GCE Computer Science (7517)

The Practical Project

|  |  |  |
| --- | --- | --- |
| Centre number |  | Centre name |
| 22133 | The Perse School |

|  |  |  |
| --- | --- | --- |
| Candidate’s full name |  | Candidate number |
| Abhinav Gupta | 2262 |

|  |
| --- |
| Project title |
| Travelers of Catan |

Contents

[1 Analysis *(9 marks)* 3](#_Toc451422998)

[1.1 Identification of the problem 3](#_Toc451422999)

[1.2 Research carried out 3](#_Toc451423000)

[1.3 Identification of the prospective user(s) 3](#_Toc451423001)

[1.4 Detailed background to the problem 3](#_Toc451423002)

[1.5 Numbered measurable, appropriate specific objectives of the project 3](#_Toc451423003)

[1.6 Modelling diagrams 3](#_Toc451423004)

[2 Design *(12 marks)* 4](#_Toc451423005)

[2.1 System design overview 4](#_Toc451423006)

[2.2 Algorithms 4](#_Toc451423007)

[2.3 Data structures 4](#_Toc451423008)

[2.4 File structure and organisation 4](#_Toc451423009)

[2.5 Database design 4](#_Toc451423010)

[2.6 SQL queries 4](#_Toc451423011)

[2.7 User interface design (HCI) 4](#_Toc451423012)

[2.8 Hardware selection/design 4](#_Toc451423013)

[3 Technical Solution *(42 marks)* 5](#_Toc451423014)

[3.1 Code listing 5](#_Toc451423015)

[4 System testing *(8 marks)* 6](#_Toc451423016)

[4.1 Test plan 6](#_Toc451423017)

[4.2 Annotated screenshots of test results 6](#_Toc451423018)

[5 Evaluation *(4 marks)* 7](#_Toc451423019)

[5.1 Comparison of project performance against the objectives 7](#_Toc451423020)

[5.2 Effectiveness of the solution 7](#_Toc451423021)

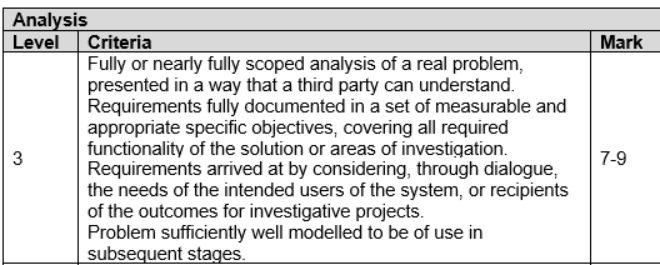
[5.3 Analysis of user feedback 7](#_Toc451423022)

[5.4 Possible improvements 7](#_Toc451423023)

[6 Appendix 8](#_Toc451423024)

[6.1 User feedback (authenticated by assessor) 8](#_Toc451423025)

# Analysis *(9 marks)*



In the analysis section we are looking for:

* A detailed description of the problem / investigation
* Clear evidence that research has been performed (dialogue)
* A clear set of objectives that will be useful across the later sections of the project
* Modelling of the proposed solution that will be of use to later design work

? Having read the analysis stage:   
 Do you understand what the project is going to do?

Has the student set clear objectives that are detailed enough?

## Background to/ Identification of the problem

Is this a detailed description of the background/ identification? Is there evidence of an understanding of how the current system contributes to the problem? Is there evidence that the user is well understood and the context is understood? If a game or puzzle, have you described the rules in sufficient detail along with any alternative rule options that may exist? Is there a problem related to having to carry a physical game around or not having time to play a full game in one sitting and the associated issues with packing away and restarting? If a puzzle is there an issue with getting access to enough different puzzles to become proficient? Is there any way to help become better at the game or puzzle?

## Research carried out

You must include detailed evidence of research into the background of the project including any knowledge acquisition for the problem domain. Include screen shots or diagrams of the **current system** and interview **transcripts** with the current user. Ensure questions are **detailed** and really scope out the full complexity of the new system and show how it can solve the problem. Identify any **key algorithms** that you may need to write – either ‘complex user defined’ ones to implement the rules of your program or known algorithms e.g. Minimax. Include an analysis of how deep (how many levels) you are likely to be able to go in a minimax implementation based on your product. Does your game allow for **repeat turns** and so would minimax need to be modified from its pure form? Identify any key **data structures** that might be useful in building your solution. Are there any **research papers or web articles** that describe a **playing strategy** in your product that you will be able to implement. Note that you are not implementing the algorithm, but identifying it as ‘needing to be implemented’. Make specific links from the research of existing solutions to the project objectives. Will your product use any networking (client/server) and if so what networking technology will you use (websockets?)

## Identification of the prospective user(s)

This is a short paragraph that identifies the user and describes them and their need. It should include detailed interviews with the prospective user (not superficial yes/no questions) where the user’s opinion is sought in detail. Make specific links from the user’s requirements to the objectives of the project.

## Numbered measurable, appropriate specific objectives of the project

These should cover all required functionality of the solution or areas of investigation ('appropriate' means the specific objectives are single purpose and at a level of detail that is without ambiguity)

NB These must be complex even if you are unable to code them all fully! They should be described in sufficient complexity as to meet the standard required of A level. Each user need should be identified as a measurable objective and then within that objective specific outcomes should be identified and listed. These should not include any add-ons of standard algorithms nor any code that will be generated by a GUI. This is the most important section for you to get right!!!!!!!!

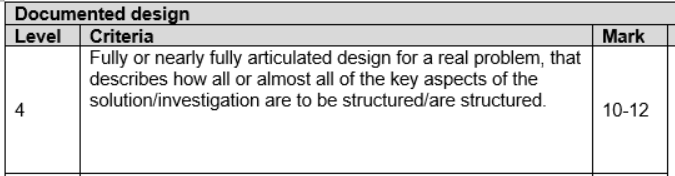
## Modelling diagrams

E.g. E-R model, High level software object model, Data Flow Diagrams, mathematical models, the flow through the program.

**Important read – Examiners Report**

Analysis There is encouragement that a student should gather details for the project from users via a dialogue of some form. Some of the interviews seen were very detailed and clearly gained relevant information for development of the project. Unfortunately, it was also common to see very short interviews which gathered no real requirements for the project to be assessed highly by centres. Students should be encouraged that in the interview it would be beneficial to ask probing questions to find out the real requirements of the user(s) and not just the kind of colours to be used or whether they like playing games. The analysis should contain a list of the objectives set by the student for their technical solution. It was pleasing to see many students provide a detailed list of objectives that indicated both the requirements to be met and the complexity that this might involve. Students who submitted vague and brief objectives would struggle to pick up high marks in the analysis section and it would also be common for the rest of the project to suffer slightly. Weak objectives also make awarding the completeness mark hard as consideration must also be placed into what an A-level student would be expected to achieve. The analysis section is to contain some modelling of the proposed system and it was pleasing to see students complete this in a variety of ways. Those projects that needed data processing usually included some discussion of the data required and DFD or ER diagrams. Students looking to produce a game sometimes struggled with the modelling section and also left the reader not understanding what their idea actually was. Students completing gaming projects could consider sketching out some ideas for the game and discussing the game flow as part of their modelling section.

# Design (12 marks)



In the documented design section we are looking for:

* An overview to the whole system design (module breakdown / objects / units / web pages …)
* Clear design & detail for some of the complex algorithms
* Detail for the data to be used (database design / data structure design)

Having read the documented design:

Do you understand how the project is going to work as a whole?

Do you have a clear understanding as to how some of the complex algorithms will work?

Do you understand how data will be processed / structured?

## System design overview

Description plus diagrams such as class diagram: For each class describe its purpose (high cohesion) and each public method (interface/low coupling)

Flowchart: Give general top-level flow of the system from running the program. May be more than one flow chart for different use cases.

Data flow diagram: Top level diagram showing how data moves between different parts of your software for different use cases e.g. logging in accesses database, saving access database.

Swim-lane diagrams for networking messages

## Data structures

Give examples of what data would be stored in the data structure in the context of your project. Do not be afraid to be verbose

## Algorithms

**Pseudocode** or similar for key algorithms essential to the success of the project. These MUST be linked into the how they fit into the project as a whole. Show how the algorithm would **affect** the **data structures** used in the context of your project. Give a **trace table/dry run** of sections of your key algorithms to **show** how they work.

## File structure and organisation

Show what files are going to be created and which software objects will be implemented in each file. Include folder structure for images, database file if used.

## Database design

Provide schema for database. Give example data that would be stored in the database for your project.

## SQL queries

Give **all** SQL queries that would be needed to implement specific project objectives. (can be written after solution completed)

## User interface design

Provide wire-frame diagrams of the graphical user interface, positioning of buttons, different screens. Give any menu commands that would be available in a terminal interface

## User guide

Write a user guide to say what software or modules the user will need to install or have available on their system, and how they should run your program from the command line

**Examiners report – an important read!**

It was pleasing to see good students carefully structure out the design of their technical solution. Effective use of diagrams to provide an overview of the whole system, key data requirements being identified and explained along with a breakdown of the complex parts leading to pseudo-code and/or code snippets would lead to a high mark. It was also common, however, to see a more random attempt at the design documentation including just pasting code across with no detail as to the design process or how it would link into the main system. So, for example, just providing stock algorithms for merge sort and binary search does not help the reader understand the design of the system.

Having a section titled ‘sample of SQL queries’ is not very beneficial in providing a reader an understanding as to how the system will work. Students would do better to design out a particular form/page and then discuss the algorithms required for that part of the system including the SQL queries to be used for that part. Students should be encouraged to think about the data to be used by the system. In a quiz system, for example, it would be beneficial to provide examples of the kind of question(s) to be asked. For a simulation it would be good to see how the formulas are to be used alongside, for example, a sketch of the trajectory of the projectile being modelled.

For a game a student could sketch out the grid or level and talk through, for example, the movement of any enemies.

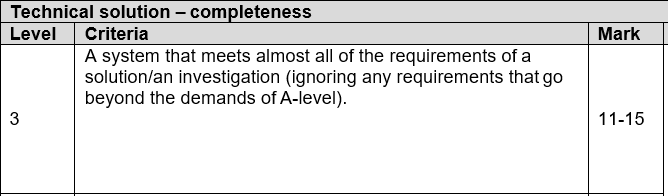
It was common to see algorithms appear without the reader having any real understanding as to how these fitted into the system and a few sketches or examples of the data to be used would help. So, for example, one student produced an excellent Sudoku solver which had some complex pseudo-code in the design section. This code was hard to understand but a few sketches of particular board layouts showing how the individual functions would perform would have really helped. It was common to see many students make use of well-known algorithms such as the merge sort. Just providing the pseudo code for this algorithm is not going to help their documented design mark. If the student talks about how this algorithm is going to be used by the system and integrated then this is beginning to pick up some credit. If the only pseudo-code or algorithm design a student attempts is based around merge sort, quick sort, binary search or other well-known algorithms without any attempt at looking at other parts of the system then the student should not be scoring highly in the documented design section.

# Technical Solution *(42 marks)*

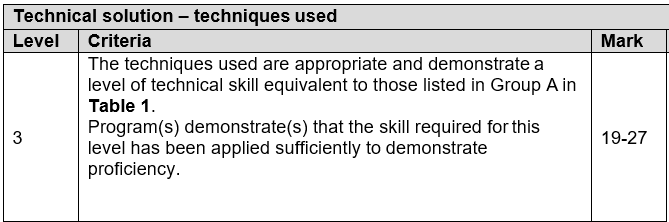
## Completeness Section

In the completeness section we are looking for a consideration as to:

* Has the project met the objectives set by the student in the analysis stage?
* How well have these objectives been met (consideration to HCI / features)?
* Does the technical solution match the original project background description?



## Technical Skills Section



In the technical skills section, we are looking for a marker to:

* Identify parts of the code where complexity is clearly evident and map to Group A/B/C  
  **[This can be helped by a student producing an overview guide]**
* Place consideration into the coding style and comment upon this
* Consider the overall effectiveness of the final solution (does it work how it should?)

### Overview Guide

You should produce a guide here that highlights techniques/ skills in sections A B and C with commentary. Your code should be commented to show where specific technical skills are being used e.g. # SKILL GROUP A – Graph Traversal

### Code listing

This should be fully commented and broken into suitable sections with subheadings. You can turn this section to landscape to make it easier to read. Ensure this is **fully annotated** with comments. Can use a website like hilite.me. Do not use ‘dark mode’ images or screenshots

# System testing *(8 marks)*

## Test plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Purpose of test | Test data | Expected outcome | Reference to test result |

## Testing Videos

Record your program being used, narrate each objective being shown to have been implemented. Give the objective in the audio of the recording. Provide bit.ly shortened links in the document to your videos which have been uploaded to youtube. Use a large monospace (e.g courier new) font with no underlining e.g.

https://bit.ly/abc0DEF12

# Evaluation *(4 marks)*

## Comparison of project performance against the objectives

## Effectiveness of the solution

## Analysis of user feedback

This should include the original user who was involved at the analysis stage

## Possible improvements

Be verbose about what could be improved – make a full statement. Get user feedback for improvement

# Appendix

## Git log

## References to web sites or other resources used

