**SDD HSC Major Project - Design**

**Due Date:**3:30 p.m. Friday 27th March 2015 (Week 9)

**Weighting:**15%

The second stage of the Software Design & Development Major Project is Planning & Designing the Solution.  This stage is key as proper planning will determine the quality of the final product.  This document outlines all the instructions and components needed to satisfy the requirements of this assessment.

**Syllabus Content:**

|  |  |
| --- | --- |
| **Designing and developing a software solution to a complex problem**  planning and designing   * algorithm design * refined systems modeling, such as: * IPO diagrams * context diagrams * data flow diagrams (DFDs) * storyboards * structure charts * system flowcharts * data dictionaries * additional resources * Gantt charts * logbooks * algorithms * prototypes * selecting software environment * identifying appropriate hardware * selecting appropriate data structures * defining files * purpose * contents * organisation | * document the software solution * generate a fully documented design for their project after communication with other potential users * use and modify a Gantt chart as appropriate * defining records * defining required validation processes * identifying relevant standard or common modules or subroutines * using software to document design * identifying appropriate test data * enabling and incorporating feedback from users at regular intervals * considering all social and ethical issues * communicating with others involved in the * proposed system * applying project management techniques   **Whole project issues**   * project management techniques * social and ethical issues * feedback from users at regular intervals |

**Outcomes:**

H1.1         explains the interrelationship between hardware and software

H1.2         differentiates between various methods used to construct software solutions

H1.3         describes how the major components of a computer system store and manipulate data

H3.1         identifies and evaluates legal, social and ethical issues in a number of contexts

H3.2         constructs software solutions that address legal, social and ethical issues

H4.1         identifies needs to which software solutions are appropriate

H4.2         applies appropriate development methods to solve software problems

H4.3         applies a modular approach to implement well structured software solutions and evaluates their effectiveness

H5.1         applies project management techniques to maximise the productivity of the software development

H5.2         creates and justifies the need for the various types of documentation required for a software solution

H5.3         selects and applies appropriate software to facilitate the design and development of software solutions

H6.2         communicates the processes involved in a software solution to an inexperienced user

H6.4         develops and evaluates effective user interfaces, in consultation with appropriate people

**Submission Requirements**

1. All documents must be submitted on Reshet.  There is no exception to this rule.  Any issues must be referred to the IT department.
2. URLs to the Logbook, Website and Backup must be submitted on Reshet.
3. All documents must be printed out and submitted to the teacher in a binder and each section clearly labelled.
4. All submissions must be made before the due date.  Should the submission be even one minute late, serious consequences may apply.
5. The submission must be complete.  Should any component be missing, serious consequences will apply.
6. The following is the list of documents that need to be submitted:
7. Gantt Chart
8. Logbook
9. Client Communication
10. IPO Charts
11. DFDs
12. Structure Charts
13. System Flowcharts
14. Data Dictionaries
15. Prototype Screenshots
16. Algorithms
17. Specifications & External Modules
18. File & Record Definitions
19. Test Data
20. Reflection
21. Website Screenshots

**Gantt Chart**

The Gantt Chart should be updated to include more detail for all milestones for the remainder of the project.  It should also record actual vs projected finish dates for the Planning & Design stage.  The Gantt chart needs to be accurate - faking it will result in lower marks.

|  |  |
| --- | --- |
| **3** | The Gantt chart is highly detailed and accurate, showing all tasks that need to be completed. |
| **2** | The Gantt chart shows the majority of tasks, but lacks some important milestones or is inaccurate. |
| **1** | The Gantt chart is lacks detail and is considered unsatisfactory. |

**Logbook**

The Logbook needs to be continually updated during the project.  Since it is online, the teacher will be keeping track of it during this stage.  Faking it will result in deduction of marks.

|  |  |
| --- | --- |
| **4** | The logbook is highly detailed, regularly updated (at least twice a week) and satisfies all of the above requirements. |
| **3** | The logbook is detailed, regularly updated and satisfies most of the above requirements, but could be improved. |
| **2** | The logbook has been updated at least once a week, but doesn’t satisfy all of the above requirements and could easily be improved. |
| **1** | The logbook is poorly completed and considered unsatisfactory. |

**Client Communication**

Client communication needs to continue throughout this stage, especially with respect to prototypes.  Failure to take into consideration the client’s perspective may result in a lower quality product.

|  |  |
| --- | --- |
| **2** | There is evidence of regular and involved communication with clients. |
| **1** | There is evidence of communication with clients, but there is a lack of detail and regularity. |

**Input Process Output (IPO) Charts**

The IPO Charts should be updated to reflect any changes from the Defining & Understanding stage.  In addition, it should now reflect all subroutines in the project.  Note that the Process column does not need to be highly detailed, but merely describe what each subroutine does.  The Input and Output columns need to be accurate.

|  |  |
| --- | --- |
| **5** | The IPO Charts clearly and completely indicate what the inputs, processes and outputs are throughout the system and are consistent with all other documentation. |
| **4** | The IPO Charts could be made more clear, are not absolutely complete, or are not completely consistent. |
| **3** | The IPO Charts give a general indication of the inputs, process and output for the system but could easily be improved. |
| **2** | The IPO Charts give some idea as to what the system is to do but are poorly completed. |
| **1** | The IPO Charts are poorly completed and considered unsatisfactory. |

**Data Flow Diagrams (DFDs)**

The DFDs need to be refined to the lowest level in the system.  All subroutines in the project should be represented by a process in a DFD.  In addition, all changes from the previous stage need to be reflected.  The DFDs need to remain consistent with all documentation.

|  |  |
| --- | --- |
| **5** | The DFDs are clear, complete, consistent and correct. |
| **4** | The DFDs are almost clear, complete, consistent and correct, but could be improved. |
| **3** | The DFDs give a good indication as to the flow of data throughout the system, but are missing components and could easily be improved. |
| **2** | The DFDs are missing a number of components but gives some indication as to the flow of data throughout the system. |
| **1** | The DFDs are poorly completed and considered unsatisfactory. |

**Structure Charts**

The Structure Charts need to be refined to the lowest level in the system.  All subroutines in the project should be represented in a Structure Chart.  In addition, all changes from the previous stage need to be reflected.  The Structure Charts need to remain consistent with all documentation.

|  |  |
| --- | --- |
| **5** | The Structure Charts are clear, complete, consistent and correct. |
| **4** | The Structure Charts are almost clear, complete, consistent and correct, but could be improved. |
| **3** | The Structure Charts give a good indication as to the structure of the system, but are missing components and could easily be improved. |
| **2** | The Structure Charts are missing a number of components but gives some indication as to the structure of the system. |
| **1** | The Structure Charts are poorly completed and considered unsatisfactory. |

**System Flowcharts**

The System Flowcharts need to be updated to reflect any changes made to the system.  They should remain consistent with all other documentation.

|  |  |
| --- | --- |
| **2** | The System Flowcharts are consistent with all other documentation and have been updated to reflect all changes to the system. |
| **1** | The System Flowcharts are poorly completed and considered unsatisfactory. |

**Data Dictionaries**

The Data Dictionaries are vital in this stage of the project as they ensure that all components of the project will interact with each other correctly.  All data that is passed between subroutines and is stored in the system must be in the Data Dictionary.  Any constants must be also be represented.  Variables local to a subroutine do not need to be defined.

|  |  |
| --- | --- |
| **3** | The Data Dictionary contains all the data input from and output to the user and stored in the system, with sufficient information. |
| **2** | The Data Dictionary indicates what data is used in the system but could for some reason be improved. |
| **1** | The Data Dictionary is poorly completed and considered unsatisfactory. |

**Prototypes**

Once the Storyboards and Screen Designs have been completed in the previous stage, a Prototype can be created for user feedback.  Prototypes can range from accurate images of all screens in the project, to fake user interfaces in Powerpoint or HTML, to actual user interfaces with limited functionality.

Prototypes are invaluable in this stage as they will ensure that, once the clients have signed off on them, they are the model to which the user interfaces are crafted in the next stage.  Accuracy is key here.  This is not to say that they will be exactly the same in the final product, but they must give an accurate user interface experience.

Getting client feedback from the prototypes is vital, as they can refine requirements that the developer may miss.

|  |  |
| --- | --- |
| **3** | The Prototype is high quality and gives a clear indication as to how the final product will look and work. |
| **2** | The Prototype gives an indication as to how the final product will look and work but is either low quality or doesn’t reflect the final product well. |
| **1** | The Prototype is poor quality and is considered unsatisfactory. |

**Algorithms**

Algorithms are vital to the understanding of the logic behind the project.  By getting the algorithms right, the implementation of the project becomes very easy.  For full details about algorithms, you should ensure that you are very familiar with the course specifications document.

You do not need algorithms for your entire project.  However, you need to include enough subroutines to satisfy the requirements of the course.  Your algorithms must show selection, repetition, file/database reading & writing, the use of external modules, and anything ‘complex’.  All algorithms must be ‘correct’ and easily understandable.

|  |  |
| --- | --- |
| **5** | The algorithms demonstrate selection, repetition, file/database reading & writing, the use of external modules, and everything complex.  There is a wide variety of algorithms, and they are completely correct. |
| **4** | The algorithms demonstrate selection, repetition, file/database reading & writing, the use of external modules, and some things complex.  There is a variety of algorithms, and they are almost completely correct. |
| **3** | The algorithms demonstrate selection, repetition, file/database reading & writing and the use of external modules.  There is a variety of algorithms, and they are mostly correct. |
| **2** | The algorithms demonstrate the ability to write algorithms, but there are number of problems and/or missing components. |
| **1** | The algorithms are poorly written and considered unsatisfactory. |

**Specifications & External Modules**

You should clearly specify the software and hardware specifications of your system.  They should clearly and precisely detail what operating systems your project will run on, what browsers, what software needs to be pre-installed and/or what hardware it will run on.  Your system should aim, in general, to run on as wide a variety of systems as possible.  The final system will be measured against what is stated here.

You should also clearly list the external modules and sections of code your system will rely on.  References to the modules and code need to be included.  The language(s) the system will be run on should be detailed here as well.

|  |  |
| --- | --- |
| **2** | The Specifications are clear and precise, are aimed to work on as wide a variety of systems as possible, and the list of External Modules are complete. |
| **1** | The above has not been satisfied and this section is considered unsatisfactory. |

**File & Record Definitions**

The precise format of all files and records need to be clearly and precisely defined.  For example, how high score tables are stored, user details on a database, etc…  It should be easy for anyone reading the format to write code to read from or write to any files or databases in your project.  If this changes in the future it will need to be updated.  You may consider using EBNF at this stage for any files.

|  |  |
| --- | --- |
| **2** | The File & Record Definitions are clear, precise and comprehensive. |
| **1** | The Definitions are incomplete or imprecise. |

**Reflection**

Reflection is a vital part of the learning process.  Innumerable studies and articles have been written on the benefits of reflection in all parts of life, especially education.  Reflection is integral to the Cultures of Thinking ethos at many schools, which is why it is consistently an assessable component in PoUs.  The most successful students will be able to reflect on their learning processes well.  The key to being able to reflect well is to ask yourself the right questions (and of course to then answer them well).  Some question you could ask yourself include:

* What have I learnt?
* How has my thinking changed from the beginning of this process?
* What problems have I encountered?
* How have I overcome issues?
* What issues have I not been able to overcome?  Why?
* What have I succeed at?
* What am I happy/unhappy with?
* What would I change if I had the time?
* What would I do differently next time?
* How do I feel about this?

There are of course many other questions that can be asked.  A successful reflection can be 500 words (or even more) in length.  Writing a couple of paragraphs is never a successful reflection.  Also consider that it is the quality of the answers that matter.  Keep in mind, that while you can reflect on other people, especially in group assessments, the reflection is all about YOU.

|  |  |
| --- | --- |
| **4** | The Reflection demonstrates a good ability to self-assess.  Many questions were asked and the answers were of a high quality. |
| **3** | The Reflection demonstrates the ability to self-assess, but could easily have been improved by asking more questions or giving better answers. |
| **2** | There was decent attempt to Reflect, but there were issues in being able to do it successfully.  It would be helpful to discuss this with the Director of Studies. |
| **1** | The Reflection was poorly completed and considered unsatisfactory. |

**Website**

The Website is the repository of all the information relating to the project.  In the future, it will contain links to download (or access) the project, installation instructions, user documentation, screenshots, description of how it works, and possibly development information (such as the project log).  It does not need to be professional, but should be clear and pleasing to the eye.

It is suggested that use is made of a web template, so that only content needs to be added.  In the future, a successful website would probably be multiple pages, but at this stage can be one page in length.  Screenshots (as well as the link to this page) should be submitted.  The website can be hosted on Hippo or on an external site such as GitHub or Google Code.

|  |  |
| --- | --- |
| **2** | The Website is pleasing to the eye and contains further information about the project, including development information. |
| **1** | The Website contains information about the project, but could be improved with little effort. |

**Test Data**

Your project needs to be designed so that it can be tested in a modular way.  As such, test data for all non-trivial algorithms needs to be designed at this stage.  By creating test data now, you will ensure that all code written will operate exactly as originally specified.

Your project will be evaluated as to how well it passes the test data created in this stage.  If there is insufficient test data, then your project may be evaluated poorly.

|  |  |
| --- | --- |
| **3** | There is sufficient Test Data for all non-trivial subroutines. |
| **2** | There exists Test Data for most components of the project. |
| **1** | There is insufficient Test Data and this sections is considered unsatisfactory. |