**GoldFolks**

**Project Plan**

**Version 1.0 approved**

Anil Ankitha,

Chan Shao Jing,

Chong Yow Lim,

Lionel Wong Zhi Neng,

Low Jin Teng Jackson,

Ng Chi Hui,

Zachary Varella Lee Zheyu

School of Computer Science and Engineering,

Nanyang Technological University

**4th Oct 2021**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision Number** | **Date** | **Primary Author(s)** | **Comments** |
| 1.0 | 18th Sept 2021 | Anil Ankitha,  Chan Shao Jing,  Chong Yow Lim,  Lionel Wong Zhi Neng,  Low Jin Teng Jackson,  Ng Chi Hui,  Zachary Varella Lee Zheyu | Initial Project Plan Draft |
| 2.0 | 25th Sept 2021 | Anil Ankitha,  Chan Shao Jing,  Chong Yow Lim,  Lionel Wong Zhi Neng,  Low Jin Teng Jackson,  Ng Chi Hui,  Zachary Varella Lee Zheyu | Final Project Plan |

**Table of Contents**

Revision History.................................................................................................................................................1

Table of Contents ..............................................................................................................................................2

1 Introduction...............................................................................................................................................3

1.1 Project Overview ................................................................................................................................3

1.2 Project Description and Scope .........................................................................................................3

2 Project Organization ................................................................................................................................4

2.1 Team Structure ....................................................................................................................................4

2.2 Roles and Responsibilities .................................................................................................................4

2.3 Team Communication .......................................................................................................................5

3 Process Definition ....................................................................................................................................6

3.1 Lifecycle Model ...................................................................................................................................6

4 Schedule .....................................................................................................................................................7

4.1 Activity Dependencies and Schedule ...............................................................................................7

4.2 Work Breakdown Structure...............................................................................................................8

4.3 Work Packages ..................................................................................................................................9

4.4 Activity Dependencies .....................................................................................................................9

4.5 Work Package Details ......................................................................................................................10

5 Project Estimates ....................................................................................................................................14

5.1 Code Size Estimation using Function Points ...............................................................................14

5.1.1 Unadjusted Function Points ....................................................................................................14

5.1.2 Adjusted Function Points ........................................................................................................16

5.1.3 Lines of Code.............................................................................................................................17

5.2 Efforts, Duration and Team Size Estimation...............................................................................17

5.2.1 Distribution of Effort...............................................................................................................17

5.3 Cost Estimates...................................................................................................................................18

6 Product Checklist....................................................................................................................................19

7 Best Practice Checklist ...........................................................................................................................20

8 Risk Management....................................................................................................................................21

9 Quality Assurance ...................................................................................................................................22

10 Monitoring & Control ............................................................................................................................23

**1 Introduction**

**1.1 Project Overview**

GoldFolks is an all-in-one mobile application designed to be a companion to help the elderly age healthily and gracefully. It helps to remind the elderly of their medication needs, maintain and improve their cognitive ability through brain teasers and physical exercise.

**1.2 Project Description and Scope**

GoldFolks is an all-in-one, user-friendly application that targets the elderly. It aims to reduce the current problems faced by the elderly today where they must install multiple applications and switch between them back and forth, which can be difficult and confusing for the less technologically savvy elderly.

The basis of the all-in-one mobile-application platform takes several main features as stated below:

* + Users would be able to log into the application
  + Users can create, edit and delete medication reminders.
  + Once the medication reminders have been created, a notification shall be sent to the user at the stipulated time.
  + Users can play cognitive games such as Mental Math and Simon Say.
  + Users can watch exercise videos filtered by categories such as full-body, abs, legs or low-impact workouts.

Due to the nature of the system and budget constraints, GoldFolks app will take several possible constraints into account.

* + The application will be mobile based as it is the most frequently used platform for elderly. Development for web applications could start when the app for iOS and Android platforms meets expectations.
  + Medication is assumed to be taken daily for the purpose of medication reminders.

**2 Project Organization**

**2.1 Team Structure**

The following is the list of executive roles, as required by CMM level 3.

* Project Manager: Chan Shao Jing
* Lead Developer: Zachary Varella Lee Zheyu
* Release Engineer: Ng Chi Hui
* Front-end Developer: Anil Ankitha
* Back-end Developer: Chong Yow Lim
* QA Manager: Low Jin Teng Jackson
* QA Engineer: Lionel Wong Zhi Neng

**2.2 Roles and Responsibilities**

**Project Manager: Chan Shao Jing**

* Oversees project progress
* Coordinates project meetings
* Approves and executes project plan
* Assigns tasks and reports status of project to team members
* Manages and motivates team members
* Represents the team during external meetings with clients/external parties
* Tracks and monitors project progress
* Ensures that the project is delivered on time and within the budget
* Periodic review of the project
* Evaluates the outcomes of the project established

**Lead Developer: Zachary Varella Lee Zheyu**

* Responsible for technical aspects of project release
* Directs the software development team in the design, development, coding, testing and debugging of applications
* Writing testable, scalable and efficient code and leading code reviews
* Mentoring junior team members and ensuring they adhere to software quality standards
* Ensures the use of good software design principles
* Integrates each software module into the entire software system
* Translates logical design into detailed design
* Planning, leading and documenting technical specifications for features or system design
* Work closely with senior stakeholders to understand business requirements and help translate these into technical requirements for the development team

**Release Engineer: Ng Chi Hui**

* Plans release windows and cycles
* Implements and manages release processes
* Manages risks and resolves issues that affect release scope and release schedule
* Measure and Monitor development progress to ensure that releases are delivered within budget and requirements
* Communicate all key project plans, commitments and changes including requirements

**Front-end Developer: Anil Ankitha**

* Participate in the entire SDLC, generating work products including documentation, source code, unit and integration tests
* Work with key stakeholders in creating prototypes, mockups and staging site
* Ensures the technical feasibility of the designs
* Ensures that key design principles are followed
* Test and ensure the integration of coded modules into functioning system
* Translates UI/UX design wireframes to code that will product visual elements in application
* Works closely with and assists lead developer

**Back-end Developer: Chong Yow Lim**

* Participate in the entire SDLC, generating work products including documentation, source code, unit and integration test
* Collaborate with Front-end developers to integrate user-facing elements with server-side logic
* Implements data security and protection
* Design and develop data storage solutions
* Creates Application Programming Interface (API) for the server-side logic
* Works closely with and assists lead developer

**QA Manager: Low Jin Teng Jackson**

* Develops quality assurance plans
* Ensure software quality personnel resource level
* Creates and manages test plans
* Designs testing strategies
* Assist software quality personnel in resolving software quality issues

**QA Engineer: Lionel Wong Zhi Neng**

* Ensures acceptable software quality
* Designs testing strategies
* Conduct product assessments
* Assists the QA Manager in creating and managing test plans
* Verify software requirements
* Executes test procedures

**2.3 Team Communication**

Communication is critical to the development of the project. GoldFolks main channels of communication include:

* Weekly meetings are held on Saturday at 2pm, to update on group progress and discussions.
* All documents are regularly updated on Teams Channel and Wiki page.
* Source codes and issues are tracked with Trello and GitHub.
* Group announcements and updates are sent through WhatsApp.
* Split up into subgroups as necessary, in order to work more co-operatively on specific problems.

**3 Process Definition**

**3.1 Lifecycle Model**

The **Agile** development methodology approach has been selected to develop the GoldFolks application. The main idea of Agile is to develop the program with small incremental releases in iterations by breaking down a bigger problem in multiple sub-problems. The product is thus delivered to the customer in an incremental manner. Agile also provides a constant feedback mechanism while the product is delivered. Lastly, in the Agile approach testing is integrated with the project lifecycle which begins very early in the software development process. This frequent testing allows more time and less effort to resolve the bugs.

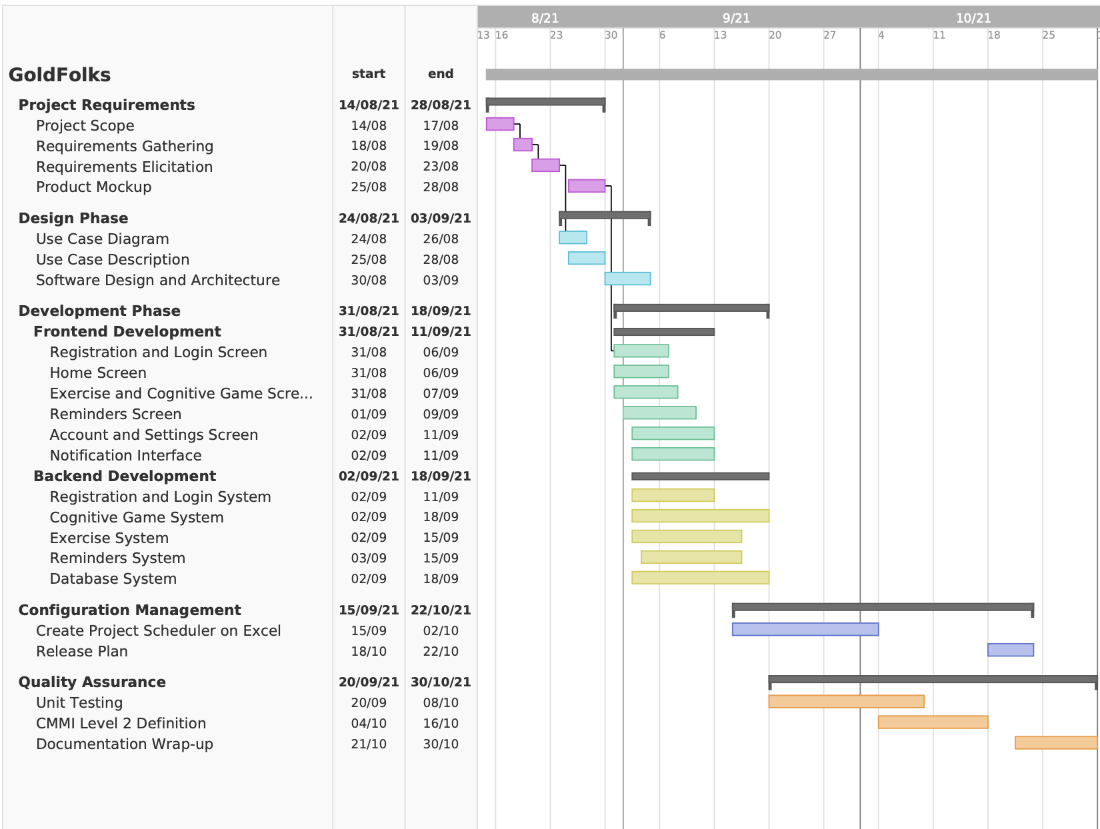
The Agile development approach is selected due to the short time frame given for the official release. This method will allow for changes to be made within the duration of the project and this is especially important. Even when most requirements are well-defined at the initial stages of the project, functionalities addition and changes are not uncommon. Moreover, due to the independent nature of our individual functionalities, it would be more efficient for us to split the development of the functionality for sprint planning.

Compared to the traditional waterfall model, the agile approach is thus a much more flexible model to follow.

**4 Schedule**

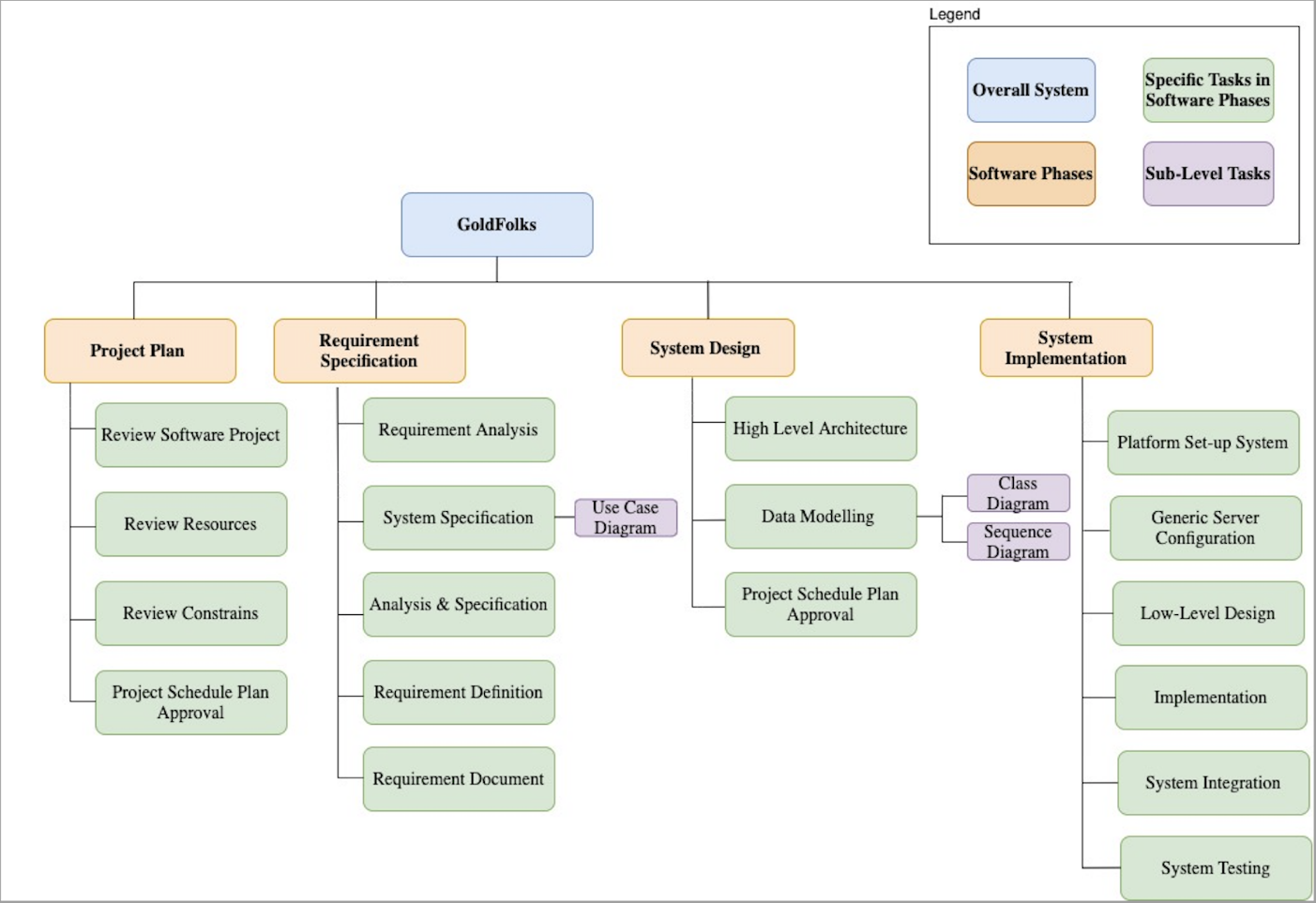
**4.1 Activity Dependencies and Schedule**

The following project scheduling timeline has been delivered in a Gantt Chart below:



*Figure 1. Gantt Chart for Project Deliverables*

**4.2 Work Breakdown Structure**



*Figure 2. Work Breakdown Structure*

**4.3 Work Packages**

The entire project work is broken down by the important phases of the software development life cycle. They include the following:

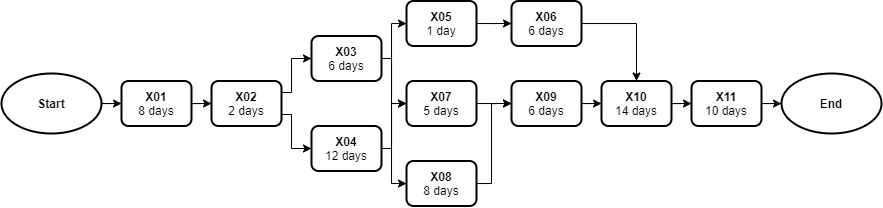
1. Project Proposal
2. Requirement Elicitation
3. Software Quality Assurance
4. System Requirement Specification
5. Risk Management
6. Project Plan
7. User Interface Design
8. Technical Architecture
9. Data Modeling
10. Coding & Unit Testing
11. Integration & System Testing

**4.4 Activity Dependencies**

The following table describes the dependencies of the deliverable work packages:

|  |  |  |  |
| --- | --- | --- | --- |
| Work  Package # | Work Package Description | Duration | Dependencies |
| X01 | Project Proposal | 8 days | - |
| X02 | Requirements Elicitation | 2 days | X01 |
| X03 | Software Quality Assurance | 6 days | X02 |
| X04 | System Requirement Specification | 12 days | X02 |
| X05 | Risk Management | 1 day | X03, X04 |
| X06 | Project Plan | 6 days | X03, X04, X05 |
| X07 | User Interface Design | 5 days | X03, X04 |
| X08 | Technical Architecture | 8 days | X03, X04 |
| X09 | Data Modelling | 6 days | X08 |
| X10 | Coding & Unit Testing | 14 days | X06, X07, X09 |
| X11 | Integration & System Testing | 10 days | X10 |

The following Activity Network Diagram describes the above in more graphical detail:



**Critical Path Analysis:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Work Package** | **Earliest Start** | **Earliest Finish** | **Latest Start** | **Latest Finish** | **Slack Time** |
| X01 | Start of day 0 | End of day 7 | Start of day 0 | End of day 7 | 0 days |
| X02 | Start of day 8 | End of day 9 | Start of day 8 | End of day 9 | 0 days |
| X03 | Start of day 10 | End of day 15 | Start of day 23 | End of day 28 | 13 days |
| X04 | Start of day 10 | End of day 21 | Start of day 10 | End of day 21 | 0 days |
| X05 | Start of day 22 | End of day 22 | Start of day 29 | End of day 29 | 7 days |
| X06 | Start of day 23 | End of day 28 | Start of day 30 | End of day 35 | 7 days |
| X07 | Start of day 22 | End of day 26 | Start of day 25 | End of day 29 | 3 days |
| X08 | Start of day 22 | End of day 29 | Start of day 22 | End of day 29 | 0 days |
| X09 | Start of day 30 | End of day 35 | Start of day 30 | End of day 35 | 0 days |
| X10 | Start of day 36 | End of day 49 | Start of day 36 | End of day 49 | 0 days |
| X11 | Start of day 50 | End of day 59 | Start of day 50 | End of day 59 | 0 days |

Critical Path: Start → X01 → X02 → X04 → X08 → X09 → X10 → X11 → End

**4.5 Work Package Details**

Work packages are listed below. A team member, indicated in bold, has been assigned as primarily

responsible for each work package and will coordinate that package.

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X01 - Project Proposal |
| **Assigned To** | All members |
| **Effort** | 8 Person Days |
| **Start Date** | 16-08-2021 |
| **Purpose** | To define project objectives, identify project scope, determine target specifications. |
| **Inputs** | Problem identified, some possible solutions and requirements from users. |
| **Activities** | 1. Draft problem statement. 2. Development of Business case. 3. Draft design objectives, based on the selected solution. 4. Define Technical approaches of the solutions. 5. Establish brief project management. |
| **Outputs** | Project Proposal |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X02 – Requirements Elicitation |
| **Assigned To** | Zachary, Ankitha |
| **Effort** | 2 Person Days |
| **Start Date** | 24-08-2021 |
| **Purpose** | To gather requirements that can help further refine the details of the project, various requirement elicitation techniques will be used |
| **Inputs** | Customer requirements, all outputs from work package Project Proposal (X01) |
| **Activities** | 1. Holding formal interviews with the customer to understand the requirements of the software. 2. Brainstorming on the possible requirements that might be useful for the customer. 3. Building a prototype to gather more requirements and build a common understanding of the requirements of the customer. |
| **Outputs** | A list of requirements gathered from interviews, brainstorming, prototyping and observations |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X03 - Software Quality Assurance |
| **Assigned To** | Lionel, Jackson |
| **Effort** | 6 Person Days |
| **Start Date** | 26-08-2021 |
| **Purpose** | To establish the goals, processes, and responsibilities required to implement effective quality assurance functions for the GoldFolks project. To ensure a project complies with standard and of a certain quality. |
| **Inputs** | All outputs from work package Requirement Elicitation(X02) |
| **Activities** | 1. Establish project purpose and scope. 2. Identify management organizational structure, its roles and responsibilities, and the software quality tasks to be performed. 3. Define documentation standards that the project shall follow. 4. Establish tools and methodologies on how software quality can be upkeep with. |
| **Outputs** | Software Quality Assurance |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X04 - System Requirement Specifications |
| **Assigned To** | Chi Hui, Shao Jing, Yow Lim |
| **Effort** | 12 Person Days |
| **Start Date** | 26-08-2021 |
| **Purpose** | To establish a common understanding between the customer and the software project team of the customers’ requirements to be addressed by the project. Information gathered from requirement elicitation will be formally documented. |
| **Inputs** | All outputs from work package Requirement Elicitation (X02) |
| **Activities** | 1. Document problem statement, background, and overall description 2. Identify the constraints of the project 3. With elicited requirements, draft operational, functional, input, process, output, hardware, software, and deployment requirements. |
| **Outputs** | A written document of the System Requirements Specification |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X05 - Risk Management Plan |
| **Assigned To** | Lionel, Jackson |
| **Effort** | 1 Person Days |
| **Start Date** | 07-09-2021 |
| **Purpose** | To identify and analyze all risk (positive or negative) associated with the project. The plan will outline how these risks would be mitigated and how risk management activities will be performed, recorded, and monitored |
| **Inputs** | All outputs from work package Software Quality Assurance (X03), System Requirements Specification Details (X04) |
| **Activities** | 1. Identify and analyze the risk associated with the cost, resource scheduling, and technical aspect. 2. Determine methodologies to mitigate the risks identified. |
| **Outputs** | A written document of Risk Management Plan |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X06- Project Plan |
| **Assigned To** | Chi Hui, Shao Jing, Yow Lim |
| **Effort** | 6 Person Days |
| **Start Date** | 08-09-2021 |
| **Purpose** | To define the overview of the project process in the document and give a project schedule. |
| **Inputs** | All outputs from work package Software Quality Assurance (X03), System Requirements Specification Details (X04) and Risk Management(X05) |
| **Activities** | This work package includes:   1. Define project overview and scope 2. Identify the project team organization 3. Model after project processes 4. Plan project schedule 5. Estimate project costs 6. Identify and allocate available resources by estimating effort, duration, team size 7. Define product checklist 8. Identify best practices 9. Brief introduction to risk management 10. Reiterate quality assurance 11. Define monitoring and control methodologies |
| **Outputs** | A written document of the Project Plan |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X07 - User Interface Design |
| **Assigned To** | Zachary, Ankitha |
| **Effort** | 5 Person Days |
| **Start Date** | 07-09-2021 |
| **Purpose** | To create an effective communication medium between the system and the customer. The initial UI prototype will serve as a reference point of how the system product will look like and how can the customers interact with the product. This will help to mitigate confusion or disagreement. |
| **Inputs** | All outputs from work package Software Quality Assurance (X03), System Requirements Specification Details (X04) |
| **Activities** | 1. Referencing the Customer Requirements and System Requirements Specification Details, draft an initial Hi-Fidelity prototype design. 2. Seek opinions from different stakeholders and further refine the user interface design with design principles and Shneiderman’s Eight Golden Rules 3. Repeat Step 1 and 2 until the user interface design is coherent with the requirements. 4. Build the user interface (screen layouts only) as a minimum viable product. |
| **Outputs** | Working Prototype of User Interface (front end only) |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X08 - Technical Architecture |
| **Assigned To** | All Members |
| **Effort** | 8 Person Days |
| **Start Date** | 13-09-2021 |
| **Purpose** | To perform high level architectural design, developing an overview of how each subcomponent in the system interacts with each other. |
| **Inputs** | All outputs from work package Software Quality Assurance (X03), System Requirements Specification Details (X04) |
| **Activities** | 1. High-level design entails defining the architecture of the software system and 2. Defining the various components required for the system and how they are interrelated to each other. 3. Deciding on the software and hardware infrastructures, such as what operating system on which the software is built, the language used to implement the software, and so on. Designers must consider topics including maintainability, portability, and reusability will be addressed here as well. |
| **Outputs** | High-level design and Architectural specification |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X09 - Data Modelling |
| **Assigned To** | All Members |
| **Effort** | 6 Person Days |
| **Start Date** | 15-09-2021 |
| **Purpose** | To develop the data model which forms the basis for the project database. |
| **Inputs** | All outputs from work package Technical Architecture (X08) |
| **Activities** | 1. Perform analysis on data flow and entity relationships. 2. Consider data types and tables that are required to represent stored data in the system. |
| **Outputs** | A written document for data modelling |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X10 - Coding & Unit Testing |
| **Assigned To** | All Members |
| **Effort** | 14 Person Days |
| **Start Date** | 21-09-2021 |
| **Purpose** | To implement the system as per the Requirement Specifications, system Architecture and UI design. This work package includes such additional activities as preliminary unit testing to test out the various quality assurance testing factors such as optimality and reliability for individual components before integration. |
| **Inputs** | All outputs from work package Project Plan(X06), User Interface Design (X07), Data Modelling (X09) |
| **Activities** | 1. Programmers will implement the modules according to the design specifications noted in the System Requirement Specifications. 2. Programmers shall carry out unit test plans. |
| **Outputs** | Source code, header files, integrated system test plan, and test cases |

|  |  |
| --- | --- |
| **Project** | GoldFolks |
| **Work Package** | X11 - Integration & System Testing |
| **Assigned To** | All Members |
| **Effort** | 10 Person Days |
| **Start Date** | 05-10-2021 |
| **Purpose** | To integrate the various components into the system and build the data flow between them. To identify and fix logical and syntactical errors produced during the implementation of the System and setting up drivers and stubs to see how the module responds to various inputs. Black box testing, as well as white box testing, might be conducted to check for logical errors. All the testing procedures will be documented in the Test Plan report. If problems are found, they will be noted and fixed at the earliest possible time. |
| **Inputs** | All outputs from work package Coding & Unit Testing (X10). |
| **Activities** | 1. The Integration testing team may try to simulate how a user might interact with the system. 2. Development of stubs and drivers as well, but here this is more geared towards the higher (overall system) level. 3. Testers will also examine issues such as system performance and integrity. Heuristics assessment plays an important role in this work package, as intelligence components will define eventual system success. |
| **Outputs** | A written document of Requirement Test Coverage Report. |

**5 Project Estimates**

**5.1 Code Size Estimation using Function Points**

We calculated unadjusted function point based on the complexity of functions provided by this

system. Code size is then estimated by adjusted function point.

**5.1.1 Unadjusted Function Points**

GoldFolks supports the following proposed functions:

User:

* Log into the application
* Register a new account
* Request a new password
* Create/edit/delete medication reminders
* Select exercise videos to watch
* Play Mental Math or Simon Says

The measure of unadjusted function points is based on five primary component elements of these functions: Inputs, Outputs, Inquiries, Logical Files, and Interfaces. Each element ranges from Low Complexity, Medium Complexity to High Complexity. The detailed evaluation of the complexity is as follows:

**Rating Inputs:**

* Multiple choice input for Mental Math game
* Pattern selection for Simon Says
* Medication reminder details (creating/editing/deleting medication reminders)
* Retrieving videos from YouTube API
* Retrieving existing medication reminder data
* Account registration

|  |  |  |  |
| --- | --- | --- | --- |
| **Files Type Referenced (FTR)** | **Data Elements** | | |
|  | 1-4 | 5-15 | Greater than 15 |
| Less than 2 | Low (3) | Low (3) | Average (4) |
| 2 | Low (3) | Average (4) | High (6) |
| Greater than 2 | Average (4) | High (6) | High (6) |

**Rating Outputs:**

* Displaying medication reminder notifications
* Displaying game tutorials
* Displaying selected exercise videos
* Display game selection outcome (correct or wrong answer)
* Display game scores

|  |  |  |  |
| --- | --- | --- | --- |
| **File Types Referenced (FTR)** | **Data Elements** | | |
|  | 1-5 | 6-19 | Greater than 19 |
| Less than 2 | Low (4) | Low (4) | Average (5) |
| 2 or 3 | Low (4) | Average (5) | High (7) |
| Greater than 3 | Average (5) | High (7) | High (7) |

**Rating Inquiries:**

* Login
* Forgot password

|  |  |  |  |
| --- | --- | --- | --- |
| **File Types Referenced (FTR)** | **Data Elements** | | |
|  | 1-5 | 6-19 | Greater than 19 |
| less than 2 | Low (3) | Low (3) | Average (4) |
| 2 or 3 | Low (3) | Average (4) | High (6) |
| Greater than 3 | Average (4) | High (6) | High (6) |

**Rating Logical Files:**

* Medication reminder data (offline database)

|  |  |  |  |
| --- | --- | --- | --- |
| **Record Element Types (RET)** | **Data Elements** | | |
|  | 1 to 19 | 20 - 50 | 51 or More |
| 1 RET | Low (7) | Low (7) | Average (10) |
| 2 to 5 RET | Low (7) | Average (10) | High (15) |
| 6 or More RET | Average (10) | High (15) | High (15) |

**Rating Interfaces:**

* YouTube
* Cloud (User account information, game scores, medication reminders)

|  |  |  |  |
| --- | --- | --- | --- |
| **Record Element Types (RET)** | **Data Elements** | | |
|  | 1 to 19 | 20 - 50 | 51 or More |
| 1 RET | Low (7) | Low (7) | Average (10) |
| 2 to 5 RET | Low (7) | Average (10) | High (15) |
| 6 or More RET | Average (10) | High (15) | High (15) |

**Summary of above analysis:**

|  |  |  |
| --- | --- | --- |
| **Element** | **Complexity** | **Detail** |
| Inputs | Low | Multiple choice input for Mental Math game |
| Low | Pattern selection for Simon Says |
| Average | Medication reminder details |
| Low | Retrieving videos from YouTube API |
| Low | Retrieving existing medication reminder data |
| Low | Account registration |
| Logical Files | Average | Medication reminder data (offline) |
| Outputs | Average | Display medication reminder notifications |
| Low | Display game tutorials |
| Low | Display exercise video |
| Low | Display game selection outcome |
| Low | Display game score |
| Inquiries | Low | Login |
| Low | Forgot password |
| Interfaces | Low | YouTube |
| Low | User account information |
| Low | Game scores |
| Average | Medication reminder data (online) |

Calculation of Unadjusted Function Points:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Characteristic** | **Low** | | **Medium** | | **High** | |
| Inputs | 5 | × 3 | 1 | × 4 | 0 | × 6 |
| Outputs | 4 | × 4 | 1 | × 5 | 0 | × 7 |
| Inquiries | 2 | × 3 | 0 | × 4 | 0 | × 6 |
| Logical Files | 0 | × 7 | 1 | × 10 | 0 | × 15 |
| Interfaces | 3 | × 5 | 1 | × 7 | 0 | × 10 |
| **Unadjusted FP** | 52 |  | 26 |  | 0 |  |
| **Total=L+M+H** | 78 | | | | | |

**5.1.2. Adjusted Function Points**

|  |  |  |
| --- | --- | --- |
| **Influence Factors** | **Score** | **Detail** |
| Data Communications | 5 | Application is more than a front-end, and supports more  than one type of teleprocessing communications protocol. |
| Distributed Functions | 4 | Distributed processing and data transfer are online and in  both directions. |
| Performance | 4 | Response time or throughput is critical during all business  hours. No special design for CPU utilization was required. Processing deadline requirements with interfacing systems are constraining. |
| Heavily used | 3 | Some security or timing considerations are included. |
| Transaction rate | 0 | Daily peak transaction period is anticipated. |
| Online data entry | 5 | More than 30% of transactions are interactive data entry |
| End-user efficiency | 4 | Four to five of the efficiency designs are included |
| On-line data update | 3 | Online update of major internal logical files is included. |
| Complex processing | 0 | Any one of the complex components |
| Reusability | 4 | The application was specifically packaged and/or  documented to ease re-use, and the application is customized by the user at source code level. |
| Installation Ease | 3 | No special considerations were stated by the user *but* special  setup is required for installation. |
| Operational Ease | 4 | Effective start-up, back-up, and recovery processes were  provided, but no operator intervention is required (count as two items). |
| Multiple sites | 0 | User requirements do not require considering the needs of  more than one user/installation site. |
| Facilitate change | 0 | Flexible query and report facility is provided that can handle  complex requests, for example, *and/or* logic combinations on one or more internal logical files (count as three items). |
| Total score | 39 | |
| **Influence Multiplier**  = Total score × 0.01 + 0.65 = 39 × 0.01 + 0.65 = 1.04 | | |
| **Adjusted FP**  = Unadjusted FP × Influence Multiplier = 78 × 1.04 = 81.12 | | |

|  |
| --- |
| **Scoring (0 – 5)** |
| 0 = No influence |
| 1 = Insignificant influence |
| 2 = Moderate influence |
| 3 = Average influence |
| 4 = Significant influence |
| 5 = Strong influence |

**5.1.3 Lines of Code**

GoldFolks is implemented using Flutter, which is written in Dart. According to Capers Jones statistics, the average number of source lines per Function Point (FP) for Dart is 47.41.

Therefore, we have: **Lines of Code** = 81.12 FP × 53 LOC/FP = **3846 LOC.**

**5.2 Efforts, Duration and Team Size Estimation**

To estimate the effort and duration required for the project, we use function points as the basis to calculate Effort, Duration, Team Size and finally the schedule. The estimates are expanded to account for project management and extra contingency time to obtain the total average effort estimates. From these averages, the duration of each work package in working days is estimated based on the following calculations.

• All members are expected to work 7 days a week.

• Effort = Size / Production Rate = (3846 LOC) / (31 LOC/PD)[[1]](#footnote-2) = 124.065 PD

• Duration = 3 × (Effort)1/3 = 3 × (124.065)1/3 = 15 Days

• Initial schedule = 15 Days / 7 days a week = 2.14 Weeks

• Team size = 124.065 PD / 15 D = 9 Persons

• Working hours include 4 hours in a working day.

• Total person-hours (PH) = 124.065 PD × 4 hours = 496.26 PH

**5.2.1 Distribution of Effort**

|  |  |  |  |
| --- | --- | --- | --- |
| **1990’s Industry Data** | **Work Package** | **Distribution** | **Estimates** |
| Preliminary Design 18% | Project Proposal | 4% | 19.85 |
| Requirement Elicitation | 1% | 4.96 |
| Software Quality Assurance | 3% | 14.89 |
| System Requirement Specification | 6% | 29.78 |
| Risk Management | 1% | 4.96 |
| Project Plan | 3% | 14.89 |
| Detailed Design 25% | User Interface Design | 7% | 34.74 |
| Technical Architecture | 11% | 54.59 |
| Data Modelling | 7% | 34.74 |
| Code & Unit Testing 26% | Coding & Unit Testing | 26% | 129.03 |
| Integration & Test 31% | Integration & System Testing | 31% | 153.84 |
|  | **Extrapolated Total Effort:** |  | **496.26** |
|  | 2% for project management |  | 9.93 |
|  | 3% for contingency |  | 14.89 |
|  | **Total Effort:** |  | **521.08** |

These duration estimates assume that each team member works an equal amount on any given work package.

**5.3 Cost Estimates**

**Hardware:**

**Developer workstations:**

|  |  |
| --- | --- |
| **7** - Dell Precision Workstation 330 | Total  $0.00 |
| Pentium IV 1.4GHz single processor |
| 256 MB RAM |
| 20.8GB IDE drive |

**Software:**

**Development Tools:**

|  |  |
| --- | --- |
| Firebase | $0.00 |
| Flutter | $0.00 |

**Documentation/version control tools:**

|  |  |
| --- | --- |
| GitHub | $0.00 |
| Tortoise SVN | $0.00 |
| Diagrams.net | $0.00 |
| Visual Paradigm | $0.00 |
| Figma | $0.00 |
| Android Studio | $0.00 |
| Trello | $0.00 |
| Team Gantt | $0.00 |
| MediaWiki | $0.00 |

**Software License Provided by Third Party:**

|  |  |
| --- | --- |
| Microsoft Office | $0.00 |

**Other Resources:**

**Staff:**

|  |  |
| --- | --- |
| 9 Employees with 1164.45 working hours with $18.00/hour | $20,960.10 |

**Stationery:**

|  |  |
| --- | --- |
| Paper, photocopying and other miscellaneous costs | $50.00 |

**Total:** $21,010.10

The mobile application will be run on the user’s mobile device. GoldFolks is not responsible in any way for supplying said device, nor is it responsible for any issues that arise due to the fault of the user’s device. GoldFolks’ responsibility is only to develop the mobile application to satisfy all stated functional and system requirements. Hence, any costs incurred due to faults of the user’s device are to be borne by the user.

**6 Product Checklist**

The plan is that the items listed below will be delivered by the stated deadlines.

|  |  |
| --- | --- |
| **Project Deliverable** | **Estimated Deadline** |
| Meeting Minutes | Every meeting |
| Trello Backlog | Updated after every lab |
| Team Registration | 16th August 2021 (Lab 1) |
| Team Information | 30th August 2021 (Lab 2) |
| Project Proposal |
| Use Case Model & Description |
| System Requirement Specification | 13th September 2021 (Lab 3) |
| Quality Plan |
| Project Plan | 4th October 2021 (Lab 4) |
| Risk Management Plan |
| Prototype |
| Design Report on Software Maintainability | 18th October 2021 (Lab 5) |
| Configuration Management Plan |
| Change Management Plan |
| Release Plan |
| Presentation Slides | 1st November 2021 (Final Submission) |
| Test Plan |
| Test Cases and Requirements Test Coverage Report |
| CMMI Level 2 Definition |
| Peer Review Report |

**7 Best Practices**

Throughout our project’s planning and development phases, we will be following these best practices to ensure quality and successful completion of our deliverables.

|  |
| --- |
| **Best Practices** |
| Document what we do; all documentation must be in a standardized format. |
| Pay attention to requirements, check for ambiguity, completeness, accuracy, and consistency. The requirement documentation must contain a complete functional specification. |
| Keep it simple. Complexity management is one of the major challenges. Strive to:  • Minimize interfaces between modules, procedures and data.  • Minimize interfaces between people, otherwise exponential communication cost  • Avoid fancy product functions, design if the functionality meets the customer requirements |
| Require visibility and transparency. We must see what we build so we can measure progress and take management action. For example, the manager must have good communication with his or her employees; require developers to make code available for review; review design for appropriateness. |
| Plan for continuous change. This includes:  • All manuals' designs, tests, source code should have revision numbers and dates revision history comments, change marks to indicate the changes  • New revisions should be approved before being made and checked for quality and compliance after being made  • Use a configuration management system and make processes  • Required maintenance |
| Don’t underestimate/overestimate. We must be careful to obtain accurate estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation and maintenance. |
| Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members |
| Software testing will use both black box and white box testing. It will involve unit, functional, integrating and acceptance testing. |
| Utilize good coding practices, such as:  • Follow design patterns where possible  • Leaving comments for explanation  • Common best practice coding principles |

**8 Risk Management**

Besides the general risk management, the following risks have been identified for the GoldFolks project:

**Unable to meet Project Deadline**

Impact Severity: High

Probability: Medium

Impacts: Reduced functionalities or delay the deployment date.

Risk Reduction: Have regular meetings and updates from the developer team to check on their progress.

**Hardware Failure**

Impact Severity: High Probability: Low

Impacts: Unable to meet deadlines as codes cannot be written or ran.

Risk Reduction: Maintain backups via version control software, e.g., git to minimize the impact.

**Feature Creep**

Impact Severity: High

Probability: Medium

Impacts: Will delay the project and may not be able to meet the set deadlines.

Risk Reduction: Maintain traceability between documentation and code to not go beyond the specified requirements.

**Updates to code not communicated to every group member**

Impact Severity: Medium

Probability: High

Impacts: Overlapping of codes which may lead to errors while running the program, and documentation may be inaccurate.

Risk Reduction: Ensure transparency between group members when it comes to updating of work done, which will lead to less miscommunication and conflicting information.

**Staff Unavailability**

Impact Severity: Medium

Probability: Medium

Impacts: Staff's work will not be completed.

Risk Reduction: Transfer the workload to other staff workers in order to ensure that work is still completed by deadlines.

**System not meeting requirements**

Impact Severity: High

Probability: Low

Impacts: Scheduled releases may be affected as that would lead to more debugging and testing.

Risk Reduction: Ensure that testing is done early so that the system can be checked against requirements with ample spare time.

**Poor code design (e.g., high coupling)**

Impact Severity: Medium

Probability: Low

Impacts: Modules cannot be separated

Risk Reduction: Design a system architecture and class diagram and ensure that it is followed when coding the application.

**COVID-19**

Impact Severity: Medium

Probability: Low

Impacts: Unable to meet up due to restrictions, teammates contracting the virus.

Risk Reduction: Ensure that we follow safe distancing measures and minimize physical meetings.

**Misunderstanding/Miscommunication of requirements from stakeholders**

Impact Severity: High

Probability: Low

Impacts: Requires us to redesign our application which delays the project

Risk Reduction: Establish a clear set of requirements and ensure that it is signed and agreed by both parties beforehand.

**Sensitive data being leaked/hacked**

Impact Severity: High

Probability: Low

Impacts: Loss in customer trust and brand reputation

Risk Reduction: Encrypt personal information given to the application so that hackers will not be able to access it.

**Incomplete Documentation**

Impact Severity: Low

Probability: Low

Impacts: May result in miscommunication between team members and may create issues in the future when maintaining the software

Risk Reduction: Perform consistent review of existing documentation and create a list of documentation to be maintained. Ensure that version control is in place for each document so that it is easy to track changes and verify information.

**Inconsistent Programming Style**

Impact Severity: Low

Probability: Medium

Impacts: Code will be harder to read and understand and it will be difficult for another person to maintain the software

Risk Reduction: Adhere to proper programming style and principles e.g., names, alignment, locality, comments.

**9 Quality Assurance**

The project will achieve quality assurance by following the standard set by the company. The specific procedures and details shall be provided in the Quality Plan.

Specific test procedures and details shall be provided in the Module/System Test Plan.

In addition, GoldFolks shall make use of the following testing methodologies:

* + - **Unit Testing** involves testing the functionality of the system components individually.
    - **Integration Testing** involves the testing of compatibility and interactions between individual components.
    - **System Testing** involves testing the whole system as a unit with every single component linked.
    - **Validation Testing** involves testing to ensure our individual components meet the user’s requirements, needs, business logic and possible scenarios.
    - **User Acceptance Testing** involves testing of the prototype by actual intended users.

Furthermore, these methodologies will be used to test two important aspects of GoldFolks:

* **System Function** will be tested to ensure that software flaws are eliminated, and
* **Algorithmic Function** will be tested to ensure that heuristic aspects of the project (such as game score rankings) perform realistically to provide value to the users.

GoldFolks’ methodology makes broad use of realistic test cases. Detailed test data is an important part of the final project delivery. Although GoldFolks’ client is expected to furnish and enter data regarding account details and medication reminders, GoldFolks shall provide a comprehensive and detailed subset of this data for testing purposes. GoldFolks will validate code and heuristic result ranking technology using realistic scenarios. In addition, extreme cases (such as ridiculous inputs by users) will be used to ensure that the system behaves correctly in degenerate cases.

**10 Monitoring & Control**

Many procedures are required in order to be able to successfully monitor the progress of a software project. Some of the most important are:

**Quantitative measurement of resource consumption:** Estimates of GoldFolks’ resource requirements, primarily in terms of human resources, can provide a quantitative measurement of project progress when compared to progress in terms of project milestones. The percentage estimates of each milestone’s resource requirements provided in this document allow for easy progress tracking.

**Identification of major project risks:** Early identification of major risks to the project allows for placement of preventative measures before problems can develop. Major risks have been identified in the Risk Management section of this document, along with the measures being taken to avoid them.

**Regular reviews of project progress:** Throughout the duration of the GoldFolks project, GoldFolks shall meet weekly to review the progress of all project tasks, including management, planning, analysis, development, and testing.

**Timeline Planning and task decomposition:** This document outlines an estimated timeline for the project. A reasonably accurate timeline can be assembled by hierarchically decomposing tasks into measurable subcomponents and estimating requirements for each. At the same time, this decomposition can assist in task assignment and balancing. Throughout the implementation phase, these subcomponents can allow for fine-grained measurement of progress. Project subcomponents and timeline estimates are included in the Estimates and Work Breakdown Structure sections of this document.

1. Lines of code per Person Day statistics based on Industrial Benchmarks, 1997: 31 LOC/PD for United States; 62 LOC/PD for Canada [↑](#footnote-ref-2)