COMP 3100 – Web Programming

Project - Iteration 1 (P2)

Winter 2022

This deliverable is due on the March 18th, 2022 at 10:00 PM Newfoundland time. No submissions done outside D2L will be marked (e.g., email). Please organize yourself with your team to submit the document on time. Grade deductions for late submissions will be applied, check the syllabus for the detailed grade decrease.

The main goal of the second iteration is to evaluate your server-side code. In the first iteration, your team has submitted your project's overall idea and goals with some functionalities that your system would like to provide. Now it's time to code it!

Your team must use an MVC-like framework, as we have seen in the lectures. The project should have the 'models' with classes and objects that must be manipulated on the server-side. All create, read, update and delete (CRUD) functions for those objects must be created and documented at this stage. Following the MVC pattern, your project code must have the 'controller' folder that will manipulate your application's business logic. All projects must use a database, most of the teams will use MongoDB, but I approved some other options depending on the project's final goal. Finally, you must assemble an API using express to provide all your functionalities over HTTP requests. The last part of this deliverable is your server-side tests using Mocha.

Your project will be evaluated into two parts as follows:

Project document (40 marks)

1. The team must provide a document that is an update of the former, documenting what was done to implement the server-side of the application. What functionalities were implemented? What were added that were not planned? What functionalities were removed? Such information should be provided in this document.
2. The document must have a section named Models where you document your models and what you can do with them (methods). Why are they necessary for your application?
3. The document must have a section named Routes and Controllers that should report the paths used in your API. What each one of them does? How to use them? (Short code-snippets on how to access these paths are welcome in your report)
4. You should show how you designed your Data Model collection(s)? Which pattern is it following (embedded or normalized)? Why you chose this particular pattern?
5. The document must have a section named Tests documenting the tests performed. Which tools were used? Which tests were performed? Why? How complete are your tests? Did they cover most success and failure cases?

Project code (60 marks)

1. How much of the functionalities you declared in the first iteration are you covering? Is it complete to start assembling the client-side? Is there anything missing?
2. Your code should follow the structure detailed in the beginning of this document. Is it following the MVC pattern with the models, controllers and routes properly?
3. Is your data being saved properly in the database? How well designed is(are) your collection(s)?
4. Are your tests complete, with several success and failing cases? Are you testing your models and your full requests properly? Are they covering success and failing cases? Are you doing simple and complex use cases for testing?

5. How well-documented are your functions and codes? Are they easily readable?

Your grade will be determined as follows:

Criteria	**Level 4**	**Level 3**	**Level 2**	**Level 1**	**Score**
Project document (40 marks)					
Functionalities documentation	8 points Followed all the guidelines	6 points Followed most of the guidelines	4 points Followed some parts of the guidelines	2 points Followed none of the guidelines	MAX 8
Models section	8 points Models' description is clear	6 points Models' description is presented, with minor points to be clarified	4 points Models are presented, but major points are not clear (need further clarification)	2 points There was none or poor description of the Models' (it is too vague)	MAX 8
Routes and controllers	8 points Routes and controllers' description are clear	6 points Routes and controllers are presented, but there are things that need to be clarified	4 points Routes and controllers are presented, but major points are not clear (need further clarification)	2 points There was none or poor description of Routes and controllers' (it is too vague)	MAX 8
Data Model	8 points Data Model description is clear	6 points Data Model is presented, but there are things that need to be clarified	4 points Data Model is presented, but major points are not clear (need further clarification)	2 points There was none or poor description of Data Model (it is too vague)	MAX 8
Tests	8 points Tests description is clear	6 points Tests description is presented, but there are things that need to be clarified	4 points Tests description is presented, but major points are not clear (need further clarification)	2 points There was none or poor description of Tests (it is too vague)	MAX 8
Project Code (60 Marks)					
Functionalities coverage	12 points All functionalities are covered and are ready for assembling the client-side	9 points Most functionalities are covered and are ready for assembling the client-side	6 points Some functionalities are covered and are ready for assembling the client-side	3 points Few functionalities are covered and are ready for assembling the client-side	MAX 12
MVC structure	12 points MVC structure was followed to its full extent	9 points MVC structure was followed, but some parts are not following it	6 points MVC structure was not followed in full, but some	3 points MVC structure was not followed	MAX 12

			parts are are following it		
Database	12 points All data is being saved properly and the model design is being followed properly	9 points Data is being saved properly but the model design is not being followed to its full extent	6 points Some data is being saved properly and model design is not being followed to its full extent	3 points None of the data is being saved and the model design is not matching what was proposed	MAX 12
Tests	12 points All tests are fully implemented and they cover all success and failing cases	9 points Most tests are fully implemented and they cover most success and failing cases	6 points Some tests are fully implemented and they cover some success and failing cases	3 points Few or none of the tests are implemented and they cover few success and failing cases	MAX 12
Code documentation	12 points Documentation is well-done and is easily readable	9 points Documentation is done and is readable	6 points Documentation is somewhat done and is difficult to read	3 points Documentation is not done and cannot be read	MAX 12