CA2 – AES File Encryption

|  |  |  |
| --- | --- | --- |
| **Grade** | **Criteria / Characteristics** | **Mark** |
| **Exceptional**  80%-100% | - **All Requirements** and supporting functionality fully implemented at a high degree of difficulty and to a high quality, has met all feature deadlines, **additional features incorporated**, and  - **verifiable evidence** of incremental development of the project based on commit/push history from shared Repo and in-class discussions with lecturer, consistent pattern of rigorous and well documented commits, high quality commit messages **and**  - work presented/discussed/interviews demonstrate **thorough** understanding of all concepts, algorithms, principles and patterns, structures/idioms used in the presented solution, **and**  - **highest quality code** delivered [inter-alia; readability, variable naming, DRY principles, use of patterns, efficiency, correct and most appropriate data types, quality comments] **and**  - easy to use user interface, **and**  - comprehensive application of relevant testing techniques and tools, **and**  - all other project specification requirements and deliverables met to the highest standard. |  |
| **Excellent**  70% - 79% | - **All Requirements** and supporting functionality fully implemented at a high degree of difficulty and to a high quality, has met all milestones, **and**  - **verifiable evidence** of incremental development of the project based on commit/push history from shared Repo and in-class discussions with lecturer, at least **two** commits for each feature with good comments, and  - work presented/discussed/interviews demonstrate **excellent** understanding of all concepts, algorithms, principles and patterns, structures/idioms used in the presented solution, **and**  - **high quality** code delivered [inter-alia; readability, variable naming, DRY principles applied, use of patterns, efficiency, correct and most appropriate data types, quality comments] **and**  - easy to use user interface, **and**  - **main functionality tested** using relevant testing techniques and tools, **and**  - all other project specification requirements and deliverables met to a **very high** standard. |  |
| **Very Good**  60%-70% | - **Most Requirements** (~70%)\* and supporting functionality fully implemented at a medium degree of difficulty and to a good quality, has met majority of milestones, **and**  - **verifiable evidence** of incremental development based on commit/push history from shared Repo and in-class discussions with lecturer, at least ONE commit for each feature with comments, **and**  - work presented and interviews demonstrate good understanding of most concepts, algorithms, principles and patterns, structures/idioms used in the presented solution, **and**  - **generally good quality** code delivered, with possibly some weak/omitted aspects [inter-alia; readability, variable naming, DRY principles applied, use of patterns, efficiency, correct and most appropriate data types, quality comments, maintainability], **and**  - some consideration of user interface design, **and**  - **some functionality tested** using relevant testing techniques and tools, **and**  - all other project specification requirements and deliverables met to **a high** standard. |  |
| **Good**  50% - 59% | - **Majority of Requirements** (>50%) and supporting functionality fully implemented at a basic degree of difficulty and to a medium quality, **and**  - **verifiable evidence** of incremental development based on commit/push history from shared Repo and in-class discussions with lecturer, at least ONE commit for each feature completed with comments, **and**  - work presented and interviews demonstrate **basic** understanding of most concepts, algorithms, principles and patterns, structures/idioms used in the presented solution, **and**  - **generally medium quality code** delivered, with possibly some weak/omitted aspects [inter-alia; readability, variable naming, DRY principles applied, use of patterns, efficiency, correct and appropriate data types, quality comments, maintainability], **and**  - some consideration of user interface design, **and**  - **some functionality tested** using relevant testing techniques and tools, **and**  - all other project specification requirements and deliverables met to **basic** standard. |  |
| **Pass**  40%-49% | - **Few Requirements** (>40%) and supporting functionality fully implemented at a basic degree of difficulty and to an acceptable quality, **and**  - **verifiable evidence** of incremental development of the project based on commit/push history from shared Repo & discussions with lecturer, at least ONE commit for each feature with some comments, **and**  - work presented and interviews demonstrate **basic** understanding of most concepts, algorithms, principles and patterns, structures/idioms used in the presented solution, **and**  - **generally medium quality** code delivered, with possibly some weak/omitted aspects [inter-alia; readability, variable naming, DRY principles applied, use of patterns, efficiency, correct and appropriate data types, quality comments, maintainability], **and**  - **some functionality tested** using relevant testing techniques and tools, **and**  - some of the other project specification requirements and deliverables met to **basic** standard. |  |
| **Fail**  <40% | **Any of the following:**  **- Less than 40% Features implemented OR**  **- poor/no understanding** of concepts, structures used in the presented solution under interview OR  - unable to explain own code (consider possibility of plagiarism)  **- no commit history made available** |  |

\*(relative degree of difficulty of features will be taken into consideration); DRY (Don’t Repeat Yourself)

Students will normally be interviewed to assess their understanding of work submitted.