Semester Project - 2024

IDATT2105 Full-stack application development

Read this document carefully and be sure you take the details into account!

The project assignment is voluntary and can only be carried out by students who receive a C, and want to improve their grade to A or B. Note that all parts of the project (programmed solution, documentation, and presentation) are expected to maintain **very** good quality to achieve grade B or A.

The requirement is to implement a full-stack web application including

- A frontend using Vuejs (v3) related framework and libraries
- A backend using Java (v17 or v21) and Spring boot framework (v3) / Spring framework (v5 (or v6))
- **A database using MySQL (v8) and/or H2**

The project contains several parts and the more the group does - with good quality - the better! **So, every group should assess what they prioritize and complete the chosen functionalities as much as possible. Unfinished functionality will not be evaluated and should not be part of the delivery!** The evaluation will be based on good code and necessary documentation for testing and further development, also user interface and well-implemented functionality are important in assessment.

Note that plagiarism is not allowed. If the source code of two or more groups is too similar, the grade of all those groups may be adversely affected and can result in being banned from the study!

Note also that the total number of students per group is from **2 to 4**, and the expected effort per student is about 40 hours.

More on general and product requirements are described in the following.

General Requirements

1. The submission deadline is Sunday **7th of April 2024, kl. 23:59.**
2. The required materials should be submitted in **Blackboard**
3. One submission per group. Remember to write the names of all participants in the group.
4. Technical requirements
    - 4.1. Clean CSS. No use of CSS frameworks like Tailwind
    - 4.2. Spring Boot/REST based backend. Each endpoint should have proper authentication/authorization, for example based on JWT and Spring Security.
    - 4.3. Database communication using Spring JDBC / JPA
    - 4.4. The solution should contain tests. The code coverage shall be at least 50%
    - 4.5. The group should make sure to use CI/CD during development.
    - 4.6. OWASP and universal design principles/techniques should be used in the project
    - 4.7. Session storage can be used to provide short-lived login session on the frontend
5. Documentations requirements (for each module)

5.1. End points (API) must be documented, for example, using Swagger. Note, an explanation of what the endpoints do and what the different attributes are is required. In addition, code must be documented as usual (javadoc).
5.2. System documentation is also a requirement., i.e., documentation that enables a new developer to quickly get the project up and running for testing and further development (architecture sketches/class diagram). Instructions for how to run the project can preferably be done as a README file, while other documentation should be as a PDF.
5.3. Test data that can be used while testing the app, for example, test user credentials, database credentials, etc.
5.4. The prerequisites must be documented if the project/module is dependent on other projects

6. Submission materials.
   6.1. A zip file including all the modules/files
   6.2. A runnable source code of each project/module. For example
      - Properly documented Source and Test files.
      - Configuration files like pom.xml (maven), package.json, Dockerfile, etc.
      - Flyway or normal DB schema scripts with test data

   6.3. Description of how to run tests and get the system running. This should be made easy (for the user) - as a script, docker/maven command or the like.

Product requirements / features

The product is a quiz bank like web application that provides a comprehensive and user-friendly platform for creating, managing and taking quizzes for educational, training, or entertainment purposes.

The layout and design of user-interface elements such as screens is open-ended and it´s up to each individual group's creativity. Students, however, can take inspiration from existing web applications such as Quizlet, Kahoot, Quizizz, Brainscape, and so on. UI screens should be designed such that they are mobile devices friendly in terms of size, pagination and scroll ability. In addition, the application shall have the following features.

1. User Authentication: Allow users to register, login, and manage their accounts securely.
2. Quiz Creation: Enable authenticated users to create quizzes by adding questions, specifying categories, setting difficulty levels, and adding multimedia elements such as images or videos. Optionally, AI based generation of questions can also be done based on some user input or requirements.
3. Tagging and Categorization: Allow users to tag questions with keywords and categorize them into topics or subjects for their easy organization and retrieval.
4. Search and Filter: Implement search and filter functionalities to help users quickly find relevant questions based on keywords, categories, difficulty levels, and other criteria.
5. Question Management: Provide tools for users to add, edit, delete, and organize questions within their quizzes. Support different question types such as multiple choice, true/false, fill in the blank, etc.

6. Quiz Templates: Offer pre-designed quiz templates or customizable templates to simplify the quiz creation process for users.
7. Collaboration: Enable collaboration features such as sharing quizzes with other users, inviting co-authors to collaborate on quiz creation, and providing commenting or revision history features.
8. Import and Export: Allow administrators to import questions from external sources such as CSV files or other quiz formats, and export quizzes in various formats for sharing or backup purposes.
9. Randomization: Provide options for randomizing question order and answer choices within quizzes to prevent cheating and enhance fairness.
10. Scoring and Feedback: Automatically score quizzes and provide immediate feedback to users upon completion. Optionally, allow users to review their answers and see explanations for correct solutions.
11. Progress Tracking: Enable users to track their progress over time, view past quiz attempts, and monitor performance statistics to identify areas for improvement.
12. Feedback and Support: Offer channels for users to provide feedback, report issues, and access customer support for assistance with using the web app.

Project presentation

The project presentation will be done in **week 15**. The detailed presentation plan for each group will be published on Blackboard. Each group will have 15-20 minutes to present their work. There will be 5-10 minutes of QA session after the presentation. More info will come closer to the presentation.

Questions about the assignment?

Use the forum on Blackboard. The idea, however, is that if the desired functionality is ambiguous, the group is largely free to interpret the task themselves. **The group should prioritize the most important functionality and finish that, before taking on more functionality. Therefore, make a list of your prioritizations and why.**