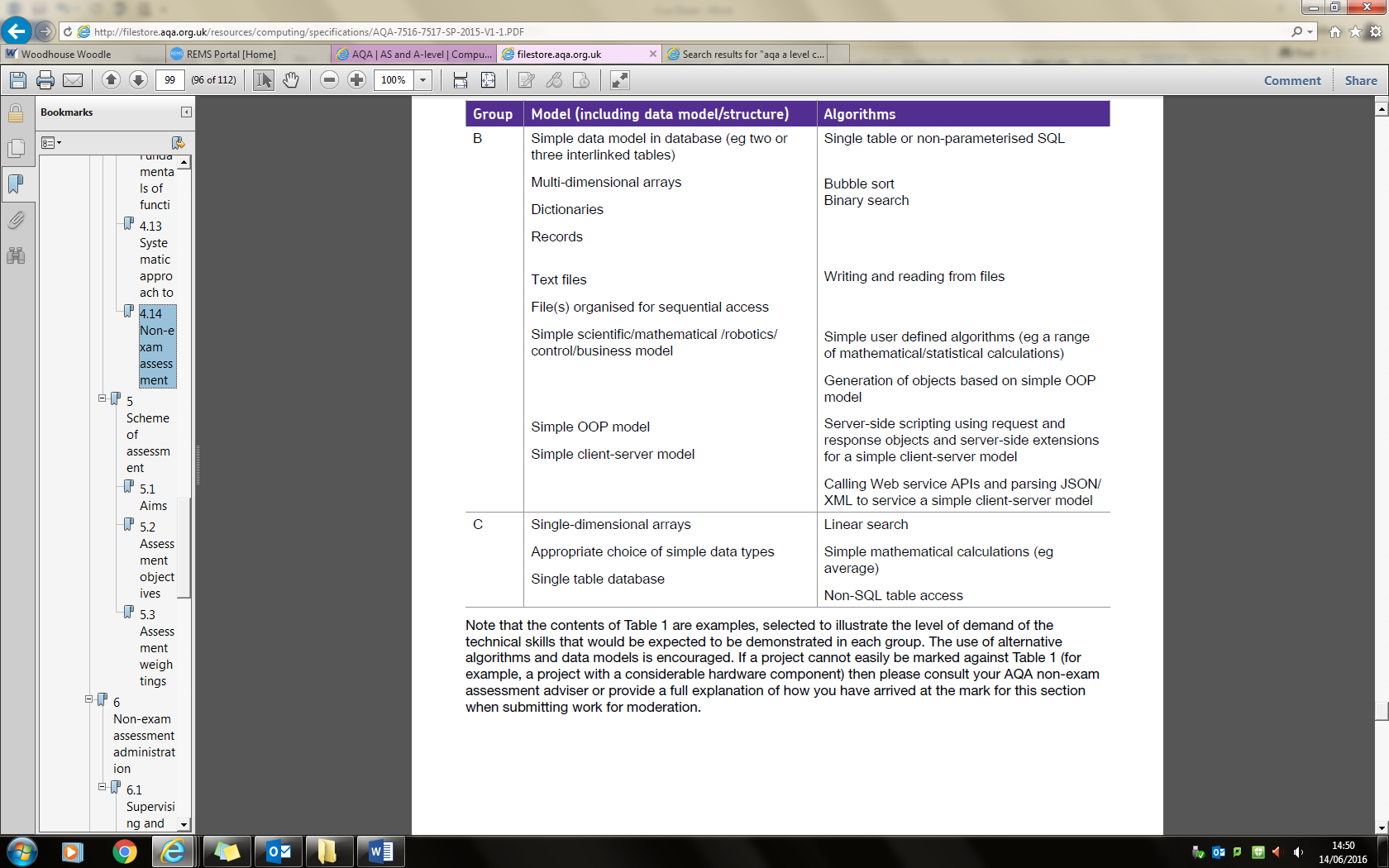
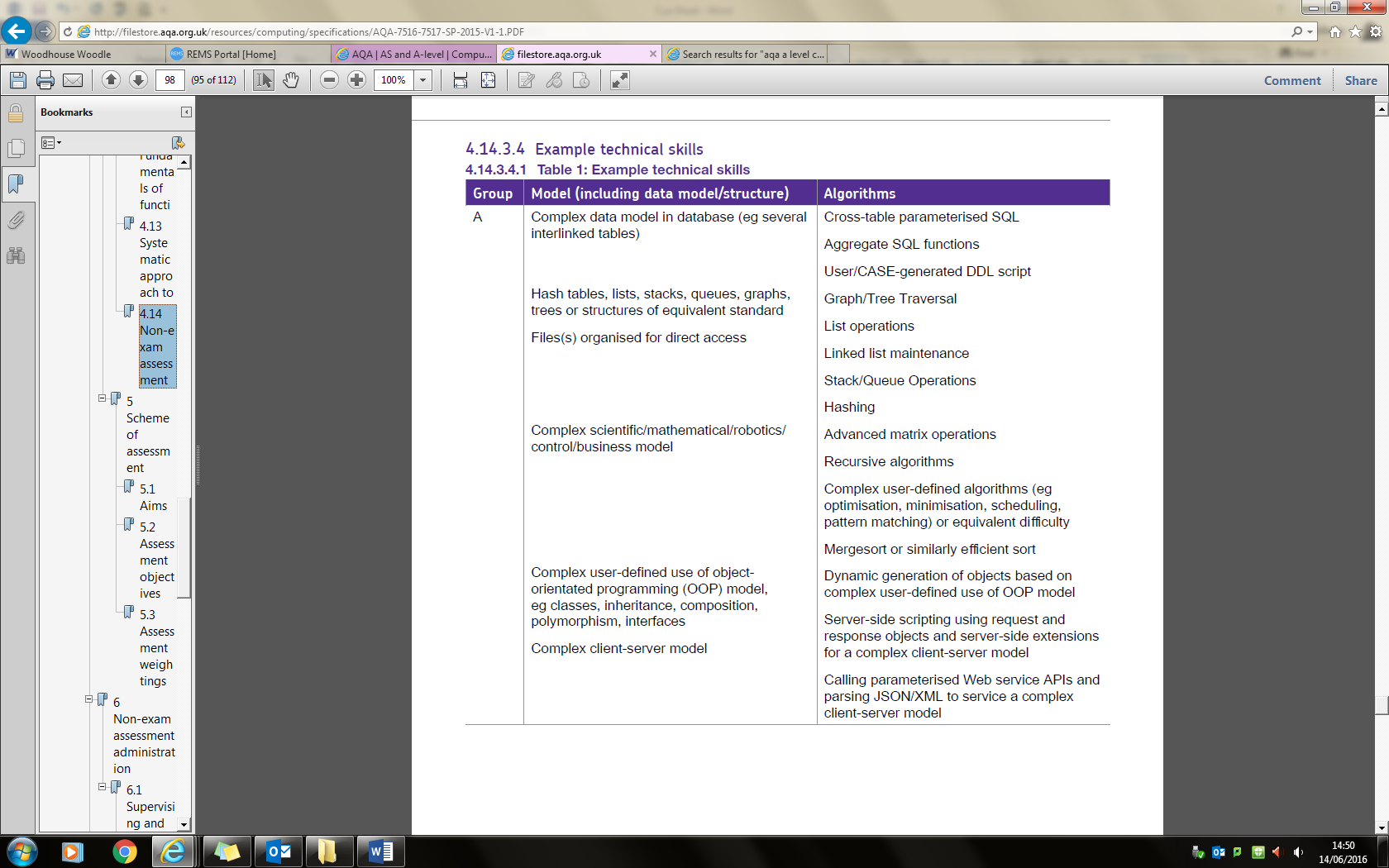
NEA: Pre-Analysis

Use the cue questions provided to complete your proposal. You will need to identify:

* the real problem (could have a real end user) & the need for a solution
* the potential (genuine) problem to be solved (what it must do)
* a numbered list of tentative key objectives – to include **technical skills**
* a possible method of solution (including programming language)

The proposal document should be a maximum of 2 sides of A4.

**Technical Skills:**

NEA: Proposal Cue Sheet

|  |  |
| --- | --- |
| **Cue Question** | **Answer** |
| **Type of Project?** | Investigating a RPG 2D game |
| **Who is your client (**solution project**)?**  Named person, role plus company/organisation**,** address, size of organisation (no. of staff/members/customers), when was it established & its growth over the years.  **Explain what are you investigating?**  Type of game/app/topic. | People who like adventure game or want to get satisfaction.  Gamers |
| **What is the context?** (Describe the background circumstances, for example a short description of the organisation /business client works for/short description of features you are investigating/comparison of comparable software/app/systems) | It is a maze runner game with monster combat system. And upgrade the weapon in order to deal more damage to the monster |
| **What is the problem that your project is trying to solve / what are you trying to achieve from the investigation and create?** (The reason for undertaking the project / The reason for choosing the type of game/app/topic..) Give some background if possible. | Learn Unity and C# graphical programming. Clear OOP plan and map generation. |
| **Solution based - Do they currently have a system? What type of system? How does it work?**; i.e. manual, partly manual, computerised (spreadsheet), etc. Describe in detail how the system works. |  |
| **Solution based - How you propose to solve the problem?** Include a possible method of solution (including programming language) |  |
| **What *exactly* will you produce?** (These are the ‘project deliverables’)  These **must** include the *solution* (explain what it is, the information content to be included and how they will be presented. | 2D RPG game on mobile platform  What the game should do |
| **How will the solution be used?** (include hardware and software/programming applications/language needed) | Unity engine |
| **Who will use it?** (The target audience- narrow this down because you will have to say what you have done to make the solution suitable). | people who like strategy game and level challenging or just spending their free time |
| **How will the solution affect working practices?** (or the way things are done). | Collaboration and problem solving, enjoy the game and keep playing even though all level are passed. |
| **What resources will you need?** List all Hardware (apart from a computer) & Software. | Unity(coursea or youtube tutorial ), Visual Studio with C# support, GIMP (photo editing), testing device(mobile phone) unity asset store for graphic and texture design |
| **Objectives – quantitative & qualitative**  (include technical skills on page 1 you wish to use and what is the purpose for using them) | Achievement system(tree)  Login system (stat restore and backup)  Leaderboard(local server)  Game save  Health system  Inventory system  Token/money system  AI enemy(shortest path algorithm)  Combat system  Undo system  Ghost system  Level system  Objectiive and Quest system  Map generation(using hash) system |

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **Algorithm/Model** | **Used** | **How/Where** |
|  | * Scientific * Business * Mathematical * Control * Business | **Control** | **A 2D platformer game is mostly control based model which requires user input and a output for game object changed** |
| A | Complex Data Model:   * >= 3 tables interlinked * Cross-table SQL * Aggregate SQL * User/CASE-generated DDL scripts |  | Database for player detail, preference, sync location, setting , password and username, level reached and highest score, token |
|  | * Hash tables * Lists * Stacks * Queues * Graphs * Trees |  | Stack for undo system  Queue for objective checking or level unlocking  Hash table for map generation |
|  | Recursion  Merge sort | Merge sort | Sort data before input into the database |
|  | Complex OOP:   * Classes * Inheritance * Composition * Polymorphism * Interfaces | Classes  function | Game object include lighting, map, camera setting and entity, interface for different system  (for reuse) |
|  | * Server side scripting using request and response objects. * Calling parameterised APIs * Parsing JSON/XML |  | Json for output data(hash of map) |
| B | Multi-dimensional arrays |  |  |
|  | * Bubble sort * Binary search |  | Merge sort is used for time efficiency (shortern the loading time) |
|  | * Dictionaries * Records |  | Shop and the value |
|  | * Text Files |  | Backup and sync to drive and restore from text file(may use has to encrypt the data) |

Idea