CS 476: Software Project and Demonstration Requirements

Dr. Samira Sadaoui

Through the software projects, students will gain experience in designing and implementing real-life applications (distributed, real-time and if possible concurrent). Also, they will learn managerial skills vital for the project success. Students will develop high-quality application software that is Web-based (browser-based interfaces), user-friendly (with nice GUIs) and distributed on three tiers at least.

1. Timeline

1. Each team of two students should submit via UR Courses the project title as well as the functional and quality requirements no later than September 19th.
2. On November 19th, each team should submit the complete project report, programs and presentation slides on UR Courses. Also print the entire project report.
3. Project presentation and software demonstration will start on November 19th. The ordering of the presentations is determined by a random draw during the lectures. The presentation takes 10 minutes, demo. 10 and questions/discussion 5.

2. Project Report

The final project report should include all the following artifacts covering the whole software development process. Include a cover page with the project title, names and e-mails of team members.

1. (2 pts) Problem definition. Also, include the type of problems you are solving and the motivations of your project (1 page).
2. (5 pts) Feasibility study in terms of benefits only (2 pages). Clearly show if you are developing a new system, extending or improving existing ones (include references of current solutions). What are the benefits of your application when compared to existing solutions?
3. (17 pts) Software requirements elicitation and specification:
   (a) (1pt) Functional requirements list (**only the ones that you have implemented**) for each user role. Explain each requirement briefly.

(b) (6 + 4 pts) Use case diagrams with all the use cases and actors. Describe in detail two use cases (the most complex ones and not common accross applications) with activity diagrams.

(c) (6 pts) Software qualities: correctness, time-efficiency, robustness and user friendliness. Explain in detail why each quality is necessary for your specific application.

4. (28 pts) Top-level and low-level design:

   (a) (3+4 pts) Logical software architecture. Decide which architectural style you will use for your application. Explain in detail why did you choose this style.

   (b) (8+8 pts) Design patterns. Explain in detail why did you select these patterns. Give the complete class diagram for each selected pattern. Also provide the algorithms corresponding to the pattern's important methods. For each class, provide the data types of the attributes and prototypes of the methods.

   (c) (5 pts) Class diagram by incorporating the design patterns.

5. (9 pts) Programs:

   (a) (3 pts) Component diagram regarding the organization of the source code.

   (b) (3 pts) Deployment diagram regarding the hardware configuration of the code. Indicate the supported Web browsers, the application/Web servers and the data base solution.

   (c) (2.5 pts) Screenshots of all table contents of the system data.

   (d) (0.5pt) Include a link of your Web-based application.

6. (4 pts) Technical documentation:

   (a) (1 pt) List of programming languages.

   (b) (1 pt) List of reused algorithms and programs. Include their exact sources.

   (c) (2.5 pts) List of software tools and environments. Provide briefly their benefits specifically for your application.

7. (15 pts) Acceptance testing:

   (a) Functional testing with five test cases (screenshots of inputs and outputs).

   (b) Robustness testing with five incorrect input data (screenshots of inputs and outputs).

   (c) Time-efficiency testing of five functions with screenshots.

3. Project Presentation

For the project presentation and software demo, please focus on the following points:

1. Problem and motivations.
2. List of the main functional requirements.
3. Design patterns. Explain precisely their usability for your specific application. **Show their class diagram and code**.
4. List of software development tools.
5. Class and deployment diagrams within a UML tool.

4. Important Notes

1. The two design patterns Facade and Singleton cannot be used as they are very simple.
2. All the UML diagrams must be legible and produced with a UML tool, such as starUML, ArgoUML, Visual Paradigm or Microsoft Visio. Please submit the screen-shots of the diagrams.
3. The code is constructed based on the specification and design documents. It should be distributed and contains the design patterns.
4. Submit the entire code (compressed) on UR Courses. Do not print the code!
5. Only the theam leader submits the required documents (report, slides and code).
6. Student attendance is mandatory for all the presentations.