Final Deliverable – Software Project

Final assessment – 70% of your mark for this module

As part of this assignment your group should submit all of your deliverables (software + version control logs + report.) This is a chance to reflect on the process in a considered and holistic way and to produce a robust report describing both your process and your deliverable components. How you choose to structure your report is up to you, but as a general set of guidelines you might want to include the following headings.

Background

Here you should detail your agenda including your aims, literature and sources of information and the scope of your project. This should set the scene for the work that follows including ideation steps and a brief summary of your group and process. This is an opportunity to reflect on your initial design that you produced for coursework 1 and identify a clear set of deliverables based on the work that you have previously done.

Planning and Research

This section should focus on evidencing the research that you have undertaken in order to substantiate the project. This could include a literature review, some visual evidence of milestones and process (e.g. Gantt charts) and any evidence of planning and iteration. It should be clear to the reader how and why you have settled on certain approaches for your project and how resources and time are allocated in relation to the functionality of your system. There should be a detailed breakdown of your plan in terms of a working process and set of expectations for deliverables (e.g. dates, times, functionality etc.) This is not the same as your planning for your project proposal, though you should be reflecting on the work you pursued therein.

Prototyping and Iteration

Should show evidence of iteration from a concept to a proof-of-concept. There should be a clear rhetoric, justification for design decisions and a process of systematic evaluation that results in clear improvements to the system design over time and iteration. Equally, functional prototypes should evidence the development of code over time including testing of input/output. Functionality should be justified through an analytical approach to development e.g. “we added this feature because…” This would ideally be clear from the report without making reference to other aspects of the project such as code and version control logs.

Design

This section should focus on how you have come about your design specifications through prototyping and evaluation. You should (re)detail the specifications of your system at this point with a clear view about what it is you are building and the approaches that you intend to take. It should be clear what you are intending to build and how you intend to go about producing such a build in the time frame given. This should also include technical design elements such as the tech stack you have chosen and any dependencies or libraries used.

System Development

There should be a clear and concise description of the development of code, assets and resources that make up the **core** components of the system. You should reflect on your engagement with agile and user-centred design processes. This should include any steps or stages in your development process and testing. You can make reference to your central git repository here, though you might want to include snippets of code and/or processes to better describe how outcomes were achieved. This should show a clear breakdown of how you turned backlogs, feature requests and use cases into working prototypes. You should also describe the processes that you engaged in e.g. test-driven development, agile techniques and how they enabled the development and iterative, incremental changes that contributed towards the success/failure of your projects. If you choose not to explore agile techniques then you should explain why they were not fit for purpose and how the chosen development approach is superior.

Analysis

You should produce a comprehensive analysis of your solution(s) relating to a wide range of resources including literature.

Evaluation

You should evaluate your software development as both a set of processes (ie team working, methodologies employed) and the thing that you have built. Higher marks will be awarded where the evaluation is critical in nature, alluding to future work and highlighting areas for potential improvement.

Conclusion/Summary

You should summarise the outcomes of your project and explain the achievements and outcomes of the project. Here you might want to discuss where it fits in a dynamic workflow or the impact that it might have on a community of stakeholders or users.

Individual reflection

You may also include an individual reflection that describes your role in the project, how you worked in a team and the challenges faced as a result of the ways of working.

For this second submission, you should produce a document of no more than 10,000 words (not including appendix.) The marking criteria for your final assessed component 70% or 70 marks of the course is as follows.

Technical	**Total Marks available**
Code solves challenges presented in the aims/objectives	4
Iterative design	4
Approach is fit for purpose without drastic oversights	4
Elegance/aesthetics/readability of code	2
Sensible approach to structure (e.g. index.html, style.css, functionality.js)	2
Evidence of collaboration and team work. Individual submissions should show engagement with a wide range of external resources and utilisation of tools such as peer review and discussion.	4
Evidence of milestones and reflective updates (e.g. updated Gantt charts, resource allocation.)	2
Difficulty of technical challenge	4
Novelty of technical challenge	4
	30
Testing	
All parts of the system working	2
UI evaluation	1
Well documented tests	1
Effective error handling	1
Systematic testing regime including design of appropriate test cases	1
Justification of testing methods	1
User evaluation involving representative stakeholders formative	2
User evaluation involving representative stakeholders summative	1
	10
Report	
Good introduction and fair discussion of literature	2
Clear statement of problem and effective problem analysis	2
Clear structure	1
Justification of design decisions	2
Good argumentation and justification of claims/problem analysis	2
Clear documentation and user guide	1
Sufficient and appropriate references, and good citing method	2
Good layout and formatting, especially of tables, figures, formulae and code example	2
Well written with systematic analysis/evaluation	4
Insightful discussion of results	4
Evaluation of own work in relation to original proposal and plan	4
Conclusion and discussion of future work	4
	30