McGill

Department of Electrical and Computer Engineering
ECSE 321 Introduction to Software Engineering
Winter 2020

Project Deliverable 3
Web Frontend and Architecture Modeling

This deliverable consists of the following parts:

1 Architecture Modeling

You need to provide an **architecture model** (in the form of a block diagram or UML Composite Structure diagram) that highlights the main functional components of your system, their hierarchy and their interconnections with other components. You need to include the diagram/figure of the architecture in the wiki of your team together with a **brief description of the key functional components** (max 1 paragraph each).

2 Implementation of Web Frontend

You need to provide a **graphical user interface** for the web frontend using HTML, CSS, and JavaScript technology (preferably using Vue.js). In addition to functional correctness, usability and design of the web frontend will also be evaluated.

3 Integration of Web Frontend with Backend Services

You need to **integrate your web frontend** with your **existing backend services**. You need to issue asynchronous calls to backend services via the RESTful API developed in Sprint 2, and correctly process the results. You are required to update the backend services as needed – your backend will be evaluated also as part of Sprint 3.

4 Build System, Continuous Integration, and Delivery

An automated deployment process for the web frontend is required. Travis CI should verify that the dependencies of the frontend application can be acquired using *npm install*, then deploy a new version of your web frontend to Heroku. Note that this step requires to create a *new Heroku application without a database* (i.e. a second Heroku application in addition to the existing one used for the backend and database). Since no test cases are required (although allowed) to be developed for the web frontend, test cases of the frontend are not required to be integrated to Travis CI either.
Every push to the GitHub repository should trigger a build job on Travis CI to initiate continuous integration and delivery for both the **backend** and the **web frontend**. The build system and the CI specification must conform to the Technological Constraints.

5 Project Management and Project Report

A key aspect of agile software development is the use of a project backlog to coordinate development and project documentation activity. Each group is expected to **continuously use** the **issue tracking** features on GitHub and an **agile project board** in order to create and manage the project backlog. Each issue needs to have an assignee to trace core responsibilities within the team.

The team should provide a welcome page that introduces the team and describes the main scope of the project in the README.md file in the root of each team's repository. In addition, the README.md file should contain an overview table with names, team roles, and individual efforts (in hours) with separated entries for each deliverable.

Project Deliverable 3 shall be also accompanied with a succinct *project report* as *part of the project wiki* which records the meeting minutes and the key design decisions taken by the team. This project report should be navigable from the README.md file of the project. Altogether, the team should comply with all Technological Constraints mentioned in the project description.

Submission

The project will be carried out with the same teams as you worked with for Deliverables 1 and 2. For Deliverable 3, your team is required to submit a commit link (i.e. a URL representing your last GitHub commit that counts as a deliverable) by ***Sunday, March 22, 2020 11:59pm***.

Each team member must make contributions to the deliverable. A team member who does not contribute to the deliverable receives a mark of 0 for the deliverable. A team member may optionally email a confidential statement of work to the instructor before the due date of the deliverable. A statement of work first lists in point form the parts of the deliverable to which the team member contributed. In addition, the statement of work also describes whether the workload was distributed fairly evenly among the team members. A statement of work may be used to adjust the mark of a team member who is not contributing sufficiently to the deliverable. It is not necessary to send a statement of work, if a team distributed the work for the deliverable fairly evenly and each team member contributed sufficiently.

Marking Scheme (10% of total score)

Component	***Points***
Updates of backend Implementation (business methods, RESTful services, unit tests)	5
Architecture model	10
Implementation of web frontend GUI (incl. design + usability)	25
Integration of web frontend with backend services (incl. asynchronous backend calls of RESTful API)	25
Build system + Continuous integration (npm build, Travis CI, Heroku for the web frontend; Gradle + Travis CI + Heroku for the backend)	20
Project management (e.g. backlog, issues, sprint planning, teamwork report, documentation quality)	15
Total Marks:	100

<u>Note:</u> The total mark may be adjusted based on the actual contributions of a team member to the deliverable.