PROJECT QUALITY PLAN

1. Introduction to the project

Medisay is a user-friendly online and mobile application for medication management. It reminds the patients to take their medicines at the prescribed time, and to visit their doctor on their appointments.

In addition, the patient's information is shared with a person who advises them and takes care of their health. The patient's adviser will be able to monitor the patient's health improvements, and check whether they take their pills and attend their appointments on time.

The application also keeps track of the patient's measurements (blood glucose, blood pressure, weight, etc.) and sends a report of them occasionally to the patient's adviser and doctor.

The sole purpose of Medisay is helping patients take care of their health effectively, in addition to facilitating the follow-up of their health by their families and those who care for them, from anywhere and at any time. The application aims to make patients' lives easier and healthier, especially the elderly and Alzheimer's patients.

First, we will describe the purpose, scope, define the general constraints that must be met, we will list functional requirements, and quality plans of the project.

The project quality plan will provide descriptions to ensure that the purpose of the project is understood, it will define standards, objectives, and responsibilities to be followed, also we will define the systemic and integration requirements after that we will list our quality goal that we want to achieve it.

Next, we will complete the quality plan and describe the quality objectives, planned reviews, verification, validation, and acceptance tests, then we will plan the configurable management and finally the responsibilities of the quality team.

2. Purpose of Quality Plan for project

The Purpose of the quality plan is to defines the quality goals and activities performed. We should do it to ensure the satisfaction of these goals, provide a complete list of planned software tests (acceptance tests). And it is important because it reflect the major acceptance criteria indicated in the customer's requirement document. It contains a list of all SDLC activities and deliverables to be reviewed to ensure that quality meets requirements. The quality plan should specify configuration management tools and procedures, including those change-control procedures meant to be applied throughout the project. A quality plan, rightly applied, protects the project, the product, the customer, and it will reduce the cost and time Without strong quality planning, a project carries an increased mistake in the project.

3. Project Scope

Medisay will helps the patients by remind them about their daily medication and mark it if taken. It allows the patients to set a reminder for their appointments and display them. Medisay will let the patient's advisor to follow all aspects that related to the patient's health by let him/her to have a report for the patient measurements, also he/she can display all patient's previous and upcoming appointments. To save the patient's health the advisors will know if the patient does not take his/her medication 30 minutes of its schedule time. The patient's doctor is responsible to mark the attendance of the patient's appointments and display them. All these tasks will do very securely so no one unless the patient and his/her advisor can access the application using their private information and their strong passwords. it will run on two platforms, web-based for the doctor and IOS application for the patient and his/her advisor. Medisay will be implemented using java script for web-based website and swift for IOS Application.

4. General Constraints

1. The system shall run under IOS platform, and it should have the ability to transfer to Android platform.
2. The system shall serve 600 concurrent users.
3. The system shall be available in Arabic.
4. The system's source code shall be written in flutter.
5. The system shall use SQLite as its main database.

5. Functional Requirements

1. The patient shall be able to sign up by a full name, email, phone number, password, gender, and birth date.
2. The patient shall be able to login to app using username and password.
3. The patient shall be able to set a reminder for his daily medications.
4. The patient shall be able to set a reminder for his medical appointments.
5. The patient shall be able to view list of his appointments.
6. The patient shall be able to fill his measurements (blood glucose, blood pressure, weight, height).
7. The patient shall be able to mark the taken medication on time.
8. The patient's advisor shall be notified when the patient did not take the medication within 30 minutes (unmarked medication).
9. The patient's advisor shall be able to edit the patient's profile.
10. The patient's advisor shall be able to get all the reports for the patient's measurements (blood glucose, blood pressure, weight, height).

11. The patient's advisor shall be able to register using his name, email, password, phone number,
patient name, and patient id.
12. The patient's advisor shall be able to login using his email and password.
13. The patient's advisor shall be able to view list of patients' appointments (previous and upcoming appointments).
14. The Doctor shall be able to display the patient list of appointment.
15. The Doctor shall be able to mark the attended appointments.

6. System and Integration Requirements

1. The system shall not allow patients to mark the attended appointments by themselves (attended appointments shall be marked only by the doctor).
2. The system shall authenticate patient's advisor registration via a verification code sent to the patient's email.
3. The salt technique should be used with the stored passwords, to avoid cryptographic failures.
4. Default accounts and their passwords should be disenabled after first access (user should be forced to change them).

7. Quality goals

Requirements	**McCall's Quality Factors**
The system shall ensure that time of servers and other devices shall be synchronized to a reliable reference time	Correctness
The system shall respond within 2 seconds to patient s' request.	Correctness
The system shall comply the measurement's report with medical reports standards	Correctness

The system must be available 24/7 of the time.	Correctness
The system load time shall not be more than one second for users.	Efficiency
The system shall be able to support up to 660 patients at the same time.	Efficiency
The system shall be with minimum design to prevent cognitive issues in senior citizens.	Usability
The system shall display a tutorial on how to use the application	Usability
The patient shall be able to set a reminder for his daily medications in 3 clicks.	Usability
The total time of failure status of "Medisay" should not exceed 10minutes per month.	Reliability
The "Medisay" documentation should be clear, selfdescriptive, and have a high degree of consistency.	Maintainability
The system shall be developed to adapt to the addition of multiple languages without major changes to the source code.	Flexibility
The "Medisay" shall be developed in two platforms, a web-based application which will serve the doctors, and cross platform mobile application which will serve the patients and their advisors.	Portability

The iOS application must support iPhone devices running on OS versions:3.6 and above.	Portability
The system shall ask the user for a password containing 8 characters at least to verify his identity.	Integrity

8. Planned Reviews

Project Reviews.

- **Requirements gathering and analysis:**
  1. Software requirements specification contract.
  2. List of all the recommendations of the adviser of the patient's ,and the recommendations of patients.
  3. List of all Saudi Health care ministry recommendations.
- **Software Design:**
  1. Deployment diagram.
  2. Use case diagram.
  3. class diagram.
- **Implementation:**
  1. Notification interface implementation.
  2. Software source code.
  3. Registration interface implementation.
- **Testing:**
  1. Integration testing for the response time of the notification .
  2. Test the plan review.
  3. Integration testing for the synchronization of the patient and adviser .
- **Maintenance:**
  1. Updating and patching critical update plans.
  2. Develop test cases for each function on the application.
  3. follow the Coding standards and guidelines .

- **Walkthroughs**
- **Requirements Gathering and analysis:**
  1. Software requirements specification contract. Does the software requirement specification consist of all users need ? Are all requirements correct and complete ? Is the requirement seen and signed by the customer ?
  2. List of all the recommendations of the adviser of the patient's ,and the recommendations of patients.Does the system requirement consider all the adviser and patient recommendations ? Are they correct and complete
  3. List of all Saudi Health care ministry recommendations. Does the system requirement consider all Saudi Health care ministry recommendations ? Are they correct and complete ?
- **Implementation:**
  1. Notification interface implementation. Does the implementation use a high priority type notification and utilize best practice? Does the content of the notification display the required information for the patient?
  2. Software source code. Does the source code implement all the required functionality of the system ? Does the system function contain the required information to be executed ?
  3. Registration interface implementation. Does the implementation adhere to the design phase specification for each user?
- **Testing:**
  1. Integration testing for the response time of the notification . Are all measurements of the response time of the notification acceptable for our system ? Is the testing of the response time of the notification well-planned?
  2. Test the plan review. Is the plan made correctly? Does it consist of all the test cases needed to test the functionality of the system? Does the plan make the acceptable measurement for our system if needed?
  3. Integration testing for the synchronization of the patient and adviser. Is the testing of the synchronization of the two users well-planned? Is the testing of the synchronization of the two users(patient and adviser) going well?
- **Maintenance:**
  1. Updating and patching critical update plan. Does the Updating of the system increase the quality of the system? Does the updating of the system not affect some other part of the system?

2. Write test cases for each function on the application. Does all the test cases manage the added functionality ?Does all the test cases manage the modification and updating of the function ?
3. follow the Coding standards and guidelines . Dose the coding standard and guidelines followed in the correct way? Does the code follow **all** Coding standards and guidelines?

9. Planned Verification Tests

1. Testing if the system operates in IOS platform, and it can transfer to Android platform.
2. Testing if the system handles up to 600 concurrent users.
3. Testing if the users doesn't take more than 1 hour to learn how to use the system.
4. Testing each functionality of the system with valid data and invalid data (test cases).
5. Testing if the links between the patient and a doctor is secured.

10. Planned Validation Tests

1. Testing if the application can let the patient set a reminder for his daily medications.
2. Testing if the application can let the patient's advisor register using his name email, password, phone number, patient name, and patient ID.
3. Testing if the application can let the patient's advisor edit the patient's profile.
4. Testing if the application will notify the patient's advisor when the patient does not take the medication within 30 minutes (unmarked medication).
5. Testing if the application can let the doctor display the patient list of appointments.

11. Planned Acceptance Tests

- The first acceptance test will be in the early stage, we will let the elderly in our families interact with Medisay and done the checklist that prepared to cover all functionalities that the users can do using the system.classifies them in three categories (easy,medium,hard). The hard functionalities will be improved before releasing the system.
- The second acceptance test will be after the implementation phase, we will let the users interact with the final product. Any problems found will be fixed before releasing the product.
- The third acceptance testing will be after the implementation phase, There will be an

alpha testing, we will let the patients try the system, ask them a question to let them think loudly. And gathering their feedback about their feelings and satisfying the system.

12. Planned configuration management.

Software Storage
for the web application, we will use the web localStorage.
The localStorage property of the window interface allows you to access a Storage object for the Document's origin; the stored data is saved across browser sessions and has no expiration time.

for the mobile application, we will use Firebase Database.
The Firebase Realtime Database is cloud-hosted. Data is stored as JSON and synchronized in real-time to every connected client. When building cross-platform apps with React Native SDK, all of the clients share one Realtime Database instance and automatically receive updates with the newest data.

Security and Backups
For the web app, we will use the localstorage-encrypt library, it helps with making the web application more secure by encrypting the client-side data before storing it in the local storage.

For the mobile app, Firebase services encrypt data in transit using HTTPS and logically isolate customer data. In addition, Firebase Realtime Database also encrypt data at rest using Firebase Test Lab.

Blaze plan users can set up their Firebase Realtime Database for automatic backups, a self-service feature that enables daily backups of your Database application data and rules in JSON format to a Cloud Storage bucket.

Version Control
For version control we will use git, Git is software for tracking changes in any set of files, and used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

The object format of Git's repository files uses a combination of delta encoding (storing content differences), compression and explicitly stores directory contents and version metadata objects. Git is compatible with Visual studio Code which we will use during developing our app.

13.Quality Assurance Process Metrics

Phase1: Requirement [1]

Much of requirements management involves handling requirement additions, modifications, and deletions. Therefore, this metric track the status and impact of requirements change requests in order to achieve the maintainability goal.

Metric Name	Metric	What it does?	Data need to be collected?
Requirements Volatility	Requirements Volatility = (added + modified + deleted requirements)/ initial number of requirements	Monitors how many changes are incorporated throughout development after you baselined the requirements for a specific release.	1.Initial number of requirements. 2.Number of added requirements. 3.Number of modified requirements. 4.Number of deleted requirements.

[1] "Identifying and measuring requirements quality," Jama Software, 23-Feb-2022. [Online]. Available: https://www.jamasoftware.com/requirements-management-guide/measuring-requirements/identifying-and-measuring-the-quality-of-requirements. [Accessed: 07-Apr-2022].

Phase 2: Design [2]

Average Severity of Development Errors is a metics that measure Severity of Development Errors (design and code).

Metric Name	Metric	What it does?	Data need to be collected?
Average Severity of Development Errors	Average Severity of Development Errors = total weighted development (design and code) errors detected in development process / total number of development (design and code) errors detected in the development process. $ASDE = \frac{WDE}{NDE}$	Detect adverse situations of increasing numbers of severe errors in situations where the weighted errors (design and code) are generally decreasing.	WDE = total weighted development (design and code) errors detected in development process. NDE = total number of development (design and code) errors detected in the development process.

[2]Galin, D. (2004). Software quality assurance : from theory to implementation.Pearson Addison Wesley

Phase 3: Implementation [3]

To support control of software development projects(implementation and design) .Use of these metrics data is aimed at enabling preventive and corrective actions throughout the organization.

Metric Name	Metric	What it does?	Data need to be collected?
WCED	WCED =Weighted number of code errors /Number of thousands lines of code WCED=WCE/KLOC	Measures the weight of code error density by using the number of error that have been weighed by severity with respect to total lines, to determine indicator for unacceptable software quality.	1. WDE: weighted code errors detected. 2. KLOC: Total thousands lines of code.

[3]Galin, D. (2004). Software quality assurance : from theory to implementation.Pearson Addison Wesley.

Phase 4: Testing [4]

The test case pass rate gives you a clear picture of the quality of the product being tested.
The value of this metrics should increase as the project progress. In case, test case passes rate decrease, it means that the QA team has to re-open the bugs which are even more alarming.

Metric Name	Metric	What it does?	Data need to be collected?
Test Case Pass Rate	Test Case Pass Rate = Number of Passed Test Cases/Total Number of Executed Test Cases	Indicates the quality of solution based on the percentage of passed test cases.	1. Number of Passed Test Cases 2. Total Number of Executed Test Cases

[4]Guru99. 2022. What is WHITE Box Testing? Techniques, Example & Types. [online] Available at:<https://www.guru99.com/white-box-testing.html#2> [Accessed 2 April 2022].

Phase 5: Maintenance [5]

During this phase the defect arrivals by maintenance time and customer problem calls (which may or may not be defects). What can be done during the maintenance phase is to fix the defects as soon as possible and with excellent fix quality.
Full availability (FA): where **all** software system functions perform properly**.**

Metric Name	Metric	What it does?	Data need to be collected?
Full availability (FA)	FA=(number of hours software system is in service during one year –number of hours where at least one function is unavailable during one year including total failure of the software system) / number of hours software system is in service during one year.	It is a simple metric to Indicates of all software function performs properly and corrective maintenance quality during the life cycle. Using this metric can provide meaningful information for managing the maintenance process.	1. The number of the hours software system is in service during one year. 2. The number of hours where at least one function is unavailable during one year including total failure of the software system.

[5]Galin, D. (2004). Software quality assurance : from theory to implementation.Pearson Addison Wesley.

14. Project Team Quality Responsibilities

Name	*Role*	*Signature*	*Date*
Sarah Aboumahmoud	System and integration requirements, planned configuration management Metric of Requirements.	Sarah	7-4-2022
Shahed Hamadmad	Project Scope,Planned Acceptance Tests. Metric of design.	Shahed	7-4-2022
Fatimah Dagriri	General Constraints, Planned Validation Tests Metric of testing.	Fatimah	7-4-2022
Ruba Alfelaih	Purpose of Quality Plan for project, Planned Verification Tests Metric of maintenance.	Ruba	7-4-2022
Ruyuf albarrak	introduction of the project, planned Reviews. Metric of implementation.	Ruyuf	7-4-2022