|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification details** | | | |
| **Training Package Code and Title** | ICT - Information and Communications Technology (Release 8.1) | | |
| **Qualification National Code and Title** | ICT50220 Diploma of information Technology (Release 2) | **State code** | BGJ4 |
| **Assessment Title** *(as per DAP)* | Assessment Task 1 | | |
| **Unit National Code & Title** | ICTPRG443 Apply intermediate programming skills in different languages | | OBT13 |
| ICTICT430 Apply software development methodologies | | OBU02 |
| ICTICT523 Gather data to identify business requirements | | OBU27 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Date Due** | Week 10 | | **Date Received** | |  | |
| **Student Name** | Samuel Bailey // 30106121 | | | | | |
| **Student Declaration** | I declare that the evidence submitted is my own work:  SBailey | | | | | |
| **Assessor Name** |  | | | | | |
| **Assessment Decision** | Satisfactory | | | Not Yet Satisfactory | | |
| **Assessor Signature** |  | | | **Date** | |  |
| **Is student eligible for reassessment (Re-sit)?** | No | Yes | | **Re-assessment Date:** | | Week Nineteen |

|  |  |  |  |
| --- | --- | --- | --- |
| **Feedback to student** | | | |
| *Via Blackboard (LMS) – Please check [Grade] section.* | | | |
| **Feedback from student** | | | |
| *Via Blackboard (LMS) – Please use [Comment] section during submission.* | | | |
| **Student signature** |  | **Date** |  |

|  |  |
| --- | --- |
| **Assessment Instructions** | |
| **TO THE ASSESSOR** |  |
| Type of Assessment | Individual Portfolio |
| Duration of the assessment | 10 class sessions (Weeks 1-10) |
| Location of assessment | Classroom |
| Conditions | Assessor to ensure that the noise levels, natural interactions and time variances are maintained as it would be in the Software Development industry.  Learners are required to complete the required tasks in class and submit the required documentation electronically via Blackboard |
| Elements and Criteria | As detailed in the assessment plan  You are required to make sure that all students meet the elements, performance criteria and oral communication items as outlined in the provided solution |
| **TO THE STUDENT** |  |
| Purpose of Assessment | You are required to show you can:   * Carry out intermediate programming activities, such as coding, debugging, testing, and creating applications. * Use a traditional and non-traditional development methodologies for application development. * Identify, analyse, and document business requirements according to project requirements.   The student must demonstrate the ability to complete the tasks outlined in this assessment and is expected to use systematic analytical processes and effect time management to meet the goals/deadlines outlined in the DAP. |

|  |  |
| --- | --- |
| Allowable Materials | Blackboard Topics: SDLC, Weekly readings (PDF), Example programs and Independent Outside of Class Activities |
| Required Resources | Web links and example code can be downloaded from the Blackboard portal.  PC with MS Visual Studio, MSOffice.  Internet Access to MSDN, GitHub and www.citems.com.au/ |
| Reasonable Adjustment | In some circumstances, adjustments to assessments may be made for you. If you require support for literacy and numeracy issues; support for hearing, sight or mobility issues; change to assessment times/venues; use of special or adaptive technology; considerations relating to age, gender and cultural beliefs; format of assessment materials; or presence of a scribe you need to inform your lecturer. |
| Assessment Submission | All questions and programming activities must be attempted. All written answers must be submitted in this assessment document in the appropriate space.  Use of research tools and peers in formulating answers are acceptable – but work submitted must be your own work.  Final project documentation is to be uploaded to the appropriate area in the Blackboard course created for this unit.  If you are marked as NYS (Not Yet Satisfactory) on your first attempt, you will be provided with another opportunity to re-attempt the assessment. |
| Portfolio Description | A project of programming tasks and written questions which should be completed in class and finished in the students’ own time on a weekly basis as per the Delivery and Assessment schedule.  Task 1 – Planning Phase  Task 2 – Design Phase  Task 3 – Development Phase  Task 4 – Testing Phase  Task 5 – Review Phase |

# Task 1: Planning Phase

## 1.1 Brief

CITE Managed Services has been approached to develop a Minimal Viable Product (MVP) that manages sensors for Sensing4U that provides metrics and statistics to a wide range of companies ranging from electrical providers to farm management. These clients have a wide range of sensors and automated devices to maintain and extract data from throughout the course of their work.

You have been assigned to the project to perform initial discovery of information by researching existing techniques and technologies that are being used within the space that Sensing4U operates in along with being the primary contact. Once your preliminary discovery has happened you are to meet with the client to brief them on options that they can take along with trade-offs with your different approaches.

The initial requirements from Sensing4U are:

* Device brands are varied and must be capable of reading from a generic file type,
* Time frames are varying as some devices report information more frequently than others,
* The application must be capable of undergoing rapid iteration as bugs cannot be left unresolved for long periods of time,
* External documentation must be generated so allow for other teams working in parallel to interface with the application once it has been developed.

## 1.2 Competitor Analysis

The initial phase of application development is to determine what competitors are doing, along with the risks and benefits of implementing it into a system. Competitors will try and hide the risks of the feature while emphasising the benefits.

Sensing4U is currently aimed at competing with applications that attempt to consume sensor data and display it within a single application or dashboard for users to perform data analysis, maintenance tasks, or identify issues with sensors.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Description w/ risks & benefits** | **Source** |
| Real-Time Data Graphing | A type of real-time data analysis that can receive streamed data, use algorithms to process data as it arrives and display them in a graph.   * Risk: By automating how you want data to be displayed (which graph to use), you may end up using a graph that doesn’t highlight the important relationships you need to see or miss something. * Benefit: This is incredibly helpful for making real-time decision on your data, increasing potential response times and decision making. * Risk: Collecting real-time data over multiple IoT objects can lead to a large amount of data to handle. | <https://epochjs.github.io/epoch/real-time/> |
| Time Series database | A database held either locally or on the cloud that specialises in time-stamped data.   * Risk: They can be hyper-specialised for time-stamped data, meaning they are limited at helping data analysis for different aspects other than time. * Benefits: Well-suited to help look at data over large areas of time and queries focused on time periods. | <https://www.influxdata.com/time-series-database/> |
| Dashboard | A UI or GUI that shows the user of the program the data and allows for data analysis.   * Risk: Given the large amounts of data, it must be built to allow for good performance, likely only showing specific data and not overloading it. * Benefit: Helps to visualise the data and navigate it. | <https://www.dspace.com/en/inc/home/applicationfields/foo/sensor-data-management.cfm> |
| Data Sharing capabilities | The ability to share data sets or save graphs/set for communication and collaboration. This could be writing the data out or creating an image file of a created graph.   * Risk: data security is a large concern and when dealing with time-stamped data it may be important to keep track of who accessed the data and wrote or shared it to keep things copasetic. * Benefit: Allows for much easier collaboration and analysis, helping with presentations and business meetings about progress. | <https://www.researchgate.net/publication/271549440_My_sensors_A_system_for_secure_sensor_data_sharing_over_internet> |
| **Summary** | |  |
| ***Hint:*** *how would these features integrate with the requirements given*  Real-Time Data graphing would only be possible with some sensors given that some are set up to report more frequently than others. This makes it seems like a secondary feature that could be added later given that it isn’t useful for all sensors initially.  The dashboard and time-series databases are both incredibly useful and should be implemented upon first launch hopefully.  Data sharing capabilities are useful but given the application should be able to receive the data and work on individual computers within the office, it can be delayed for our first iteration. | | ***Me*** |

## 1.3 Software Development Lifecycle

### 1.3.1 What are the key stages for developing a medium-size application?

Include a description of each of the stages to support why they are important.

|  |
| --- |
| **Planning:**  Planning is essential as it involves gathering the requirements and defining the concept and purpose of the application. For example, knowing that there will be different brands of sensors and it must read simple files to work is important to know before beginning.  **Design**  Design is when all that knowledge is put into use to create the architecture of the application code and the GUI/Dashboard. Understanding how the front and back end communicate will be an important thing to figure out in this stage.  **Development**  Development is the coding, the actual writing of the program. Problems can arrive in this stage, and the project can need to pivot, depending on the type of SDLC format the team is working on.  **Testing**  A minimum viable product must at the very least work with very little bugs, this phase ensures quality control and finds errors in the code so that it is not deployed in an unusable state. It involves running unit tests and functional testing.  **Deployment**  The actual giving of the product to the client. Uploading and delivering it, ensuring that servers are working and the client can use the software as intended.  **Maintenance**  Given that our program will need to be iterable, on a relatively fast basis, this step is crucial in catching these problems and fixing them as they arise. The design should be created with this stage in mind given its importance.  Source: <https://simicart.com/blog/mobile-app-development-process/> |

### 1.3.2 What factors influence the development process for medium-sized applications?

|  |
| --- |
| **Methodology**  Choosing which SDLC methodology you are going to use is important, as it determines what will need to be built initially, the requirements and the projected delivery time.  Source: <https://itrexgroup.com/blog/software-development-team-structure/>  **Modularization and Team Size**  Modularization is a concept that allows for splitting the functions of a program between various programmers so that those areas can be worked on simultaneously. It will be very useful for larger teams, and perhaps less so for smaller ones. It will also depend on the type of communication between the modules that will determine if they can be programmed in parallel with more co-ordination or can be split with less communication between programmers needed. Given the quick iteration needed however, modularisation could be extremely helpful given the testing and changes could be focused only on the necessary modules.  Source: <https://vfunction.com/blog/modular-software/#toc-heading-8> |

### 1.3.2 SDLC Methodology Types and Use Cases.

|  |  |
| --- | --- |
| **Style** | **Description** |
| Waterfall | All SDLC stages are completed sequentially with little to no iteration. It is useful for projects where the scope and requirements are clear and fixed, given that changes cannot be made in the middle of the process. |
| Iterative | The iterative SDLC focuses on continually rebuilding the project repeatedly and incrementally improving it in each version. The first iteration will have a small subset of requirements and go through the design and development, testing and implementation phases. Once done, another iteration will start with the same phases, but this time increasing what it can do and its complexity. This can be useful for projects needing fast turnaround times |
| Agile | Agile is also an iterative method however it expounds on this by splitting up each feature into its own iterative development and comes together to add features as they are finished, allowing for new additional features to be requested later in the development life cycle and still be met. It should be used when the project is not set in stone and could be subject to changes as you work. |
| RAD | RAD stands for Rapid application development and is like Agile and Iterative methods in its ability to start small and work towards all requirements with various iterations between then. It also requires a higher collaboration with the client as it focuses on fast delivery of prototypes to communicate need between client and programmer. |
| Spiral | The Spiral methodology tries to combine the iterative nature of Agile and the systemic approach of Waterfall. It allows for unclear client needs while its focus is on maintaining safety in high-risk projects. It works for larger more complex projects where prototypes are wanted but must be functional and safe to use. |

### 1.3.3 Recommendation for Application

With your understanding of methodologies please provide a recommendation of which methodology would best suit this application and provide reasons.

|  |
| --- |
| **Recommended Methodology** |
| ***Hint:*** *explain how they relate to the requirements currently expected and how the methodology supports the application to be developed.*  I believe the iterative method to be the most useful for this project, as we have defined some features have more use than others initially, meaning we can focus on implementing them first before iterating again and adding things like Real-Time Data Graphing and File Writing. |

### 1.3.4 Why is understanding a clients business domain important in software development?

Include how business requirements impact the choice of methodology used, and the common domain features that may impact project requirements.

|  |
| --- |
| It affects how you would approach the design of the classes and code within a program explicitly. If you were creating an application for an online bank, you would need to consider what access modifiers you would put on the value that holds an account’s current balance. |

### 1.3.5 Who are key stakeholders in a software development project?

Include what their duties may be along with why their involvement is important, user a power – interest matrix to assist in providing reasoning.

|  |
| --- |
| **Customers/End Users – High Interest, High Influence**  End Users are the most important stakeholders of a program as they will be the ones using it. They are the primary source of project direction.  **Project Managers – High Interest, High Influence**  Project managers are the overseers of the entire project, ensuring that the entire scope of the project meets the customer’s needs. They actively are managing the deadlines and development methodology.  **Development Team – High Interest, Low Influence**  The development team are invested in the creation of the program, but they usually do not have much control over the large scope of the project. |

### 1.3.6 What factors should be considered when selecting hardware or software products?

Include in your answer general features and capabilities that may impact the product.

|  |
| --- |
| **Budget**  Budget is ones of the first things to consider when acquiring anything for a profitable project. Go over budget and all your work is for naught as you have lost money on the project.  **Reliability/Robustness**  For hardware, one must ask where the sensors will be placed, will they get wet? Is it in a moving machine? What elements will it be affected by or subjected to? Ensuring you buy products that can remain reliable in the environment they will be placed will be important.  As for software, you must find software that is robust with error catching and can handle the loss of signal or inputs from sensors without crashing. For our product this will mean ensuring a strong and robust signal or connection with our devices. |

## 1.4 Quality Assurance Practices

1.4.1 List and describe some practices that can be used to ensure in software development that all stakeholders are understanding of the requirements and the current state of the application.

[CITEMS](https://www.citems.com.au/) contains documentation that may assist in QA practices.

|  |  |
| --- | --- |
| **Practice** | **Description** |
| Testing during Development | Ensuring that each module completed is unit tested and has no recurring bugs, and that they merge and function correctly together. |
| Code Reviews | A regular peer review of each piece of code. |
| Risk Management | Procedures that are set to avoid things like bugs or catch them early. This can be ensuring proper documentation, ensuring code practices compliance, etc. |
| Configuration Management | Common platforms like Git allow for the process of configuration management. It is a system to ensure that metadata (endpoints to external services, specifications of hardware resource allocation), is handled and available in one place, and ensures that each code module works in tandem. |
| Process Monitoring | This is about monitoring the software while in use, to see how it performs, where it is failing or drawing too many resources, how the CPU and RAM function while running the program and how much of each of these resources it uses. |

### 1.4.2 Identify coding standards and why they important to software development

Include at least three (3) coding standards and why they are important to software development.

|  |
| --- |
| **Naming Conventions**  Ensuring the same naming conventions is a practice used to improve readability. With code reviews and especially for projects where other’s will be reading your code, readability will be paramount so they can understand quickly what your code does and how it does it. This improves Quality Assurance and code merging later in development.  **Classes, Functions and Interfaces**  Object Oriented programming is a common programming method, and it focuses on re-usability and modularity. By placing data into objects, splitting modules into specific classes and functions, the code can become much less complex and re-usable. This also allows for errors and bugs to be found quicker due to the modular nature of the code.  **Comments and Documentation**  Comments in code go one step further than Naming conventions in improving readability. It can allow you to plainly state what a function or block of code does so that others can easily understand the overall makeup of your code. Documentation then, goes another step further, allowing you to write in plain words in a text file, what you are trying to achieve, how you are going about doing it, what problems you have found and everything a user or peer could need to know about your development process and design architecture. |

### 1.4.3 What are some key documentation techniques used in software development?

Include in your answer why they are critical in software development and a brief description of what they are used for.

|  |
| --- |
| **Requirements Documentation** is done within the planning stage, it involves finding all the user, business and technical needs of the program to help begin the design process and set its parameters.  **Design Documents** are done to explain and outline the data flow and architecture of the program before it is made. It can involve UML diagrams of states, classes and sequences.  **User Manuals** are finished near the end of a development cycle. They outline how an end user can and will use the program, step by step. |

### 1.4.4 What are the key steps in creating a project plan?

Include each step with a brief description of what their intended objective is along with two (2) tools that can be used to support the project creation process.

|  |
| --- |
| <https://www.softwaredevelopment.co.uk/blog/how-to-effectively-plan-and-execute-a-software-project-project-delivery-plan/>   1. **Define the Goal**   What are you trying to achieve? Ensure the aim of your project is clear.   1. **Establish Scope**   How large of an effort will this project require? Is it small scale or will it take a long time? Ensure that you know how much you’re willing to do and spend.   1. **Allocate Resources**   What resources do you have? What is the most important part of the project? Focus on what is the most important and divide your resources or budget accordingly.   1. **Set Timelines**   How long will this take? Are there dependent tasks? Know when things should and have to be finished.   1. **Identify Risks**   What could go wrong in our project? How do we prepare for them? Every project will have problems; this pre-planning can save a lot of time and worry in the future.   1. **Define Quality Standards**   By the end of this project and during, what is the standard we will hold our project/product to? What controls can we implement to ensure that standard?   1. **Develop Communication Plan**   How will the members of your project officially communicate? When will there be project updates/meetings?   1. **Create the Plan**   Document everything that you have decided as a guide from beginning to end.  **Tools**   * A Gannt Chart is a visual representation of the Project Timeline with task timelines and dependencies. * A Kanban Board is a board set up in an easy-to-see space. It is split into columns on what phase of work a task is in and will be moved right across columns as it is completed, allowing all workers an easy-to-understand representation of what is being done and where the project is at. |

### 1.4.5 What are the main task types in a project and how do they impact how a project is scheduled?

Include in your answer how a project determines the shortest project duration.

|  |
| --- |
| * **Tasks:** This is simply an item to be completed, they can be dependant, meaning the more dependent tasks, the longer timeframe and the less consecutive work can be done. * **Deliverables:** A deliverable is a task that ends with the production of something, like a document or prototype. A deliverable can be more likely to be dependent as other tasks may require the deliverable product. * **Milestones:** These are important tasks thatmark the completion of a phase in your project. A milestone task. They can track progress in a project and can mark moments where key decisions need to be made. * **Issues:** These are tasks that are added after the project plan, when an issue arises, a task is added to fix it. These tasks can put projects on pause and set back the timeline.   **Shortest Project Duration**  This can be calculated using a Critical Path Method. This is done by listing the tasks, their dependencies and projected durations. You then calculate 5 data points, earliest and latest start time, earliest and latest finish time and slack. These are used along with a formula to calculate the critical path forward to ensure that work is being done the fastest.  <https://knowledge.kantata.com/hc/en-us/articles/204775740-Task-Types>  [Critical Path Method (CPM) in Project Management](https://www.projectmanager.com/guides/critical-path-method) |

### 1.4.6 What are project control structures and why are they important?

Include in your answer at least two (2) types of control structures that are used in project management.

|  |
| --- |
| **Project Control Structures**  PCSs are frameworks with defined roles and responsibilities for a project’s various workers and tasks. It is a defined leadership and communication structure as well as the methods used to complete the project by leadership and employees. One such structure is a **Work Breakdown Structure**; this is a technique in project control that involves breaking down a task into smaller components and deliverables. It can be a great tool for defining the scope of work needed. Another structure is the **Responsibility Matrix, or a RACI Matrix**. This stands for Responsible, Accountable, Consulted and Informed and each member of a project’s team will be placed within the matrix. The columns of the matrix will have the task and the 4 RACI principles on them. Tasks can be read from left to right to see who corresponds to each principle per task. |

## 1.5 Client Meeting

In the meeting with the client, your recommendations can be approved or denied, in your summary include all approvals along with any recommendations that were made and denied assisting in the QA process.

Ensure that prior to the meeting a proposal document template has been completed, it must have sections for:

* Project overview with.
  + Stakeholders
  + Project name
  + Description of work
  + Start and End dates
* Background
* Deliverables
* Timeline

Once the document has been completed arrange a meeting with the client and attach a screenshot.

|  |  |
| --- | --- |
| **Item** | **Notes** |
|  |  |
|  |  |
|  |  |
| **Summary** | |
| ***Hint:*** *the client will be finalising their requirements after this meeting along with if they are going to sign a contract with CITEMS, in the summary include any limitations or areas to be further explored* | |
| **Checklist** | |
| * Project proposal document * Meeting screenshot | |

# Task 2: Design Phase

## 2.1 Requirements

After you have completed your meeting with the client you have received a new set of requirements for the application. The clients are expecting the functionality to increase over time, and they want to set the application up for easy maintenance for future iterations, therefore modularity is required for code reuse. All errors should be handled, and the user should not have any unexpected functionality.

**Main Window**

* Front end requirements
  + A data grid
  + Save and load menu options
  + A display of the average values
  + Inputs for the upper and lower bound of values
  + Icons to move between datasets
  + A visual indicator for acceptable, high, and low values
* Back-end requirements
  + A sorted dataset
  + Calculated average of the loaded data
  + A binary search function
  + The ability to move between loaded files

**Data Grid**

* A visual representation of the sensor data that is being read in; it will contain three (3) colours.
* Green represents a sensor value in a known good range.
* Red represents a sensor value that is above the known good range.
* Blue represents a sensor value that is below the known good range.
* Will represent the contents of a 2d array.
* An indication of values that fall above, below, or within the user defined values.

**File Management**

* Save and load binary (.bin) files,
* Files must be loaded into a collection that is passed by reference,
* (optional) save and load csv (.csv) files.

**Functionality**

* The data processing class must be accomplished with a singleton class
* Searching must be accomplished with a custom binary search to locate values based on a user given property of the data such as its label,
* average of the currently opened set of data

**Feedback System**

* A system for information to be given to the user when an error occurs.
* A system for information to be given to the user when an action occurs.

**Documentation**

* External documentation must be generated to allow for teams to interface with the modules developed.
* Internal documentation must be used to ensure clarity of the code for maintaining it into the future.

## 2.2 Data Structures and Variables

### 2.2.1 Knowledge of Programming Concepts and Data Structures

Research the listed concepts and structures and give a brief description of each, include in the description a use case or example for each.

|  |  |
| --- | --- |
| Concept | **Description** |
| Dynamic Variables | A dynamic variable is a variable whose address is determined when the program is run rather than a static variable which has memory reserved for it. A dynamic variable is referred to by pointers. |
| Modular programming | Modular Programming is an approach to programming where each responsibility is handled by a single smaller piece of codebase called a module. The module is encapsulated and can be reused and tested individually. |
| List | A list is a data structure that stores a mutable collection of objects or variables. Mutable means that the items within the list can be changed, reorder or added to, making them different from arrays which have a set length. |
| Array | An array is another kind of data structure that has a fixed size and is stored in one contiguous place within memory. It is indexed which allows for direct and efficient access to its elements. |
| Linked List | A linked list is a data structure that is not contiguous in memory. Instead, it consists of a series of connected nodes within memory that contain data and a pointer to the following node. This allows for dynamic memory allocation. |
| Stack | A stack is another data structure that follows a principle called Last In, First Out. Like a stack of dishes, the last element that was added to the structure will be the first one to be removed. Adding an element is called pushing and removing an element is called popping. |
| Queue | A queue is very similar to a stack but operates on the First In, First Out principle, meaning that it holds elements and ensures they are operated on in order. It is visualised with a rear or tail that adds elements and a front where items are dequeued and exited from. |
| Tree | A tree is a hierarchical data structure constructed of branching paths of nodes from a singular “root” node. Each node is connected to a parent node and can have a child node to classify data further. They can be used for file exploration and HTML hierarchy of webpages. |
| Graph | A graph data structure consisted of connected nodes, but it is non-linear and not hierarchical. Any node can be connected to multiple other nodes. These structures can be used to model complex relationships and are used in social networks and mapping. |
| Set | A set is a data structure that collects unique elements, not allowing for duplicate values. It can come in an ordered form or an unordered form, and this default varies from language to language. It is used for many mathematical applications and has many inbuilt operations for finding intersections and differences between it’s contained elements. |
| Hash Table | A hash table is a data structure that maps values to a key. A hash function is created that allows for the key to be fed into the function and return the index of the values within the hash. It is used when needing easy access to a specific data value within a large dataset. |
| Random Access Algorithms | Random access algorithms are a way of accessing data within data structures. They are significantly faster than sequential access as it can go directly to the needed value. A hash function is a classic example of a random-access algorithm. *(While random is in the name, these algorithms usually focus on targeted access rather than randomly accessing elements in a dataset.)* |
| User defined data structures | These are data structures that are not predefined but created by the programmer for their specific needs. There are some known examples, such as unions or structures. Unions are a way of saving a memory space to hold any type of data, but it can only hold one data type at a time. Structures are collections of various data types, similar to values within a class definition, however all values within a structure are public and structures have much smaller memory usage. |
| **References** | <https://www.webopedia.com/definitions/dynamic-variable/>  <https://www.dhiwise.com/post/modular-programming>  <https://www.futurelearn.com/info/courses/introduction-to-programming-with-python-fourth-rev-/0/steps/265763>  <https://www.tutorialspoint.com/data_structures_algorithms/array_data_structure.htm>  <https://www.geeksforgeeks.org/c/linked-list-in-c/>  <https://www.geeksforgeeks.org/dsa/stack-data-structure/>  <https://www.geeksforgeeks.org/dsa/what-is-queue-data-structure/>  <https://www.geeksforgeeks.org/dsa/applications-advantages-and-disadvantages-of-binary-tree/>  <https://unstop.com/blog/graph-in-data-structure>  <https://www.geeksforgeeks.org/dsa/introduction-to-set-data-structure/#properties-of-set-data-structure>  <https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/>  <https://fiveable.me/key-terms/data-structures/random-access>  <https://www.bartleby.com/subject/engineering/computer-science/concepts/user-defined-datatype> |

### 2.2.2 Data Structures in Use

There are several data structures that can be used for this project, research the following and recommend two (2) data structures that you will use in this project and why you recommend those data structures along with two (2) pros and two (2) cons for each data structure. Make sure to include how you will be using them within the project.

|  |
| --- |
| **Data Structure Recommendations** |
| ***Hint:*** *explain how they relate to the requirements currently expected*  **List/Collection**  The requirements specify the need for a Collection, which is a type of list that can have its functions overridden with custom logic. This works well with the need for a custom binary search.  Pros  **2D Array** |

### 2.2.3 Input and Output Parameters in Software Development

In your answers ensure that you address both input and output parameters.

|  |
| --- |
| **What are input and output parameters?** |
|  |
| **Why are they important?** |
|  |

## 2.3 Diagrams

|  |
| --- |
| **Binary Search Algorithm Flowchart** |
| HH |
| **Class Diagram** |
| ***Hint****: Please provide your class in UML* |

## 2.4 Programming Methodologies

What are programming methodologies and describe at least two (2) along with the main principles of one (1).

|  |  |
| --- | --- |
| **1** |  |
| **2** |  |

## 2.5 Maintenance and Language Facilities

### 2.5.1 What are some key aspects of software maintenance?

Provide at least three (3) aspects and include how programming languages support these aspects with built in facilities such as garbage collection releasing memory for the application.

|  |  |
| --- | --- |
| **1** |  |
| **2** |  |
| **3** |  |

### 2.5.2 What documentation tool and techniques can be used in software development?

Include at least two (2) techniques and one (1) tool that can be used.

|  |  |
| --- | --- |
| **1** |  |
| **2** |  |
| **3** |  |

## 2.6 Development Meeting

As with the client meeting, this is where your recommendations must be defended and finalised prior to beginning work on the application. The project plan document must be completed prior to the meeting being held to allow for the approver to understand the planned application and approach.

|  |  |  |
| --- | --- | --- |
| **Item** | | **Notes** |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
| **Summary** | | |
| ***Hint:*** *this is the final review prior to development commencing, this should be undertaken prior to any code being created.* | | |
| **Approved By:** |  | | |
| **Checklist** | * Project plan | | |

# Task 3: Development Phase

## 3.1 Code

Once you have been given approval you are able to commence work on the application, any changes to the design of the application must be noted during the development process to ensure that documentation can be updated to reflect the current state of the application when it is delivered and the client can be presented with a lot of changes along with reasoning behind the changes.

## 3.2 Development adjustments

Please list any changes from the initial design to the application here, this can be populated with git commits.

|  |  |
| --- | --- |
| **Change** **list** | **Justification** |
|  |  | |
|  |  | |

# Task 4: Testing Phase

## 4.1 Testing Plan

Thorough testing must be completed prior to handover to the client, this must be broken up into units of work. Adjust the below test plan to suit final application, this application must test the possible user interactions along with validating that the functions are working for edge cases.

* Main window testing must provide screenshots for actual results to validate the pass.
* Function testing must provide trace code screenshots to validate the pass.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Main Window** | | | | | |
| **#** | **Test** | **Data** | **Expected** | **Actual** | **Result** |
| 1 | Details of test | Data for test | Expected results | Actual Result | P |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Functions** | | | | | |
| **#** | **Test** | **Data** | **Expected** | **Actual** | **Result** |
| 1 | Details of test | Data for test | Expected results | Actual Result | P |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## 4.2 Debugging Tools and Methodologies

List at least three (3) debugging methods or tools that can be used to assist in the testing of a software application.

|  |  |
| --- | --- |
| **1** |  |
| **2** |  |
| **3** |  |

## 4.3 Testing Procedures

List at least three (3) testing procedures that can be used within software development.

|  |  |
| --- | --- |
| **1** |  |
| **2** |  |
| **3** |  |

# Task 5: Review Phase

## 5.1 Handover

With testing complete the client needs to have the application demonstrated prior to submission, they are curious to possible extensions discovered during the development process for the MVP and would like feedback on the limitations and restrictions currently present with the application.

The client would also like external documentation to be generated and presented for review. Documentation should take the form of a report that contains:

* Application name,
* Introduction outlining the intent of the application,
* SDLC used,
* Requirements,
* Any features implemented,
* How to guide for the features,
* Commit history.

|  |  |  |
| --- | --- | --- |
| **Lessons Learned** | | |
|  | | |
|  | | |
| **Limitation** | | **Notes** |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |
| **Recommendation** | |  |
|  | |  |
|  | |  |
|  | |  |
| **Feedback** | | |
| ***Hint:*** *this should be capable of being implemented within the next two (2) development cycles* | | |
| **Presented to:** |  | |
| **Checklist** | * Zipped source code, * Internal documentation, * Screenshot of client meeting email, * Handover documentation * Scope of work, * Project plan, * This document. | |

End of Assessment Task One