SOFTWARE TEST DOCUMENT

for

HEALTHKARD

Group ID: 3

Umang Thadani (1914061)
Anurag Singh (1914058)
Dhruv Solanki (1914059)
Aayush Kapoor (1914066)

Guide: Prof. Era Johri

Contents

1 Introduction

1.1 Product Overview

Currently, there is no unified framework in India that facilitates the storage of health records of all citizens. Although private companies/hospitals do solve this problem to some extent by providing e-health record services, they only cater to their own patients/customers. Therefore, there is a need to develop the foundations necessary for supporting digital health infrastructure to maintain health data in a decentralized and secure way. A few major advantages to this project will be ease of access, user consent for every sophisticated transaction, and portability across national borders.

HealthKard aims to implement the following modules:

- Creation of a unique Health ID using Aadhaar Number
- Storage of Electronic Health Records (EHRs) mapped to Health Identity in the blockchain
- Integration of different sectors in the medical industry
- Encourage better administration of the health sector by utilizing health data analytics

1.2 Test Approach

Test will be conducted to test the efficiency of the software. The reason for this test is to check for any errors and limitations. A list of various planned tests is as follows:

1.2.1 Unit Testing

Unit testing, is a testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with the functional correctness of the standalone modules. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

1.2.2 White Box Testing

White Box Testing is a software testing technique in which the internal structure, design, and coding of software are tested to verify the flow of input-output and to improve design, usability, and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing. White box testing techniques analyze the internal structures, the used data structures, internal design, code structure and the working of the software rather than just the functionality as in black box testing. It is also called glass box testing or clear box testing or structural testing.

1.2.3 Black Box Testing

Black box testing is a method of software testing that tests the functionality of an application as opposed to its internal structures of working. Specific knowledge of the application's code/internal structure and programming knowledge, in general, is not required. The tester is only aware of what the software is supposed to do, but not how i.e. When he/she enters a certain input, he/she gets a certain output; without being aware of how the output was produced in the first place. Test cases are built around specifications and requirements i.e. what the application is supposed to do. It uses external descriptions of the software, including specifications, requirements, and design to derive test cases.

1.3 Test Plan

The objectives supported by the Test Plan are:

1. To test the project such that it meets the business and user requirements.
2. The Test Plan should find defects that may get created by the programmer while developing the software.
3. To make sure that the end result is correct.
4. Testing can be helpful in gaining confidence in and providing information about the level of quality.

The scope of the Test Plan are: The major functions that would be tested are:

1. Metamask Authentication
2. Unique Health ID Generation
3. Health and Consultation History

The following points are to be considered while testing:

1. Perform unit testing on individual modules

2. Integrate the modules to perform the integration testing after the successful completion of unit testing.
3. Then, the system as a whole will be tested and we will perform the system testing.
4. Different scenarios that can occur once the project will be deployed have been considered for ensuring effective testing of the software.

1.3.1 Features to be tested

Features	Scenarios
Metamask Authentication	• Login when the user mapped with the same address exists. • Create a new user in case a user with the same address does not exist. Verification through Aadhar has to be done in such a case.
Health ID Generation	• NFT must be generated when the user creates his/her profile • The details are not editable and NFT must be tamper-proof
Health Card Dashboard	• Show details relevant to only that particular user • Authorization is taken care of in this step
Patient Request	• Medical Experts should be able to request patients for access to patient's health records and the same should be reflected on the Patient's account • Patient's health records should not be visible to healthcare professionals without consent.

Features	Scenarios
Manage Approval	• Patients should be able to grant/revoke access to medical practitioners to access health records. • The same should be reflected on the Practitioner's Dashboard.
Adding Medical Records	• Users should be able to add medical health records linked to their Unique Health ID. • Medical experts with access should also be able to add to patients' health records.
Editing Medical Records	• Users should be able to edit medical health records linked to their Unique Health ID.

1.3.2 Testing Tools and Environment

Testing Staff

Personnel	Count
Test Lead	1
Tester	2

Testing Environment

Operating System

1. MAC Operating system
2. Microsoft Windows (Windows 8, 10, 11)
3. Linux Ubuntu

Devices

1. Laptop/Desktop
2. Mobile/Tablet

Browser requirement

1. Google Chrome

2. Microsoft Edge
3. Mozilla Firefox
4. Safari

Development

1. Network: Ethereum
2. Authentication: Metamask
3. File System: IPFS
4. Editor: Visual Studio Code
5. Documentation: Overleaf

1.4 Test Cases

Test Case ID	Module Name	Input	Expected Output	Actual Output	Status (Pass/-Fail)
1	Metamask Authentication	User Wallet Address	• Ensure that Metamask is connected without any issues	Metamask is successfully connected	Pass
2	Health ID Generation	• Name • Date of Birth • Blood Group • Gender • Aadhar Number • Phone Number • E-Mail • Photo	• NFT must be generated when the user creates his/her profile • The details are not editable and NFT must be tamper-proof	NFT is generated and cannot be tampered	Pass

Test Case ID	Module Name	Input	Expected Output	Actual Output	Status (Pass/-Fail)
3	Health Card Dashboard	User Wallet Address	• Show details relevant to only that particular user • Authorization is taken care of in this step	Relevant details are shown to the user	Pass
4	Patient Request	Document ID, Patient Address and Health Expert Address	• Medical Experts should be able to request patients for access to patient's health records and the same should be reflected on the Patient's account • Patient's health records should not be visible to healthcare professionals without consent.	The request flow for patient's documents works fine	Pass

Test Case ID	Module Name	Input	Expected Output	Actual Output	Status (Pass/-Fail)
5	Manage Approval	Document ID, Patient Address and Health Expert Address	• Patients should be able to grant/revoke access to medical practitioners to access health records. • The same should be reflected on the Practitioner's Dashboard.	The patients are able to grant and revoke access of EHRs	Pass
6	Adding Medical Records	Document, Wallet Address	• Users should be able to add medical health records linked to their Unique Health ID. • Medical experts with access should also be able to add to patients' health records.	Users are able to add documents to the blockchain	Pass
7	Editing Medical Records	Document ID, Patient Wallet Address	• Users should be able to edit medical health records linked to their Unique Health ID.	Users are able to edit/delete documents	Pass