|  |  |
| --- | --- |
|  | Luiz Lacerda Ferreira, 28 Feb at 13:56 |
| Your "BackEnd Stage (5)" image indicates that you have used Node.js for the backend, despite the assignment brief stating that you must use a PHP framework. You should use the remaining time to review your submission and ensure it meets the assessment requirements. | , 29 Feb at 10:33 |
| Luiz Ferreira - Assessment - Portfolio | Luiz Lacerda Ferreira, 21 Mar at 10:11 |
| Your submission would indicate that you did not read and/or understand the assignment brief. You need to ensure you read the assignment brief and do the work it requires. | , 5 Apr at 14:01 |
| -The assignment brief requires you to use a PHP Framework such as Laravel for the back-end and either Angular or React for the front-end. -The web application presented contains no functionality other than a single button click -The report does not address the required criteria |  |

**Assessment**- Portfolio

The assessment comprises of a portfolio comprising two tasks:

* Task 1: Build building a web application (50%)
* Task 2: A reflective report (50%)

Task 1

* Learning outcomes addressed: MO1, MO3
* Weighting: 50%

You are required to build a Web Application on a topic of your choice. Pick something you have an active interest in, a hobby, a sport you play, a group or club you belong to, or any similar topic. Once you have selected your topic YOU MUST CONFIRM THIS WITH YOUR TUTOR. You will not be allowed to proceed with your topic of choice without tutor agreement. You may not change your topic without further consultation with your tutor.

The following features and technologies should be implemented:

* **Client-side framework** – The client part of the application should be implemented using a JavaScript framework such as React or Google Angular.
* **Server-side API** – Your client-side application should connect to a server-side API to obtain and save data to a persistent database.
* All source code should be committed to a **Git repository** on a regular basis.

Task 2

* Learning outcomes addressed: MO1, MO2
* Weighting: 50%
* Word Limit: 3000 words

You should write a report covering the following topics:

* An explanation of the **automated testing** that you have performed on your application and a reflection on the outcomes.
* An explanation of the **performance improvements** you have implemented following the use of Google Lighthouse.
* **A discussion around the techniques and technologies used** should be included. You should make appropriate use of literature, refer back to the experience acquired when you built your application (see task 1) and show a good understanding of the various topics taught on the module.

Rubric

**Portfolio**

| Portfolio | | |
| --- | --- | --- |
| **Criteria** | **Ratings** | **Pts** |
| This criterion is linked to a learning outcomeWeb application 1  Client-side framework | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **15 Pts**  **Full marks**  A client-side framework was used successfully, including advanced features. The application is fully functional, all requirements were implemented. | **12 Pts**  **Good effort**  A client-side framework was used successfully. The application is fully functional, most requirements were implemented. | **10 Pts**  **Competent**  A client-side framework was used, very few issues. The application is mostly functional, basic requirements were implemented. | **8 Pts**  **Pass**  A client-side framework was used but a lot of issues with implementation. The application is barely functional and at prototype stage. | **4 Pts**  **Insufficient**  A client-side framework was not used. The application is not functional. | **0 Pts**  **No marks**  No or very little work done | | 15 pts |
| This criterion is linked to a learning outcomeWeb application 2  Server-side RESTful API | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **15 Pts**  **Full marks**  Persistent storage of data via server-side RESTful API, with advanced features (authentication, security, error handling) Full CRUD functionality. A framework has been used to implement the API. | **12 Pts**  **Good effort**  Persistent storage of data via server-side RESTful API, standard features, fully functional 3 CRUD aspects were implemented. A framework has been used to implement the API, but with minor issues. | **10 Pts**  **Competent**  Persistent storage of data via server-side RESTful API, standard features, issues with implementation. 2 CRUD aspects were implemented. No use of a server-side framework, but code is structured according to an architectural pattern. | **8 Pts**  **Pass**  Persistent storage is local and/or hard-coded. Data is read-only, no ability to update, add or delete. No use of a server-side framework, code is mostly unstructured. | **4 Pts**  **Insufficient**  Persistent storage is not functional. Server-side API is not functional. | **0 Pts**  **No marks** | | 15 pts |
| This criterion is linked to a learning outcomeWeb application 3  Layout and navigation | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **10 Pts**  **Full marks**  Layout and navigation are responsive, clear and consistent. | **8 Pts**  **Good effort**  Layout and navigation are clear and consistent but not responsive. | **7 Pts**  **Competent**  Minor issues with layout and/or navigation. | **5 Pts**  **Pass**  Major issues with layout and/or navigation. | **2 Pts**  **Insufficient**  Navigation and/or layout not functional. | **0 Pts**  **No marks**  No or very little work done | | 10 pts |
| This criterion is linked to a learning outcomeWeb application 4  Git repository | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **10 Pts**  **Full marks**  All code committed to Git on a regular basis, use of issues, labels, and milestones. | **8 Pts**  **Good effort**  All code committed to Git on a regular basis. | **7 Pts**  **Competent**  All code committed to GitLab, but intermittently. | **5 Pts**  **Pass**  Some code committed to GitLab, intermittently. | **2 Pts**  **Insufficient**  Very little use of GitLab. | **0 Pts**  **No marks**  Git was not used at all. | | 10 pts |
| This criterion is linked to a learning outcomeReport 1  Testing | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **25 Pts**  **Full marks**  Excellent testing using appropriate tools. Results are clearly presented. | **22 Pts**  **Good effort**  Excellent testing using appropriate tools. Some omissions. | **18 Pts**  **Competent**  Adequate testing using appropriate tools. Omissions or lack of clarity. | **13 Pts**  **Pass**  Appropriate tools were used, but very basic testing. | **8 Pts**  **Insufficient**  Very poor, very little evidence. | **0 Pts**  **No marks** | | 25 pts |
| This criterion is linked to a learning outcomeReport 1  Analysis | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **25 Pts**  **Full marks**  Excellent analysis and comparison of all techniques and technologies used. Well referenced literature. | **22 Pts**  **Good effort**  Excellent analysis and comparison of most techniques and technologies used. Well referenced literature. | **18 Pts**  **Competent**  Excellent analysis and comparison of some techniques and technologies used. Well referenced literature. | **13 Pts**  **Pass**  Very basic analysis, discussion is simplistic, very little literature used. | **8 Pts**  **Insufficient**  Very poor, no understanding of the different techniques and technologies, no literature used. | **0 Pts**  **No marks** | | 25 pts |
| Total points: 100 | | |

[Previous](https://canvas.wlv.ac.uk/courses/44066/modules/items/1758050)