**ATMIYAUNIVERSITY**

**RAJKOT**

A

Report On

# CORONA WEBSITE

Under subject of

**MAJOR PROJECT**

B.TECH Semester– VII

(Computer Engineering)

Submitted by:

| | | |
|---|---|---|
| 1. | PUSHTI DEPANI | 190002023 |
| 2. | JANVI GARACH | 190002033 |

**Prof. Nirali Borad**

(Faculty Guide)

**Prof. Tosal M.Bhalodia**

(Head of the Department)

Academic Year

**(2020-21)**

## CANDIDATE’SDECLARATION

We hereby declare that the work presented in this project entitled “**CORONA WEBSITE”** submitted towards completion of project in **7th Semester** of B.Tech. (Computer Engineering) is an authentic record of our original work carried out under the guidance of “**Prof. Nirali Borad”.**

We have not submitted the matter embodied in this project for the award of any other degree.

Sem:7th

Place:Rajkot

**Signature:**

Pushti Depani(190002023)

Janvi Garach (190002033)

**ATMIYA**

**UNIVERSITYRAJKOT**

# <u>CERTIFICATE</u>

Date:

This is to certify that the "**CORONA WEBSITE**" has been carried out by **Pushti Depani** under my guidance in fulfillment of the subject Project in COMPUTER ENGINEERING (7thSemester) of Atmiya University, Rajkot during the academic year 2021.

| | |
|---|---|
| Prof. Nirali Borad | Prof.Tosal M.Bhalodia |
| **(Project Guide)** | **(Head of the Department)** |

**ATMIYA**

**UNIVERSITYRAJKOT**

**<u>CERTIFICATE</u>**

Date:

This is to certify that the "**CORONA WEBSITE**" has been carried out by **Janvi Garach** under my guidance in fulfillment of the subject Project in COMPUTER ENGINEERING (7th Semester) of Atmiya University, Rajkot during the academic year 2021.

| | |
|---|---|
| Prof. Nirali Borad | Prof.Tosal M.Bhalodia |
| **(Project Guide)** | **(Head of the Department)** |

# ACKNOWLEDGEMENT

We have taken many efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to Prof. Nirali Borad for their guidance and constant supervision as well as for providing necessary information regarding the Major Project titled **"CORONA WEBSITE".** We would like to express our gratitude towards staff members of Computer Engineering Department, Atmiya University for their kind co- operation and encouragement which helped us in completion of this project.

We even thank and appreciate to our colleague in developing the project and people who have willingly helped us out with their abilities.

Pushti Depani(190002023)

Janvi Garach (190002033)

# ABSTRACT

Since its emergence in December 2019, corona virus diesease 2019 (COVID-19) has impacted several countries, affecting more than 90 thousand patients and making it a global public threat.The project 'Corona Website' is based on the database ,object oriented and networking techniques as there are many areas where we keep the records in database for which we are using my sql software which is one of the best and easiest software to keep our information. This project uses node js as the front-end software which is an Object Oriented Programming and has connectivity with database.

# INDEX

# LIST OF FIGURES

# LISTOFTABLES

# INTODUCTION

## 1.1 INTRODUCTION

This project aims to develop content in the COVID 19 category and also the Respiratory and Mental Health Sections of Physiopedia as a response to the COVID-19 pandemic. We intend to populate the site with practical, credible and thought-provoking information on all aspects of management of individuals with a diagnosis of COVID-19.

It maintained a daily-updated dataset of state-level information related to the outbreak, including counts of the number of cases, tests, hospitalizations, and deaths, the racial and ethnic demographic breakdowns of cases and deaths, and cases and deaths in long-term care facilities.

## 1.2. PURPOSE

Manual hours that need to spend in record keeping and generating reports. The data in a centralized way which is available to all the event managers. Easy to manage historical data in database. Participants can register for any happening event from anywhere. Event manager can keep records of participants. Easily generate certificates for participants and winners. Certificate mail to particular participants.

## 1.3. SCOPE

1. The entire project mainly consists of 3 modules, which are

   - Admin module
   - User module

2. Admin has the authority to add/delete users,grant permission to doctors.
3. Dashboard: In this section, doctor can view his/her own profile and online appointments.
4. Appointment History: In this section, Doctor can see patient’s appointment history.
5. Patients: In this section, doctor can manage patients (Add/Update).
6. Search: In this section, doctor can search patient with the help of patient name and mobile number.

## 1.4 ORGANISATION OF REPORT

### 1.4.1 INTRODUCTION

This section includes the overall view of the project i.e. the basic problem definition and the general overview of the problem which describes the problem in layman terms. It also specifies the software used and the proposed solution strategy.

### 1.4.2 SOFTWARE REQUIREMENTS SPECIFICATION

This section includes the Software and hardware requirements for the smooth running of the application.

### 1.4.3 DESIGN & PLANNING

This section consists of the Software Development Life Cycle model.It also contains technical diagrams like the Data Flow Diagram and the Entity Relationship diagram..

### 1.4.4 RESULTS AND DISCUSSION

This section has screenshots of all the implementation i.e. user interface and their description.

### 1.4.5 SUMMARY AND CONCLUSION

This section has screenshots of all the implementation i.e. user interface and their description.

# CHAPTER 2 :
# SOFTWARE REQUIREMENTS SPECIFICATION

## 2.1 Hardware Requirements

**Table 2.1.1 Hardware Requirements**

| **Number** | **Description** |
|---|---|
| **1** | PC with 512 GB |
| **2** | PC with 4 GB RAM |

## 2.2 Software Requirements

**Table 2.2.1 Software Requirements**

| **Number** | **Description** | **Type** |
|---|---|---|
| **1** | Operating System | Windows XP / Windows |
| **2** | Language | NodeJS |
| **3** | Database | MongoDB |
| **4** | IDE | Visual studio Code |
| **5** | Browser | Google Chrome |

# CHAPTER 3
# DESIGN & PLANNING

## 3.1 SOFTWARE DEVELOPMENT LIFE CYCLE MODEL

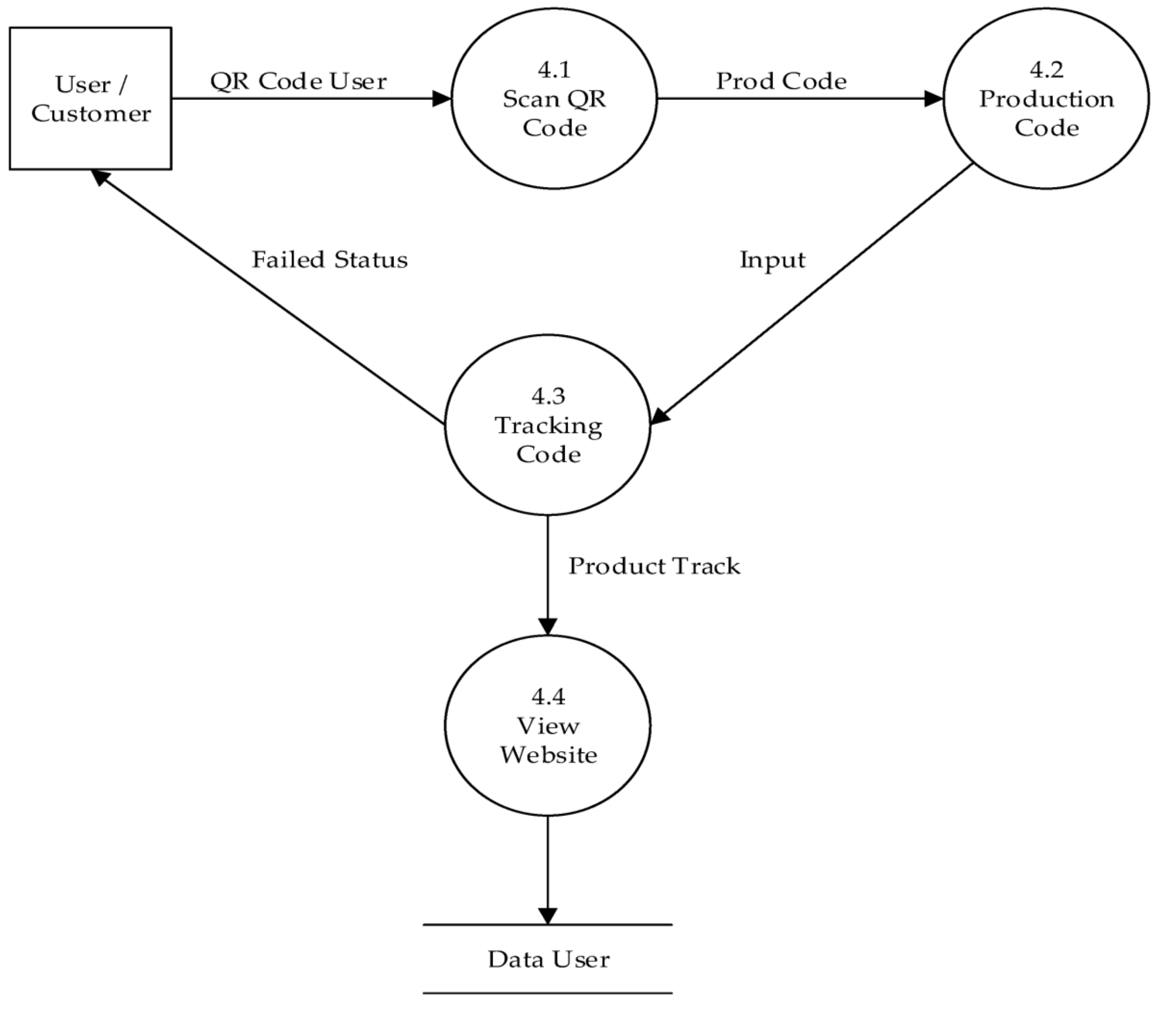
### 3.1.1 WATERFALL MODEL

The waterfall model was selected as the SDLC model due to the following reasons:

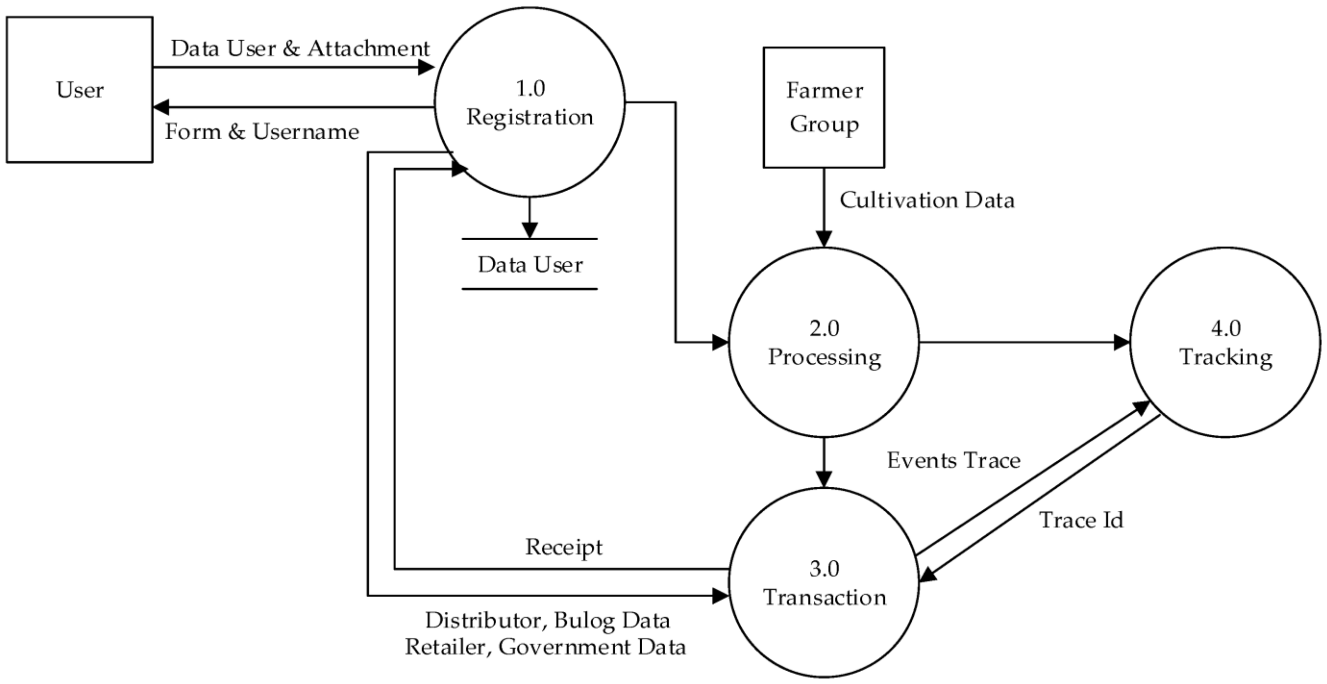
Requirements were very well documented, clear and fixed. Technology was adequately understood. Simple and easy to understand and use. There were no ambiguous requirements.
Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process. Clearly defined stages. Well understood milestones easy to arrange tasks.

## 3.2 DATA FLOW DIAGRAM

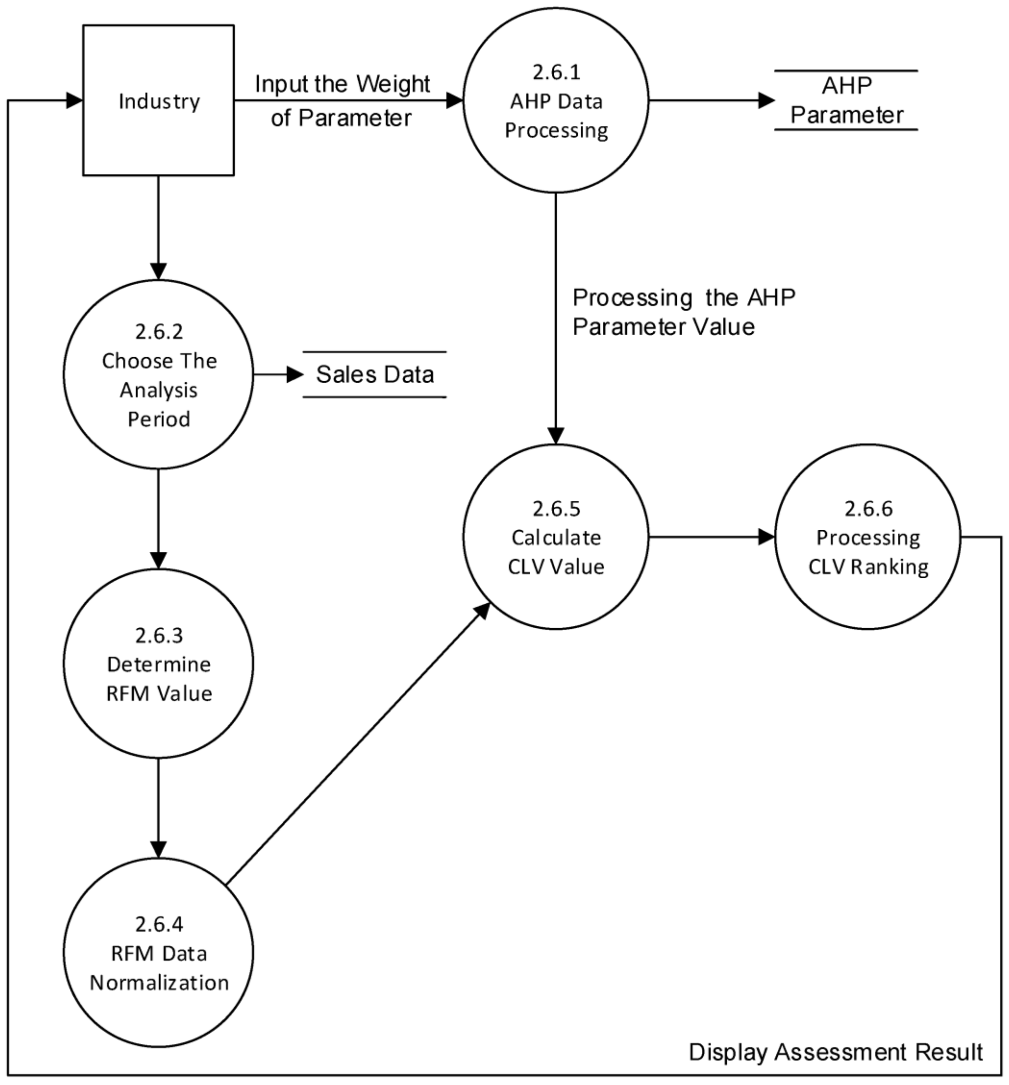
### 3.2.1 Level 0 DFD



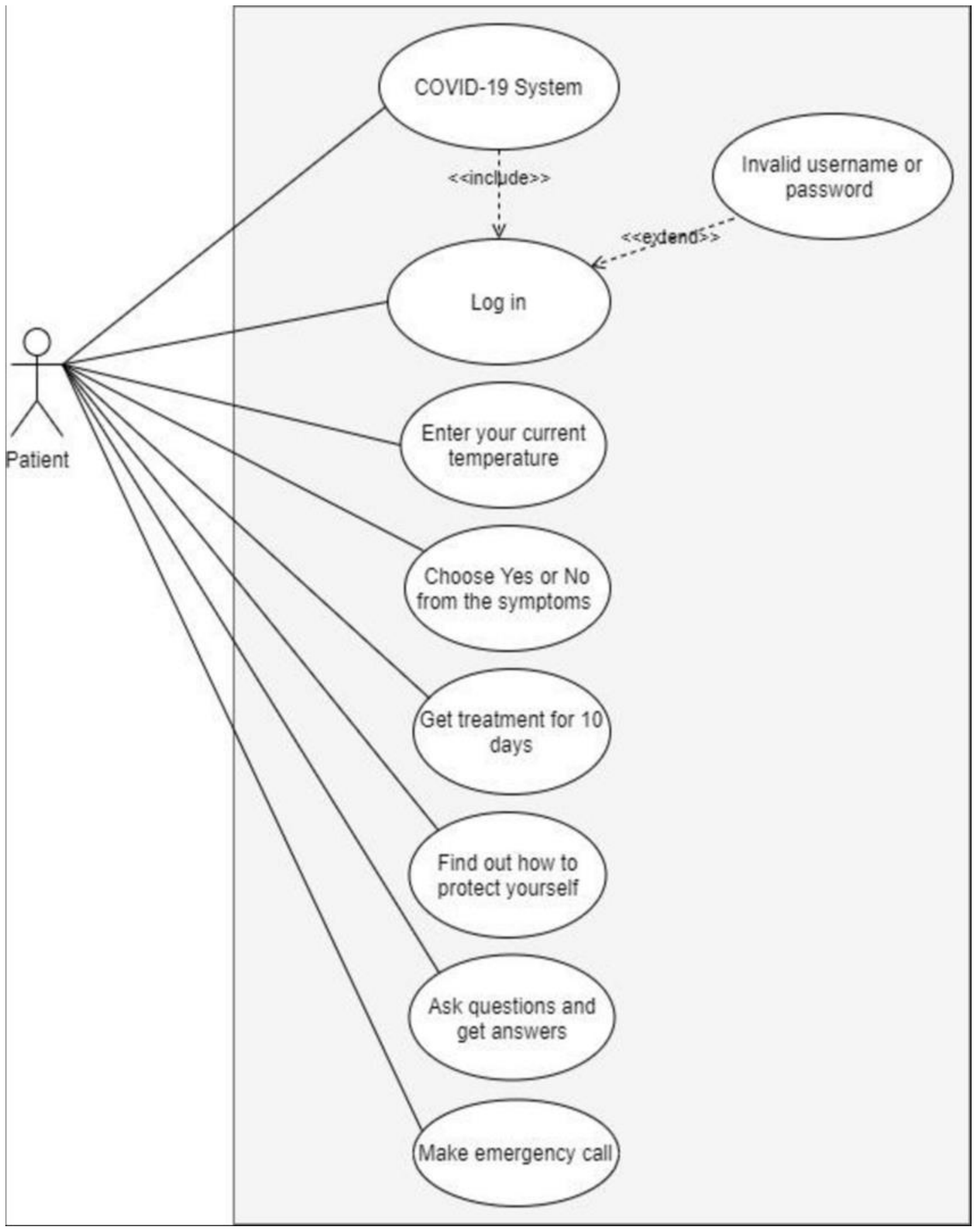
### 3.2.2 Level 1 DFD



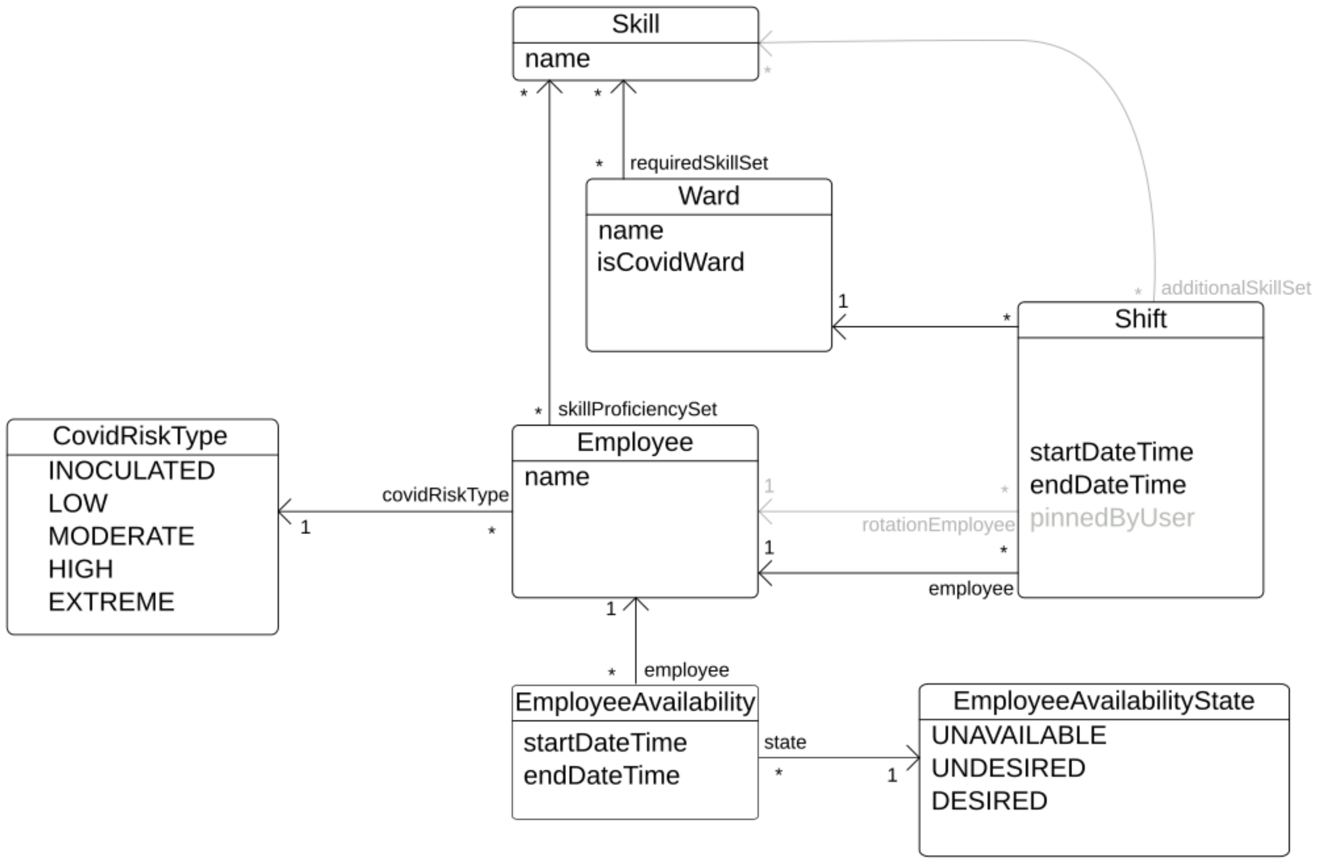
## 3.2.3 Level 2 DFD


Industry
Input the Weight of Parameter
2.6.1 AHP Data Processing
AHP Parameter
Processing the AHP Parameter Value
2.6.2 Choose The Analysis Period
Sales Data
2.6.3 Determine RFM Value
2.6.4 RFM Data Normalization
2.6.5 Calculate CLV Value
2.6.6 Processing CLV Ranking
Display Assessment Result

## 3.3 USE CASE DIAGRAM

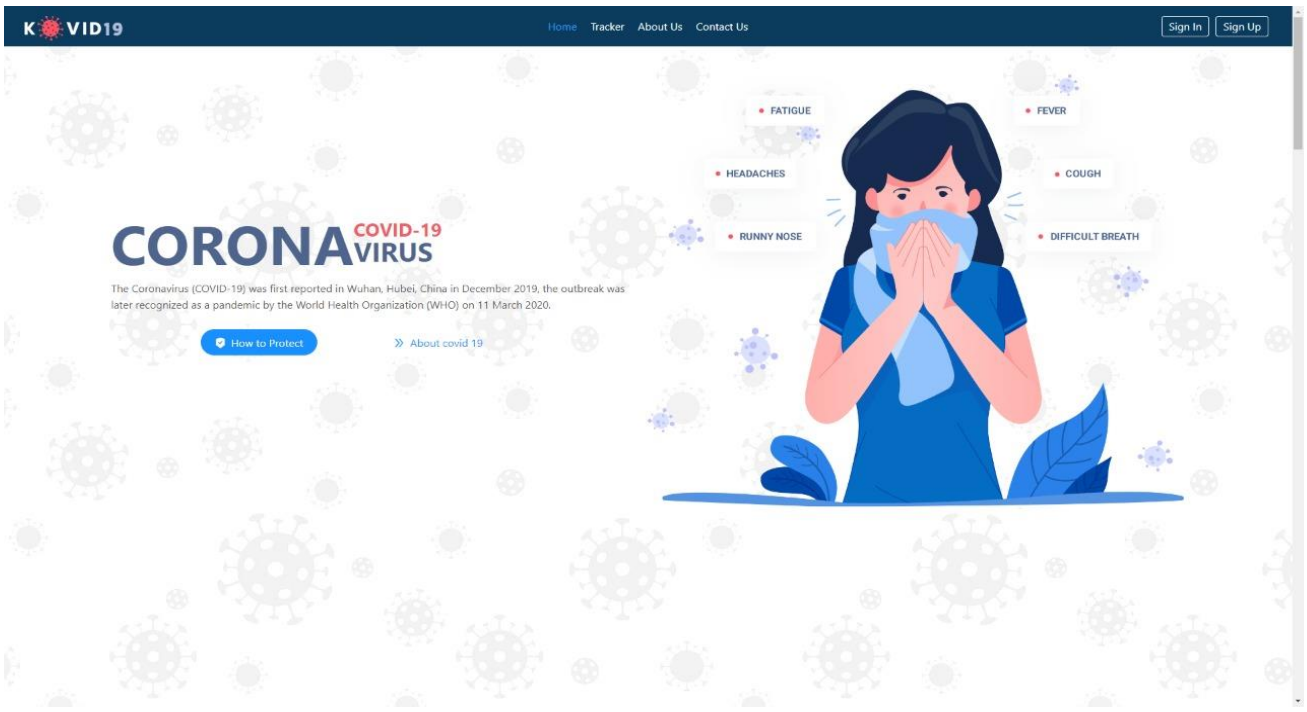

COVID-19 System
<<include>>
Invalid username or password
<<extend>>
Log in
Patient
Enter your current temperature
Choose Yes or No from the symptoms
Get treatment for 10 days
Find out how to protect yourself
Ask questions and get answers
Make emergency call

## 3.4 CLASS DIAGRAM



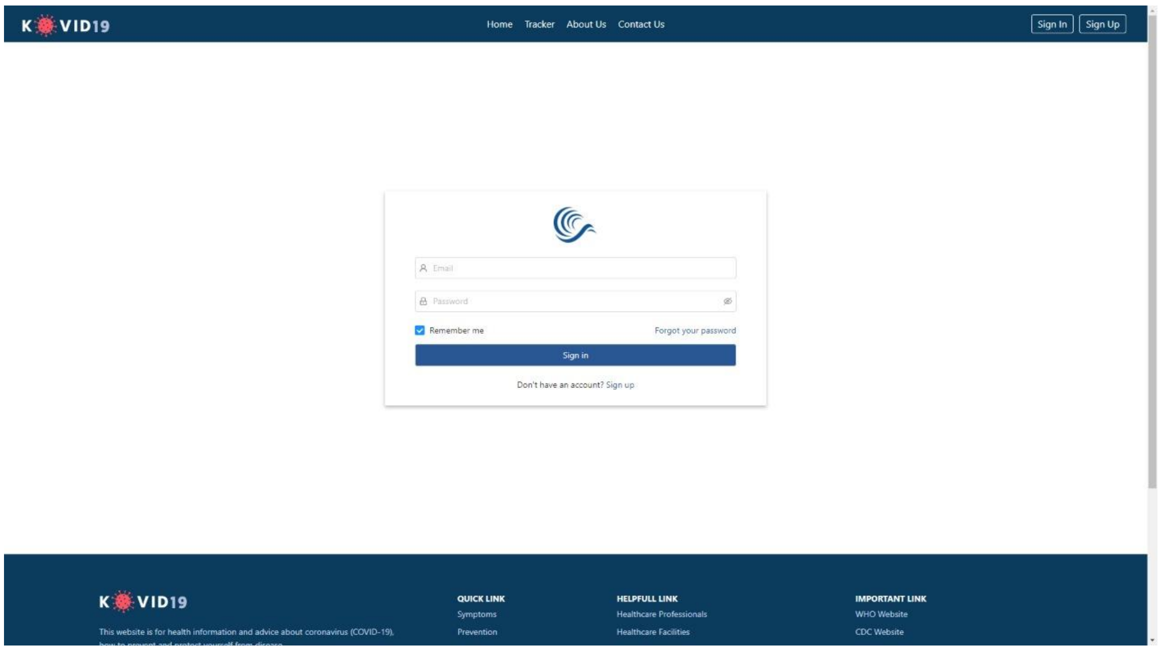
## 3.5 Input /Output Interface

Fig.3.5.1 .Home Page



The Coronavirus (COVID-19) was first reported in Wuhan, Hubei, China in December 2019, the outbreak was later recognized as a pandemic by the World Health Organization (WHO) on 11 March 2020.

Fig.3.5.2 Sign In Page



Email

Password

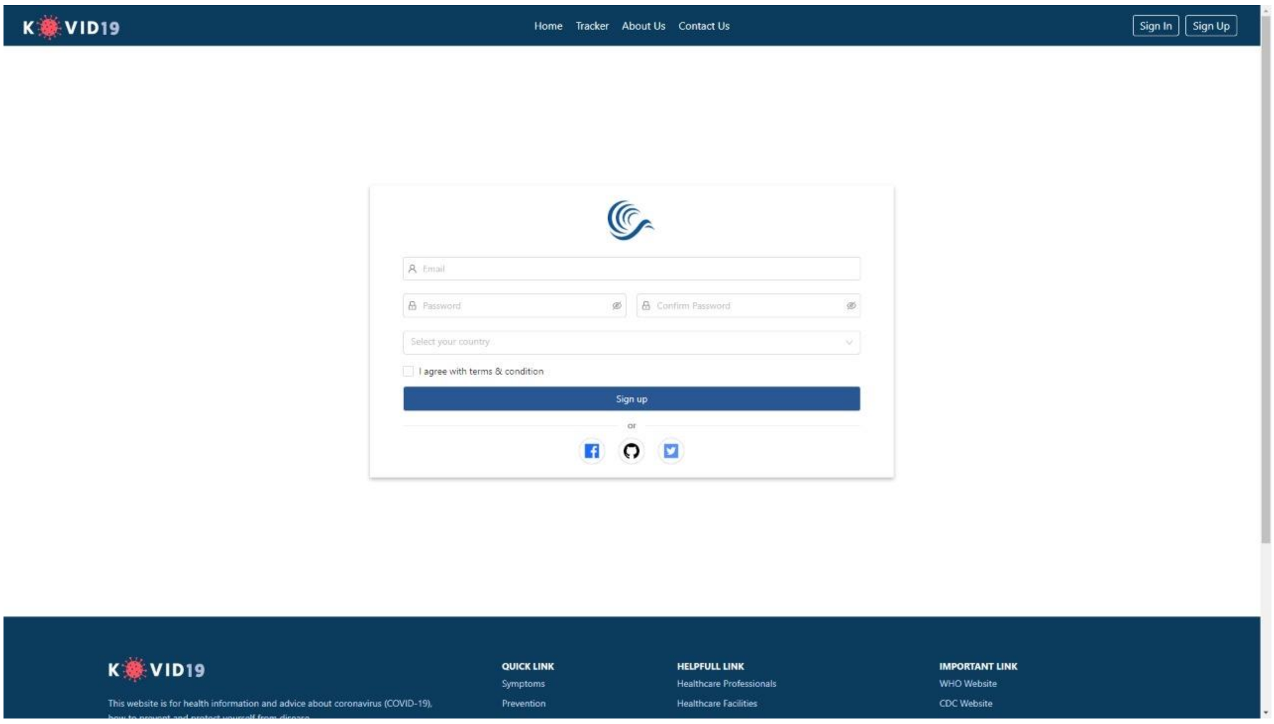
Remember me Forgot your password

Sign in

Don't have an account? Sign up

This website is for health information and advice about coronavirus (COVID-19),

**Fig.3.5.3 Sign Up Page**



This website is for health information and advice about coronavirus (COVID-19),

**Fig.3.5.4 Contact Page**

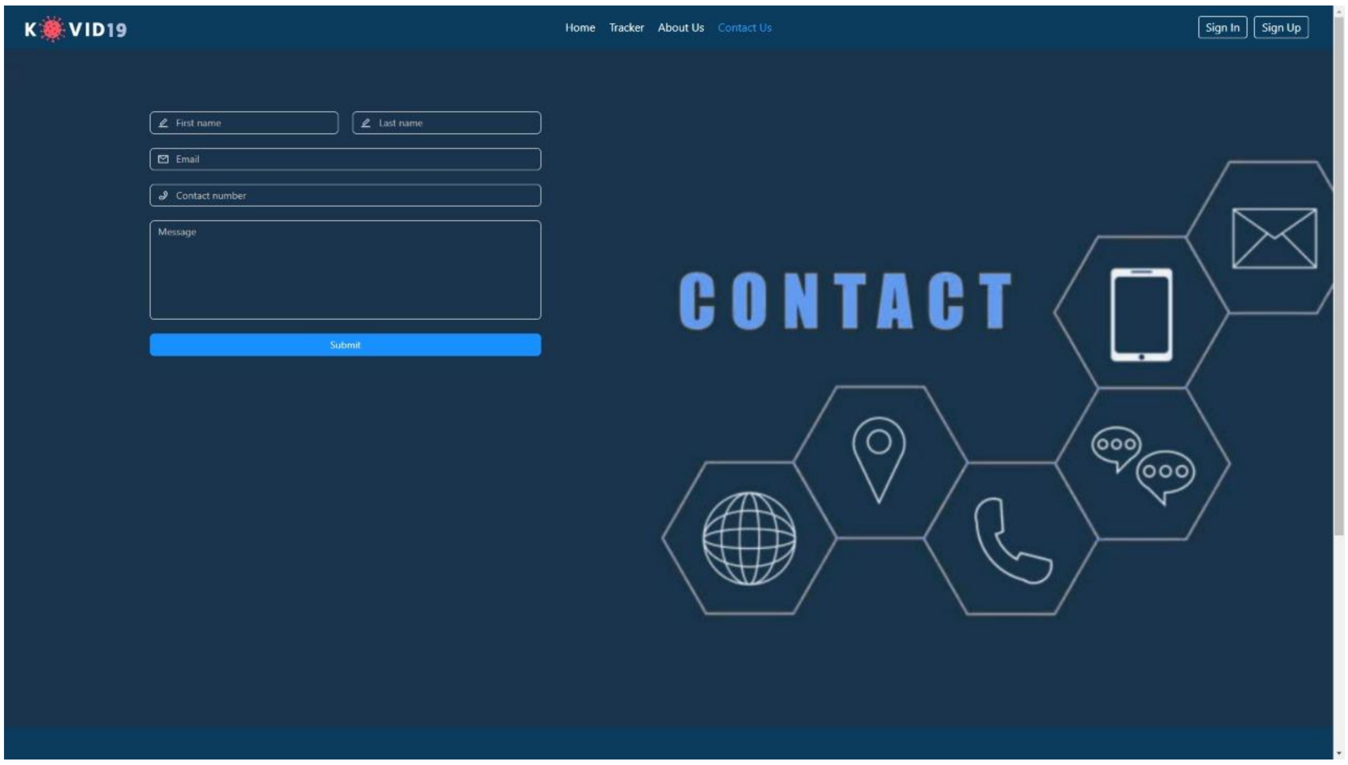
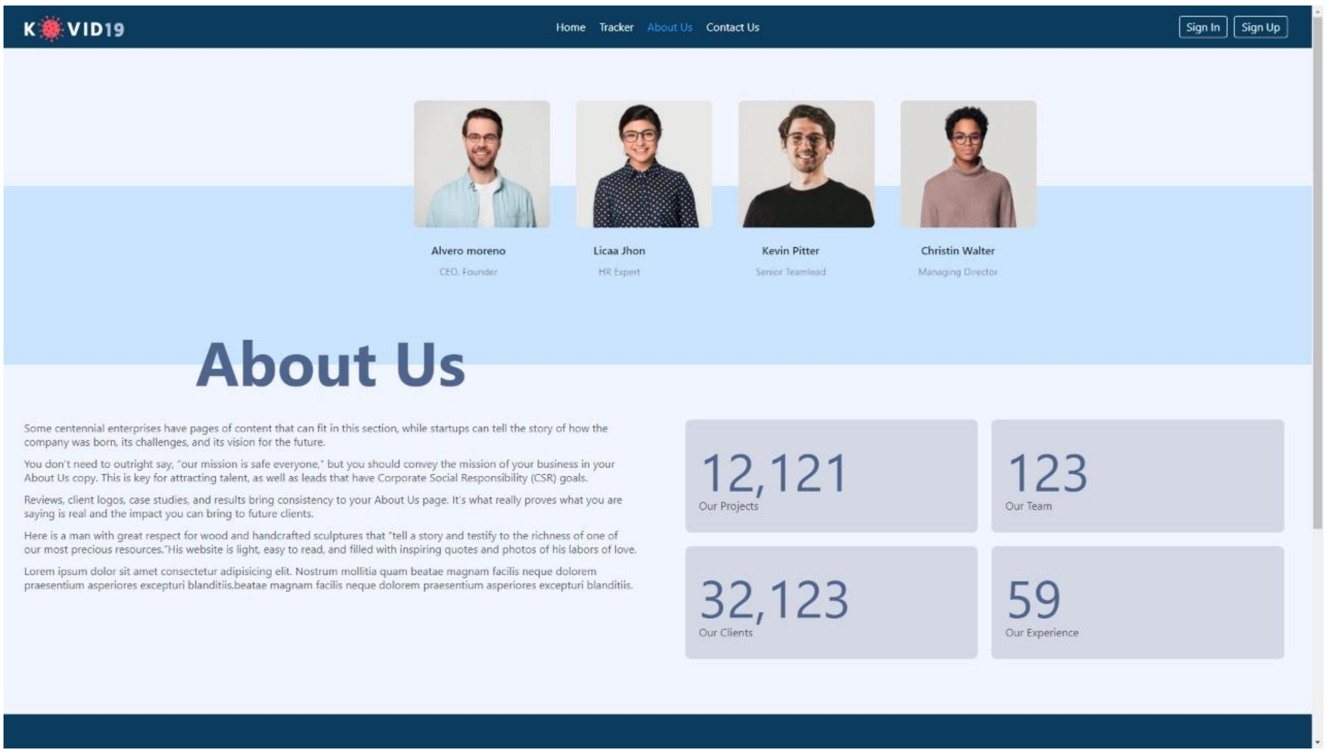


**Fig.3.5.5 About Us  Page**



About Us

Some centennial enterprises have pages of content that can fit in this section, while startups can tell the story of how the company was born, its challenges, and its vision for the future.

You don't need to outright say, "our mission is safe everyone," but you should convey the mission of your business in your About Us copy. This is key for attracting talent, as well as leads that have Corporate Social Responsibility (CSR) goals.

Reviews, client logos, case studies, and results bring consistency to your About Us page. It's what really proves what you are saying is real and the impact you can bring to future clients.

Here is a man with great respect for wood and handcrafted sculptures that "tell a story and testify to the richness of one of our most precious resources."His website is light, easy to read, and filled with inspiring quotes and photos of his labors of love.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Nostrum mollitia quam beatae magnam facilis neque dolorem praesentium asperiores excepturi blanditiis.beatae magnam facilis neque dolorem praesentium asperiores excepturi blanditiis.

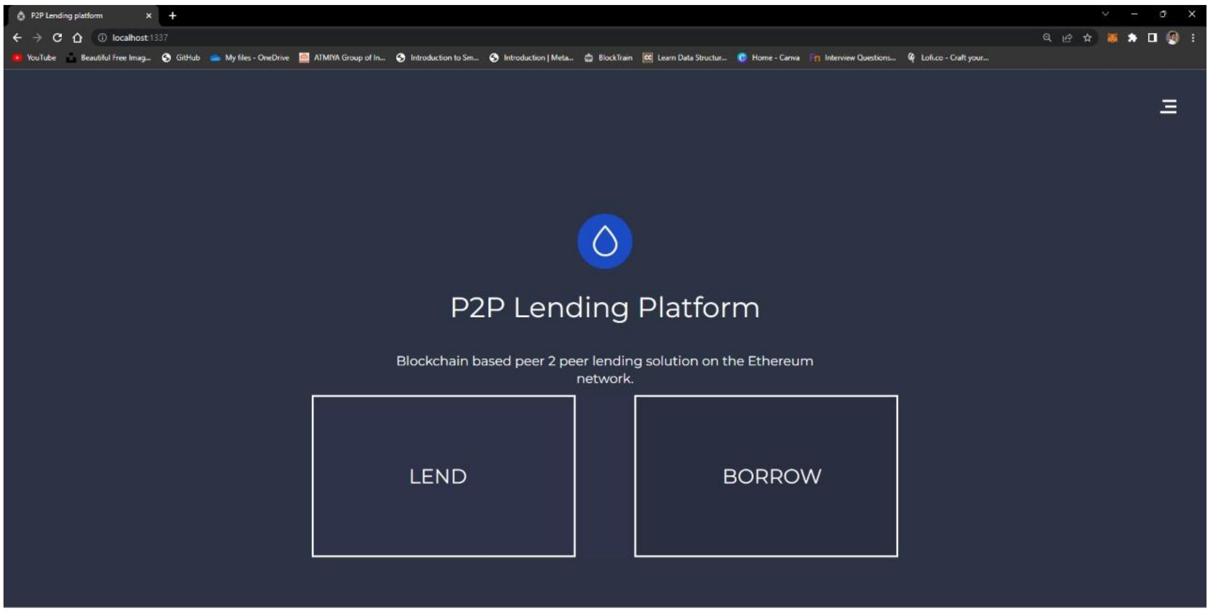
12,121
Our Projects

123
Our Team

32,123
Our Clients

59
Our Experience

**Fig.3.5.6 Lending Platform Page**



P2P Lending Platform

Blockchain based peer 2 peer lending solution on the Ethereum network.

# CHAPTER 4

# IMPLEMENTATION DETAILS

In this Section we will do Analysis of Technologies to use for implementing the project.

## 4.1 FRONT END

Node.js is a runtime environment used for executing server-side code with higher efficiency and it presents a larger bandwidth to handle large code payloads.

## 4.2 BACK END

### RDBMS TERMINOLOGY

Before we proceed to explain MySQL database system, let's revise few definitions related to database.

**Database**: A database is a collection of tables, with related data.

**Table**: A table is a matrix with data. A table in a database looks like a simple spadsheet.

**Column**: One column (data element) contains data of one and the same kind, for example the column postcode.

**Row**: A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.

**Redundancy**: Storing data twice, redundantly to make the system faster.

**Primary Key**: A primary key is unique. A key value cannot occur twice in one table. With a key, you can find at most one row.

**Foreign Key**: A foreign key is the linking pin between two tables.

**Compound Key**: A compound key (composite key) is a key that consists of multiple columns because one column is not sufficiently unique.

**Index**: An index in a database resembles an index at the back of a book.

**Referential Integrity**: Referential Integrity makes sure that a foreign key value always points to an existing row.

# CHAPTER 5

# TESTING AND IMPLEMENTATION

The term implementation has different meanings ranging from the conversation of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old system to new. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Proper implementation is essential to provide a reliable system to meet organization requirement.

## 5.1 UNIT TESTING

### 5.1.1 Introduction

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

### 5.1.2 Benefits

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

**1) Find problems early :** Unit testing finds problems early in the development cycle. In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

**2 ) Facilitates Change :** Unit testing allows the programmer to refactor code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

**3 ) Simplifies Integration :** Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier.

**4 ) Documentation :** Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface (API).Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

## 5.2 INTEGRATION TESTING

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing**.**

### 5.2.1 Purpose

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using black-box testing, success and error casesbeing simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests. The cross-dependencies for software integration testing are: schedule for integration testing, strategy and selection of the tools used for integration, define the cyclomatical complexity of the software and software architecture, reusability of modules and life-cycle and versioningmanagement.Some different types of integration testing are big-bang, top-down, and bottom-up, mixed (sandwich) and risky-hardest. Other Integration Patterns[2] are: collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and high-frequency integration.

## 5.3 SOFTWARE VERIFICATION AND VALIDATION

### 5.3.1 Introduction

In software project management, software testing, and software engineering, verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as software quality control. It is normally the responsibility of software testers as part of the software development lifecycle. Validation checks that the product design satisfies or fits the intended use (high-level checking), i.e., the software meets the user requirements.This is done through dynamic testing and other forms of review.Verification and validation are not the same thing, although they are often confused. Boehm succinctly expressed the difference between

Validation : Are we building the right product?
Verification : Are we building the product right?
According to the Capability Maturity Model (CMMI-SW v1.1)

Software Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

Software Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.

In other words, software verification is ensuring that the product has been built according to the requirements and design specifications, while software validation ensures that the product meets the user's needs, and that the specifications were correct in the first place. Software verification ensures that "you built it right". Software validation ensures that "you built the right thing". Software validation confirms that the product, as provided, will fulfill its intended use.

From Testing Perspective

Fault – wrong or missing function in the code.
Failure – the manifestation of a fault during execution.
Malfunction – according to its specification the system does not meet its specified functionality
Both verification and validation are related to the concepts of quality and of software quality assurance. By themselves, verification and validation do not guarantee software quality; planning, traceability, configuration management and other aspects of software engineering are required.Within the modeling and simulation (M&S) community, the definitions of verification, validation and accreditation are similar:

M&S Verification is the process of determining that a • computer model, simulation, or federation of models and simulations implementations and their associated data accurately represent the developer's conceptual description and specifications.
M&S Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s).

### 5.3.2 Classification of Methods

In mission-critical software systems, where flawless performance is absolutely necessary, formal methods may be used to ensure the correct operation of a system. However, often for non-mission-critical software systems, formal methods prove to be very costly and an alternative method of software V&V must be sought out. In such cases, syntactic methods are often used.

### 5.3.3 Test Cases

A test case is a tool used in the process. Test cases may be prepared for software verification and software validation to determine if the product was built according to the requirements of the user. Other methods, such as reviews, may be used early in the life cycle to provide for software validation.

## 5.4 Black-Box Testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well.

### 5.4.1 Test Procedures

Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of how the software produces the output in the first place.

### 5.4.2 Test Cases

Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output, often with the help of an oracle or a previous result that is known to be good, without any knowledge of the test object's internal structure.

## 5.5 White-Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

## 5.5.1 Levels

**1 ) Unit testing :** White-box testing is done during unit testing to ensure that the code is working as intended, before any integration happens with previously tested code. White-box testing during unit testing catches any defects early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.

**2 ) Integration testing :** White-box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness of the  behaviorin an open environment through the use of white-box testing for any interactions of interfaces that are known to the programmer.

**3 ) Regression testing :** White-box testing during regression testing is the use of recycled white-box test cases at the unit and integration testing levels.

## 5.5.2 Procedures

White-box testing's basic procedures involves the tester having a deep level of understanding of the source code being tested. The programmer must have a deep understanding of the application to know what kinds of test cases to create so that every visible path is exercised for testing. Once the source code is understood then the source code can be analyzed for test cases to be created. These are the three basic steps that white-box testing takes in order to create test cases:

Input involves different types of requirements, functional specifications, detailed designing of documents, proper source code, security specifications. This is the preparation stage of white-box testing to layout all of the basic information.
Processing involves performing risk analysis to guide whole testing process, proper test plan, execute test cases and communicate results. This is the phase of building test cases to make sure they thoroughly test the application the given results are recorded accordingly.
Output involves preparing final report that encompasses all of the above preparations and results.

## 5.5.3 Advantages

White-box testing is one of the two biggest testing methodologies used today. It has several major advantages:

Side effects of having the knowledge of the source code is beneficial to thorough testing.
Optimization f code by revealing hidden errors and being able to remove these possible defects.
Gives the programmer introspection because developers carefully describe any new implementation.
Provides  traceability of tests from the source, allowing future changes to the software to be easily captured in changes to the tests.
White box testing give clear, engineering-based, rules for when to stop testing.

## 5.5.4 Disadvantages

Although white-box testing has great advantages, it is not perfect and contains some disadvantages:

White-box testing brings complexity to testing because the tester must have knowledge of the program,

including being a programmer. White-box testing requires a programmer with a high level of knowledge due to the complexity of the level of testing that needs to be done.
On some occasions, it is not realistic to be able to test every single existing condition of the application and some conditions will be untested.
The tests focus on the software as it exists, and missing functionality may not be discovered.

## 5.6 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

# CHAPTER 6

## LIMITATIONS

Though we tried our best in developing this system but as limitations are mere parts of any system so are of our system. Some limitations of health advisor are:-

- Online payment is not available at this version
- Data delete and edit system is not available for all section
- User account verification by mobile sms is not available in this system
- Loss of data due to mismanagement.

# CHAPTER 7

# CONCLUSION

Health Advisor as an industry as an infant stage. It has long way to go. It has a bright future. This industry is growing at reasonably quick rate. There should also be a set of standard for the events that are being conducted. In the near future, the companies form a network associations in different stages in order to expand the industry so that events can be held professionally.

# CHAPTER 8

## REFERENCES

Google browser
https://www.quora.com
https://www.w3schools.com
https://www.tutorialspoint.com
https://www.javatpoint.com/nodejs-tutorial