

Innovate: Coding coursework  
Student Name: Iain McLeod

This document details the written coursework to be completed alongside your projects or as a retrospective for the Code Nation Innovate: Coding course.

Learners studying at level 3 are expected to evidence excellent English skills and complete at least the equivalent of two paragraphs (over 100 words) per question.

You can use experience of the any of the project lifecycles or projects completed during this course to answer these questions, as well as any research where appropriate (please include any sources you use).

**Collaborative technologies**

**1. Discuss how different collaborative technologies can be used and integrated, and explore any potential compatibility issues. (Describe any potential access issues from tools or devices, and how to overcome them).**

The collaborative technologies for use in all project types and working environments, and not just code development, are things like live video/audio apps, Zoom etc. Typed messaging systems like, Slack, Discord, email, IRC, instant messaging. Graphical systems like whiteboards. And VoIP and the traditional telephone.

For programming projects, version control systems like Git are essential.

Comments in the codebase, detailed meaningful ones, and not just Javadoc type comments, are also important so that you can see what additions and changes are being made and what they supposed to do. This clashes with the 'the code is its own comments' philosophy, because that requires reading the code, which takes more time and effort than reading plain English, and can also mislead if the code contains mistakes.

Things like 'live share' in Visual Studio 2022 etc. can be used, which is live collaborative editing on one machine by several people, used in conjunction with live VoIP or Zoom.

For compatibility issues, everybody should run the same versions, which should be the norm in team working.

**2. Identify risks associated with using collaborative technologies, and describe how you can prevent and manage them. (How can a developer ensure the security of a project for both client and server side?)**

Risks: Leaks of your IP due to bugs, hackers attacking the backend or frontend, untrustworthy collaborators or impersonators.

GitHub Copilot lifting whole sections of open-source projects without the required attribution.

Burning time servicing the collaboration technology and its bugs, rather than doing actual work.

The constant interruptions, 'have you got five minutes' etc., where those five minutes wreck your concentration for the following twenty. This is already a problem in office environments, and especially for programming where you have to get 'into the zone' and hold multiple thoughts in your head at the same time so that those ideas can interact.

Floods of email and messages that have to be answered, again eating into development time. This is something often complained about by office workers.

**3. Discuss the effectiveness of collaborative technologies used in this project (Describe the features, benefits, limitations and potential issues of using these tools and how this affected your project).**

We used Zoom, Slack, and GitHub, but mostly one way, to keep the course host up to date on our work. We also used Zoom and Slack to ask for help and assistance.

We didn't really use these tools in the way a collaborating team of peers would use them, since our projects were individual efforts.

**Project development, testing and management**

**4. Explain the benefits and drawbacks of using your chosen development environment and file management. (How would you ensure an efficient development environment for a future project? Reflect on any different principles, tools or technologies you could use).**

VSCode is good and free and can handle many languages, as is Visual Studio 2022, which I use for C/C++. For once it installed easily without a myriad of obscure problems that used time scouring Stack Exchange researching and fixing them.

Git is fairly essential even in one person projects, for tracking changes. I also use 7zip archives for backup on my own projects.

The file management used is what the particular technology used requires.

**5. Describe why a developer needs to test their projects, and explain potential methods of testing through the lifecycle of development project. (Explain the effectiveness of testing and discuss any potential boundaries or corners when it comes to testing).**

Because if you don't test you can guarantee bugs and incorrect behavior.

On my own projects I mostly use non-automated unit-like testing, throwing various things at new functions whilst running in a debugger, to see they are actually doing what I think they are doing, and handling all the problems I have thought of, and if I have made any typos.

**6. Describe project management techniques that you have used throughout this project and explain any specific methodologies you have utilised. (Explain the effectiveness and efficiency of these techniques or methods, and how you would adapt to improve for future projects).**

A to-do list, and that is basically it. It is a single person project, so there is no collaboration involved, other than talking to my 'line manager'/ course host.

Trello serves a useful purpose for team work, so everybody can see the state of things quickly, what jobs have been completed and which still need doing, and the priority of what should be done next. This probably works best with somebody assigned the role of project manager.

In a single person project, a to-do list in notepad works just as well. The ability to move and re-order task cards in their lists is slightly easier than moving lines around in a to-do text file, but only marginally.

We did not cover Kanban, or use it, it is one of the 'agile' systems. Personally, I like the waterfall model, where you at least partly plan ahead so you have a good codebase upon which to build. Agile can tend towards 'making it up as you go along' if you are not careful. Single person projects tend to be a bit of both as you find things that are not going to work as planned, or you think of new features to add.

Since this is a very short project it was a case of mostly 'try it see if it works'. We didn't have the use of a single-stepping debugger, which give a much better view of code as it is running.

|  |  |  |  |
| --- | --- | --- | --- |
| Level 3 Unit Tracker | | | |
| Q1 | 3.2.3, 3.2.4 | Q4 | 3.3.1, 33.1.1, 33.1.2, 33.1.3 |
| Q2 | 3.1.1, 3.1.3, 3.1.5, 3.1.6 | Q5 | 33.1.4, 33.3.4 |
| Q3 | 3.2.1, 3.4.5 | Q6 | 33.2.1, 33.2.4, 33.3.1, 33.3.2 |
| FOR INTERNAL REFERENCE ONLY | | | |