**Software Development Lifecycles (Advocate: Thiago Viana)**

**P1 Describe two iterative and two sequential software lifecycle models.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#2-sdlc-methodologies> |
| The evidence is situated under the heading “SDLC Methodologies” in the README.md for the GitHub repository (SDLC Methodologies).  This evidence is applicable as it covers iterative and sequential software life cycle models, how each model works, advantages, disadvantages and examples of what software may benefit from each specific lifecycle. |

**P2 Explain how risk is managed in the Spiral lifecycle model.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#spiral-lifecycle-model-risk-management> |
| The repository "SDCL-Methodologies" holds all the evidence relating to the above criteria and is applicable as it explains in brief how risk is managed in the spiral lifecycle model. |

**P3 Explain the purpose of a feasibility report.**

|  |
| --- |
| Link: <https://github.com/matthewsides/Feasibility-Reports#purpose> |
| The evidence is provided in the repository "Feasibility Reports", and is applicable as it defines feasibility reports, explains the components and the purpose which is in line with the criteria. |

**P4 Describe how technical solutions can be compared.**

|  |
| --- |
| Link: <https://github.com/matthewsides/Glossary-Of-Tech-Terms/blob/master/README.md#9-technical-solutions-comparision>  <https://github.com/matthewsides/ZSL-Green-House-Menace#194-ide> |
| The primary evidence for this criteria may be found in the “glossary of tech terms” repository and readme.md, under the heading technical solution comparison. This constitutes as evidence as it delves into effective analysis which is essentially what is (a methodology) used in the comparison of technical solutions (how they are compared).  Further or secondary evidence, in this instance showing use in projects rather than being generalised, can be found under the "Specification" heading and sub heading "IDE", delving into the IDE, used and the factors what lead to choosing said IDE as well as the decision to use the language that was applied during the ZSL project explaining all the factors that were and should be considered during the choice of IDE, Language, etc. This links to the criteria as it shows evidence of technical solutions being compared and also how they may be compared and what to (requirements). |

**P5 Undertake a software investigation to meet a business need.**

|  |
| --- |
| Link: <https://github.com/matthewsides/ZSL-Green-House-Menace> |
| This evidence is documented in the ZSL repository, as the project backlog details information as to when the client was met for the review of the project evidencing development, whilst documentation of meetings was also noted in the Gantt chart. Moreover, the requirements or specifications can be seen under the subheading "specifications" as well as in the heading "project backlog" that details as to what the client is looking for as well as the conceptual design of the game. |

**P6 Use appropriate software analysis tools/techniques to carry out a software investigation and create supporting documentation.**

|  |
| --- |
| Link: <https://github.com/matthewsides/ZSL-Green-House-Menace#33-tools-and-techniques>  <https://github.com/matthewsides/ZSL-Green-House-Menace#194-ide> |
| This is documented under the "Evaluation" heading and sub heading "Tools and Techniques" as it mentions the tools and techniques used to carry out a software investigation, the IDE, Graphical tools and other software and why they were used in this instance.  Whilst through the second link, details of the IDE and language used are shown, thus further meeting criteria as it shows the decision and reason behind using said software and also the use of said software. |

**P7 Explain how user and software requirements have been addressed.**

|  |
| --- |
| Links: <https://github.com/matthewsides/ZSL-Green-House-Menace#32-how-user-and-software-requirements-have-been-addressed-generalised>  <https://github.com/matthewsides/Project01-TraceBall/blob/master/README.md#10-user-stories> |
| The evidence for this criteria may be found under the heading “evaluation” and subheading “How user and software requirements have been addressed (Generalised)”, meeting the criteria as a generalised overview is shown as to how user and software requirements have and maybe met further showing imagery and stating an example pertaining to the ZSL project partaken.  Evidence of meeting the criteria or at least examples are linked in the project01 documentation, meeting the criteria as the requirements, epics and user stories are all covered and addressed in the documentation.  Further alliterating, the process of addressing user and software requirements consisted of attaining a brief with information detailing a project, there after breaking the project down into Epics, which in turn were condensed into user stories, leading to and then covering the high functional and non-functional requirements of the project application. |

**M1 Describe, with an example, why a particular lifecycle model is selected for a development environment.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies> |
| The evidence can be found in the "SDCL-Methodologies", which covers and goes into in-depth research one each lifecycle, noting the advantages, disadvantages, how it works and examples. |

**M2 Discuss the components of a feasibility report.**

|  |
| --- |
| Link: <https://github.com/matthewsides/Feasibility-Reports#feasibility-report-components> |
| The evidence can be found in the repository "Feasibility-Reports" on GitHub, whilst the evidence in the README.md is suitable to the criteria as it explains the purpose linking into the other criteria mapped above, as well as the components and lastly impact. |

**M3 Analyse how software requirements can be traced throughout the software lifecycle.**

|  |
| --- |
| <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#requirements-traceability> |
| The evidence for this criteria may be seen in the SDCL Methodologies repository under the heading “requirements traceability” and sub heading “how software requirements may be traced throughout the software lifecycle”.  The evidence provided is applicable as it defines requirement traceability also detailing how the practices are done and essentially states how software requirements can be traced throughout the software life cycle (stating similarities in the structure throughout all projects that may be undertaken). |

**M4 Discuss two approaches to improving software quality.**

|  |
| --- |
| Link: <https://github.com/matthewsides/CMM/blob/master/README.md> |
| This constitutes as evidence as the README.md in the repository "CMM" defines CMM and pertains to the levels and also two approaches that may be taken using CMM that may improve software quality. |

**M5 Suggest two software behavioural specification methods and illustrate their use with an example.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#4-behavioural--specification-methods> |
| The evidence for this criteria can be found in the SDLC repository under the heading “behavioural specification methods”. The evidence provided meets the criteria since two software behavioural specification methods have been explained with examples illustrating their use. |

**M6 Differentiate between a finite state machine (FSM) and an extended- FSM, providing an application for both.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#3-fsm-and-efsm> |
| The evidence can be found in the SDCL Methodologies repository under the heading “fsm and efsm”. The evidence given is applicable as both the finite state machine and extended – FSM are explained, with the ideology behind them, differences and examples. |

**D1 Assess the merits of applying the Waterfall lifecycle model to a large software development project.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#application-merits-of-the-water-fall-method-for-large-projects> |
| The evidence can be found in the "SDCL-Methodologies" repository, under the heading application merits of the water fall method for large projects.  This constitutes as meeting the criteria, since it assesses the merits of applying the Waterfall lifecycle model to a large scale project. Re-alliterating the definition of the water fall method mentioned under the above heading “Waterfall”, whilst stating the merits as well as making it clear as to what a large scale project may or will need. A summary or conclusion is met as well suggesting whether the model is appropriate to the suggested situation (large scale project). |

**D2 Assess the impact of different feasibility criteria on a software investigation.**

|  |
| --- |
| Link: <https://github.com/matthewsides/Feasibility-Reports#impact-of-different-feasibility-criteria> |
| This evidence is applicable to the criteria as it assesses the impact of different feasibility criteria in relation to a software investigation. The evidence may be found under the heading "Impact of different feasibility criteria" in the README.md. |

**D3 Critically evaluate how the use of the function design paradigm in the software development lifecycle can improve software quality.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#5-how-the-use-of-the-function-design-paradigm-in-the-software-development-lifecycle-can-improve-software-quality> |
| The evidence for this criteria may be found in the SDLC repository under the heading “How the use of the function design paradigm in the software development lifecycle can improve software quality”. This constitutes as meeting the criteria as how the function design paradigm along with the benefits of its use can improve software quality. Whilst the disadvantages are also covered weighing in the pros and cons. |

**D4 Present justifications of how data driven software can improve the reliability and effectiveness of software.**

|  |
| --- |
| Link: <https://github.com/matthewsides/SDCL-Methodologies/blob/master/README.md#6-how-data-driven-software-can-improve-the-reliability-and-effectiveness-of-software> |
| Evidence for this is documented in the SDLC repository’s READ.ME, under the heading “How data driven software can improve the reliability and effectiveness of software”. This evidence is viable as how data driven software can improve the reliability and effectiveness of software is covered. Through stating the benefits and capability. The disadvantages or issues are also perused briefly. Before a summary is ascertained. |