PROJECT QUALITY PLAN

PHASE 3 & 4

#	Student Name	ID	Responsibilities
1	Nouv B. Al-Qahtani	442201905	Phase 3: Project Scope, General Constraints, Planned Acceptance Tests Phase 4: Maintenance metric.
2	Nada Al-Kubra	442202368	Phase 3: Quality goals system and Integration, System and Integration Requirements, Planned Validation Tests Phase 4: Recruitment metric.
3	Yosra Ahmed	442201045	Phase 3: Planned Reviews, Planned Verification tests Phase 4: Design metric, Implementation metric.
4	Asma Al-Shehri	442200945	Phase 3: Intro for the project, the purpose of quality plan and planned configuration management. Phase 4: Test metric.

Instructor Name: Dr. Meriam Kefi

1. Introduction to the project

LOCKIFY project is an application that is supported by IOS operating system, and it aims to give the users the ability to control, reserve and check on their lockers.
Every user should have an id that allows him/her to sign in with the application. When the user wants to reserve a locker, the system should check if he/she has already reserved one. If he doesn't have then he can reserve one. Every time the user wants to open the locker, he/she has to scan the barcode for a security reason.
This document is about the quality plan that we follow for our project to ensure the project quality, and we will discuss the activities. First, we will discuss project scope and what it will do & how it will do it. Then we will mention the general constraint on the project. Lastly, we have the requirements, quality goals and the planned tests.

2. Purpose of Quality Plan for project

The quality plan ensures that project management activities are carried out smoothly, and it gives all information about our project to the team members, which helps the team members know what they are expected to do. It increases efficiency and productivity and customer satisfaction. Also, it helps the stakeholders and top-level management check the progress of a project.

3. Project Scope

Lockify, an upcoming iOS application, is being developed to cater to the needs of Individuals in Need of Temporary Storage. Its primary purpose is to assist these individuals in locating secure storage spaces. Additionally, Lockify aims to cater to Organizations that possess locker areas within their buildings.

The application will be exclusively available in Saudi Arabia. Our vision is to ensure that Lockify offers a set of features that meet high-quality standards, aligning with our comprehensive quality plan. We strive to deliver a user experience that meets the expectations of our target users while adhering to the desired quality levels.

4. General Constraints

1- The system shall be available in English and Arabic.
2- All scripts shall be written in Swift programming language.
3- The system shall run under IOS platform and have the ability to transfer to an Android platform.
4- The System shall incorporate Apple Pay payment.

5. Functional Requirements

User’s Functional Requirements

1. The user shall be able to register using his/her name, organization’s ID number, mobile number, email address, and password.
2. The user shall be able to login using his/her ID number and password.
3. The user shall be able to search for available lockers.
4. The user shall be able to view locker details.
5. The user shall be able to reserve locker.
6. The user shall be able to set the duration of the locker reservation.
7. The user shall be able to view his/her specific locker location.
8. The user shall be able to view his/her profile information (full name, id number, mobile number, email, and recently booked lockers).
9. The user shall be able to receive notifications.
10. The user shall be able to pay via Visa/Mastercard payment systems.
11. The user shall be able to modify (time, location) on his/her reservation.
12. The user shall be able to provide feedback on their locker experience.
13. The user shall be able to report any issues encountered.
14. The user shall be able to reset his/her password.
15. The user shall be able to logout of his/her account.
16. The user shall be able to delete their account.

Organization’s Functional Requirements

18. The organization shall be able to register using the organizations name, id number, official email, region, city, username, and password.
19. The organization shall be able to login using the organizations username and password.
20. The organization shall be able to determine the available lockers.
21. The organization shall be able to view the information about the specific locker reserver.

22. The organization shall be able to cancel a user’s reservation for a specific locker.
23. The organization shall be able to reset their password.
24. The organization shall be able to log out of their account.
25. The organization shall be able to delete their account.

6. System and Integration Requirements

Integrate with a secure payment gateway for handling transactions related to locker reservations.
Display accurate and timely information on locker availability to users.
Integrate with location-based services to enhance user experience.
Ensure secure communication for location data to protect user privacy.

7. Quality goals

Requirements	McCall's Quality Factors
Product operation	
The system shall respond to user requests within less than 3 seconds	Correctness
The application shall send an OTP (one time password) to the user upon logging in	Integrity
The system must have a failure rate less than one per million cases.	Reliability
The system shall be able to restore service in case of a failure in less than 30 minutes.	
The user shall be able to register within two minutes.	Usability
New users shall be able to learn how to use the system within 15 minutes.	
The system should support a minimum of 100 locker access requests per minute.	Efficiency
The application should not consume more than 100 MB of RAM per user session.	
Product revision	
The system must adapt with at least 4 of 5 changed requirements	Flexibility
Product transition	
The system can run on future iOS/Android versions without changing the behavior of applications.	Portability
The login function in the application can be reused for the development of other system	Reusability

8. Planned Reviews

Project Reviews

Requirements gathering and analysis:

1. functional requirements review.
2. List of all expected Subscribed users' recommendations.
3. Non-functional requirements review.

Software Design:

1. use case diagram.
2. sequence diagram
3. Class diagram

Implementation:

1. Registration interface implementation.
2. User profile interface implementation. `
3. Database implementation.

Testing:

1. Test plan review.
2. Testing system response time.
3. Testing the payment systems.

Maintenance:

1. Security Updates.
2. Adding new features to the program.
3. User Support and Helpdesk.

Walkthrough

Requirements Gathering and analysis:

1. functional requirements review: Does the functional requirement cover all targeted users' needs? Are the functional requirements clear, and correct?
2. List of all expected Subscribed users' recommendations: Does the system requirement consider the Subscribed users' recommendations? Are the system requirements comprehensive?
3. Non-functional requirements review: Are all Non-functional requirements measurable? Does it cover all aspects of the system?

Software Design:

1. Use case diagram: Does the use case diagram accurately depict how a user interacts with the system?
2. Sequence diagram: Does the sequence diagram show how objects and components interact with each other to complete the process?
3. Class diagram: Are all relationships between classes represented correctly?

Implementation:

1. Registration interface implementation: Does the implementation match the requirements from the design phase?
2. profile interface implementation: Does the implementation match the requirements from the design phase?
3. Database implementation: Does the user information stored securely in the database?

Testing:

1. Test plan review: Does the plan accurate and comprehensive? Does it include every test case required to verify the system's functionality?
2. Testing system response time: Does the system respond to user requests within less than 3 seconds?
3. Testing the payment systems: Does the payment system work properly?

Maintenance:

1. Security Updates: Are emerging security threats being monitored and responded to?
2. Adding new features to the software: Do the features work as planned? Does it meet evolving user needs?
3. User Support and Helpdesk: Is ongoing support provided to users, addressing inquiries, and troubleshooting problems?

9. Planned Verification Tests

1. Testing if the new user can learn how to use the system within 15 minutes.
2. Testing if the system restore service in case of a failure in less than 30 minutes.
3. Testing that the response time of the system to user requests will not exceed 3 seconds.
4. Testing the system's performance by generating a minimum of 100 locker access requests per minute.
5. Testing the time it takes for a user to complete the registration process to ensure it is within two minutes.

10. Planned Validation Tests

- Validate that the system provides accurate and real-time updates on locker availability to users.
- Validate the reservation and payment functionalities to ensure users can successfully reserve lockers and complete transactions securely.
- Validate the user registration process and the functionality of managing user profiles.
- Validate test if that the Lockify application allows users to successfully create a new account.
- Validate test if that The user can log in successfully with the newly created account.

11. Planned Acceptance Tests

`There will be two acceptance testing:
1. First acceptance testing will be after designing the software. There will be a prototype that the user will interact with. And he/she can try the system and provide feedback. It helps find bugs and flaws before the actual version of the product is built.
The prototype will implement all the functional requirements.
2. Second acceptance testing will be after implementing the software. There will be alpha testing. is done by specialized testers to ensure the system works and does everything it's supposed to do and give suggestions to improve product usability in a controlled manner.
Any failing test will be fixed before releasing the product.

12. Planned configuration management.

Software Storage

For our project we will use "Microsoft Azure Blob Storage".
In our case we need a server that provides security for the data and low-latency access at the same time. And this is suitable for our project since we deal with sensitive information about the user.
For the database the project will use Firebase Realtime Database as data storage, and we chose it because of its mobile platform and updates.

Security and Backups

As we said before, "Microsoft Azure Blob Storage" provides security for the data, and we must ensure that every data transit is encrypted. And we have to restrict who can access, modify, or delete data on the application. We will use strong authentication like personal id number as a way to determine if this user should access this data or not.
For the backups, we decided to implement a regular backup strategy (daily) to prevent any loss of data, "Microsoft Azure Blob Storage" is a good choice here.

Version Control

Git is a well-known tool that is one of its main advantages is speed and flexibility and it gives access to all previous changes which allows us to call back any version we want. Every developer has to install git in their devices and has to set up the repository. The developer can make any changes and look back for the version history locally without needing to connect to the network.

13. Quality Assurance Process Metrics

Phase 1 :

- **Measuring Requirements phase quality.**

Metric Name	Metric	What it does do?	Data need to be collected?
Requirement Quality (RQ).[1]	RQ =∑ (IRQ) / (n)	shows how well your requirements meet quality standards. Higher RQ means better quality, guiding efforts to improve overall requirement clarity and completeness.	1. IRQ = Individual requirement Quality 2. n: Total number of requirements.

Phase 2 :

- **Measuring Design phase quality.**

Metric Name	Metric	What it does do?	Data need to be collected?
Weighted Development Error Density (WDED).	WDED = total weighted development error / thousands of lines of code WDED = WDE/KLOC	This metric measure the total weighted development (design and code) errors detected in the development of the software to determine unacceptable software quality.	1. WDE = total weighted development (design and code) errors detected in development process. 2. KLOC: number of thousands of lines of code.

Phase 3 :

- **Measuring Implementation phase quality.**

Metric Name	Metric	What it does do?	Data need to be collected?
Code Reuse (CRe)	CRe = number of thousands of reused lines of code / thousands of lines of code. CRe = ReKLOC / KLOC	measure of how effectively and efficiently code is reused in software development based on the total number of lines of code.	1. ReKLOC = number of thousands of reused lines of code. 2. KLOC: number of thousands of lines of code.

Phase 4:

- **Measuring Testing phase quality.**

Metric Name	Metric	What does it do?	Data need to be collected?
Test Case Effectiveness (TCE) [2]	TCE = (Number of defects detected / Number of test cases run) x 100	It gives us that the testing process is meaningful and provides valuable information about the quality of the software. [3]	1. The number of defects detected. 2. The number of test cases run.

Phase 5:

Measuring Maintenance phase quality.

Metric Name	Metric	What it does do?	Data need to be collected?
Weighted Software System Failure Density (WSSFD)	WSSFD = WYF / KLMC WYF = weighted number of yearly software failures detected during one year of maintenance service. KLMC = thousands of lines of maintained software code.	This metric deals with the extend of demand for corrective maintenance based on the weighted number of software failures detected during one year.	1. Weighted number of failures detected during one year of maintenance service. 2. Thousands of lines of maintained software code.

14. Project Team Quality Responsibilities

Name	*Role*	*Signature*	*Date*
Asma Alshehri	Developer member Configuration Management Control	[illegible]	19/11/2023
Nouv B. AlQahtani	Quality Assurance member General Constraints, Project Scope Quality related responsibility: Acceptance Tests	[illegible]	19/11/2023
Nada ALKubra	Quality goals system and Integration, System and Integration Requirements, Planned Validation Tests	**Nada Nasser**	19/11/2023
Yosra Ahmed	Quality Assurance member Quality related responsibility: Planning reviews and walkthroughs	[illegible]	19/11/2023

References:

[1] Codoid. Software Testing Metrics. Available: https://codoid.com/software-testing/software-testing-metrics/. Accessed: October 21, 2023.
[2] GeeksforGeeks. "Software Testing Metrics: Its Types and Example." Available: https://www.geeksforgeeks.org/software-testing-metrics-its-types-and-example/. Accessed: October 21, 2023.
[3] Scaler. "Test Metrics in Software Testing." Available: https://www.scaler.com/topics/software-testing/test-metrics/. Accessed: October 21, 2023.