

## *Assessment: Emerging Technologies*

*ian.mcloughlin@atu.ie*

*Autumn 22/23*

These are the instructions for the assessment of Emerging Technologies in Autumn 22/23. These cover the full 100% of the assessment for this module.

### *Submission*

- The deadline for submission is August 31<sup>st</sup>, 2023.
- Your whole submission must be in a single GitHub repository.
- Use the form on the Moodle page to submit your repository.
- Commits in GitHub on or before the deadline will be considered.<sup>1</sup>

<sup>1</sup> Once you have submitted your URL, you do not need to do anything other than commit to your repository and push the changes to GitHub.

### *What to submit*

This assessment has three overlapping components, as follows.

#### GITHUB REPOSITORY (20%):

- Create a single GitHub repository for all your work.
- Describe in a README the contents of the repository and how to run the files within it.
- Include an appropriate `.gitignore` file.
- Ensure regular and appropriate commits are made to the repository throughout your work on the project.
- Ensure filenames are clear and no necessary files are added.

#### QISKIT NOTEBOOK (40%):

- Add a notebook to your repository named `qiskit.ipynb`.
- In the notebook, explain what the `Qiskit` package is used for.
- Demonstrate the basic usage of `Qiskit`.
- Use visualizations to explain the core concepts of `Qiskit`.

#### DEUTSCH ALGORITHM NOTEBOOK (40%):

- Add a notebook to your repository named `deutsch.ipynb`.
- In the notebook, explain the Deutsch algorithm<sup>2</sup>.
- Clearly define and explain the problem, and include an explanation of how to solve it on a classical computer.
- Simulate the solution of the problem on a quantum computer using `Qiskit`.

<sup>2</sup> The Deutsch algorithm is the one qubit version of the Deutsch-Jozsa algorithm.

## Marking Scheme

Each component will be marked using the four categories below. To receive a good mark in a category, your submission needs to provide evidence of meeting each of the criteria listed under it<sup>3</sup>.

*Research (25%)*: evidence of research on topics; appropriate referencing; building on work of others; comparison to similar work.

*Development (25%)*: clear, concise, and correct code; appropriate tests; demonstrable knowledge of different approaches and algorithms; clean architecture.

*Documentation (25%)*: clear explanations of concepts in notebooks; concise comments in code and elsewhere; appropriate, standard README for a GitHub repository.

*Consistency (25%)*: tens of commits, each representing a reasonable amount of work; literature, documentation, and code evidencing work on the assessment; evidence of reviewing and refactoring.

<sup>3</sup> In line with ATU policy, the examiners' overall impression of the submission may affect individual marks in each category.

## Advice

- Students sometimes struggle with the freedom given in an open-style assessment.
- You must decide where and how to start, what is relevant content for your submission, how much is enough, and how to make the submission your own.
- This is by design — we assume you have a reasonable knowledge of programming and an ability to source your own information.
- Companies tell us they want graduates who can (within reason) take initiative, work independently, source information, and make design decisions without needing to ask for help.
- The point of this assessment is to demonstrate you can do that.
- You need a plan, you cannot just start coding straight away.

## Policies

- You are bound by all ATU policies and any GMIT policies that have not yet been replaced by new ATU policies.
- Review the GMIT Quality Assurance Framework <sup>4</sup>.
- Pay particular attention to the Policy on Plagiarism and the Code of Student Conduct.
- If you have any doubts about what is permissible, email me to ask<sup>5</sup>.

<sup>4</sup> GMIT. Quality Assurance Framework.  
<https://www.gmit.ie/general/quality-assurance-framework>

<sup>5</sup> [ian.mcloughlin@atu.ie](mailto:ian.mcloughlin@atu.ie)