for summarising the outcome in form of a Power Point presentation. To get through the entire task in time you might want to have an initial discussion with the whole teams and then split up into smaller teams (perhaps pairs, if that fits) to do some 15-minute sprints. After each sprint you then reconvene briefly as a group and give some feedback to each other's outputs.

At the end of the lab session each group project team is welcome to submit a Power Point presentation (**one per group**) with a summary of the outputs you produced during the lab session (use case specification and screenshots / photos of UML diagrams). Your submissions will not be marked, but we are very interested to see what innovative and adventurous designs you are coming up with within the short time given. The submission link on Moodle will only be open until the end of the lab session. Please understand, that due to time constraints, we will not be able to provide individual feedback for each of the submissions.

THE STUDENTSHIP MATCH APPLICATION

You have been asked by the Student Union to develop a new "Student-Internship Match" application for them. A legacy system exists but it was abandoned some years ago. The idea is to match students with the right skillset to internship offers provided by local companies.

Before students and companies can use the application, they need to register and once they received their username/password they can log in. After registration students and companies need to add a profile which will be stored in a database (together with their login details). The profile can be edited at any time. Companies can then start posting job offers which will be stored in the database. Student profiles consist of student details and skills while job offers consist of company details, a job description, and requirements. Skills and requirements are to be submitted in form of predefined keywords so that they are easy to be matched in later searches.

Once the data has been stored in the database students as well as companies can then query the database and a query engine will try to match job offers with student profiles and vice versa. At a later stage the student union is planning to replace the keyword search with an intelligent query engine. It is a requirement that the application is designed following object-oriented principles and that it is easy to maintain and extend. If you have any ideas for extensions in order to improving the usability of the application, you are encouraged to add these to your design.

The legacy system was operated by a secretary who would handle all the data input and search requests. The only surviving design artefact from this system is the class diagram shown in Figure 1.

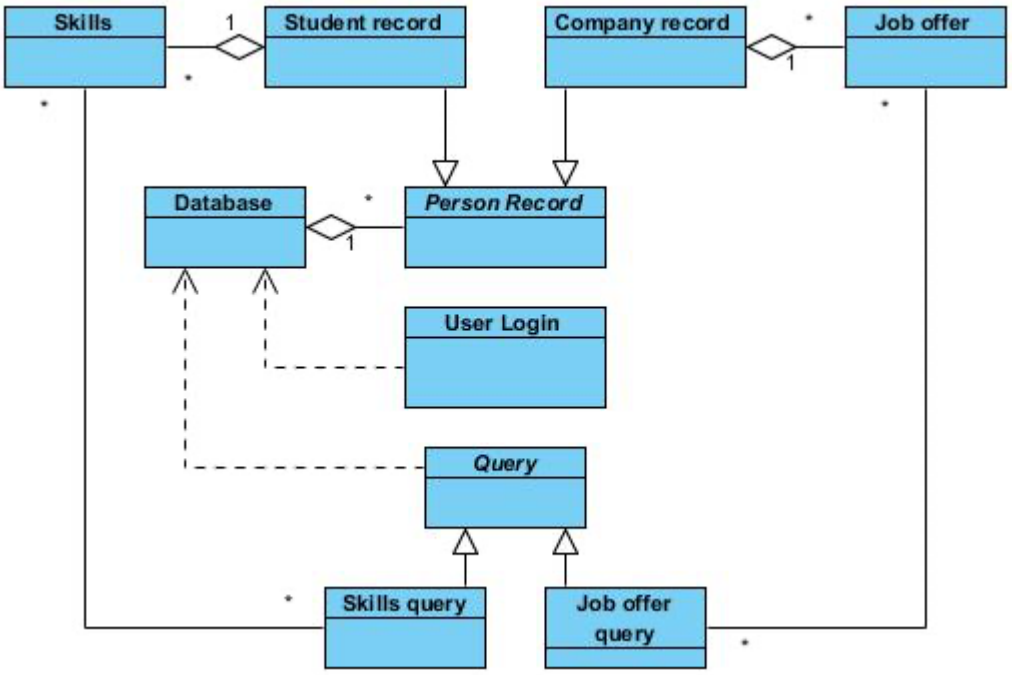

Figure 1: Legacy system class diagram

YOUR TASK

Find your **COMP2002 group project team members** and work on the object-oriented analysis and design for the refactoring and extension of the "Student-Internship Match" application.

You are asked to produce the following for the new version of the application:

- Use-case diagram for the described application
- Complete use case specification for a nontrivial use case (e.g. "search for jobs")
- Activity diagram of the same use case

- Sequence diagram of the same use case
- Class diagram
    - Classes including key attributes and operations
    - Relationships
    - Multiplicity indicators
- Nontrivial state machine diagram for one of the classes

SOME TIPS

In case you want to use Visual Paradigm for creating your diagrams, here are the links:

- Visual Paradigm Community Edition:
    - https://www.visual-paradigm.com/download/community.jsp
- Visual Paradigm User Guide:
    - https://www.visual-paradigm.com/support/documents/vpuserguide/
- Visual Paradigm Online:
    - https://online.visual-paradigm.com/diagrams/

To improve maintainability in the future, think about these points when you design your system:

- Keep it as simple as possible (KISS principle)
- Keep similar functionality together, and different functionality apart (encapsulation)
    - This is also known as "high cohesion, and loose coupling"
- How easy would it be to change a module or feature in the future?
- How good are your diagrams at explaining the system to a new programmer?