**Test plan for**

**KrishiKom**

Made by:

Shivam Sharma

(2000290120150)

Shivam Nautiyal

(2000290120149)

Test Manager:  
Prof. Shreela Pareek

Project Co-ordinator:  
Prof. Rahul Chauhan

Subitted to:  
Prof. Neha Shukla

1. [INTRODUCTION 2](#_TOC_250010)
   1. [Scope 2](#_TOC_250009)
      1. [In Scope 2](#_TOC_250008)
      2. [Out of Scope 2](#_TOC_250007)
   2. [Quality Objective 3](#_TOC_250006)
   3. [Roles and Responsibilities 3](#_TOC_250005)
2. [TEST METHODOLOGY 3](#_TOC_250004)
   1. [Overview 3](#_TOC_250003)
   2. [Test Levels 4](#_TOC_250002)
   3. [Test Completeness 4](#_TOC_250001)
3. [TEST DELIVERABLES 4](#_TOC_250000)
4. RESOURCE & ENVIRONMENT NEEDS 5
   1. Testing Tools 5
   2. Test Environment 5

# Introduction

KrishiKom is a versatile mobile application designed to offer essential agricultural resources to Indian farmers. Developed using React Native, it provides multilingual support and easy UI, ensuring accessibility. Beyond delivering agricultural information, the app focuses on fostering farming communities. It enables farmers to connect, share insights, and build a network for knowledge exchange and support.

## Scope

### **In Scope**

Scope defines the features, functional or non-functional requirements of the software that **will be** tested. Features of the Project:

1. Community Portal ModuleFunctional Requirements:Test the functionality of creating and participating in community discussions.Verify the accuracy and responsiveness of notifications for community interactions.Non-Functional Requirements:Evaluate the user-friendliness and intuitiveness of the community interface.Assess the speed and reliability of community updates and data refresh.
2. Change Language ModuleNon-Functional Requirements:Test language switch functionality and validate language-based content accuracy.3. Weather ModuleFunctional Requirements:Validate real-time weather updates and accuracy of information provided.Non-Functional Requirements:Assess the usability and readability of weather data for users.
3. Resource Sharing ModuleFunctional Requirements:Test the sharing mechanism and confirm its usability and effectiveness.Non-Functional Requirements:Evaluate the user-friendliness and accessibility of the sharing platform.
4. Market ModuleFunctional Requirements:Verify the accuracy and relevance of global market trend analysis.Non-Functional Requirements:Evaluate the ease of navigation and comprehensibility of market trend data.
5. Farmer News ModuleFunctional Requirements:Test the retrieval and display of latest news relevant to farming communities.Non-Functional Requirements:Evaluate the readability and accessibility of news content.
6. Plant Disease Detection ModuleFunctional Requirements:Test the functionality of uploading images within the "Plant Disease Detection" module.Verify the format verification and validation for images uploaded for disease identification.Ensure the accuracy of disease recognition and identification in plants and materials.Non-Functional Requirements:Validate the provision of detailed descriptions of recognized plants and materials to ensure accuracy and comprehensiveness.
7. Structure Chart and Plan for Crops ModuleFunctional Requirements:Test the functionality and accuracy of creating crop plans using the structure chart feature.Non-Functional Requirements:Evaluate the clarity and comprehensiveness of crop planning tools and diagrams.
8. Government Schemes ModuleFunctional Requirements:Verify the accuracy and relevance of information related to government schemes.Non-Functional Requirements:Assess the ease of understanding and access to scheme details.

## Quality Objective

##### Here make a mention of the overall objective that you plan to achive without your testing Some objectives of your testing project could be

##### Reliability and Stability

##### Usability and User Experience

##### Scalability and Future Expansion

##### Functional Accuracy

##### Performance and Response Time

# Test Methodology

## Overview

##### We are using an iterative testing approach to make sure our project works well. This means we test it in small steps, starting with checking if each part works on its own. Then, we see how different parts work together.

##### We keep testing as we make changes and add new things. This way, we make sure our project is always working well, even after modification.

## Test Levels

**Test Levels define the Types of Testing to be executed on the Application Under Test (AUT**). We aim to test our project at the following levels :

##### Unit Testing: This is the lowest level of testing and focuses on individual components or functions within the software. Developers often perform unit tests to verify that specific parts of the code work correctly.

##### Integration Testing: This level of testing checks how different components or modules of the software work together. It ensures that integrated parts of the software function as intended.

##### System Testing: At this level, the entire system is tested as a whole. It verifies that the software meets its specified requirements and functions properly in its intended environment.

## Test Completeness

##### Here you define the criterias that will deem your testing complete. For instance, a few criteria to check Test Completeness would be

##### 100% test coverage

##### All Manual & Automated Test cases executed

##### All open bugs are fixed or will be fixed in next release

# Test Deliverables

##### Here are the deliverables

##### Test Plan

##### Test Cases

##### Bug Reports

#####  Test Strategy

1. **Test Cases**

**Plant Disease Detection Module:**

**A screenshot of a computer

Description automatically generated**

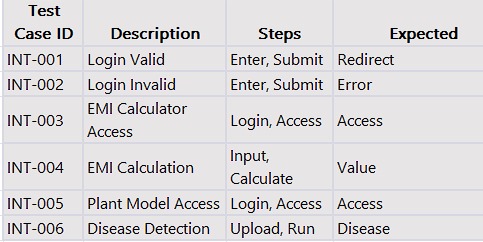
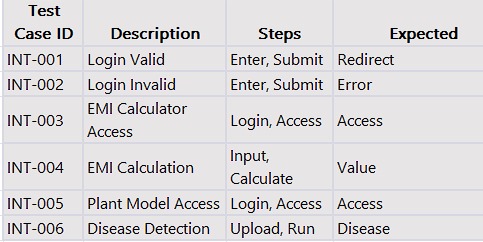
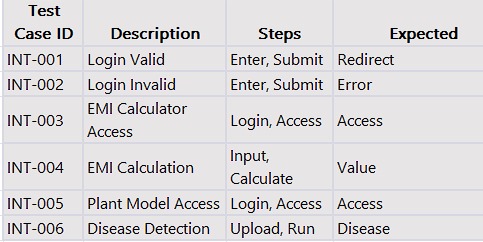
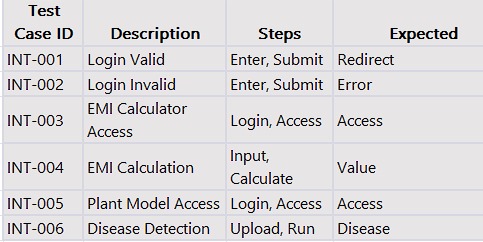
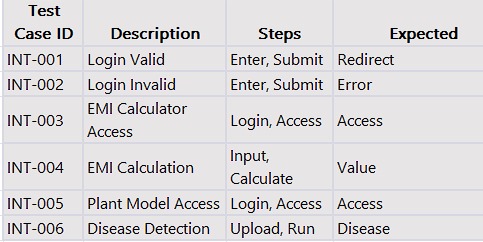
**Emi Calculation Module:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **sno** | **Principal Amount** | **Time** | **Rate Of Interest** | **Expected output** | **Actual Output** | | 1 | Positive | Positive | Positive | Calculated | Calculated | | 2 | Positive | Positive | Negative | Error Shown | Error Shown | | 3 | Negative | Positive | Negative | Error Shown | Error Shown | | 4 | Negative | Negative | Negative | Error Shown | Error Shown | | 5 | Positive | Positive | Positive | Error Shown | Error Shown | | 6 | Positive | Negative | Positive | Error Shown | Error Shown | | 7 | Negative | Negative | Positive | Error Shown | Error Shown | | 8 | Negative | Negative | Negative | Error Shown | Error Shown | |

**Login Module Decision Table:**

|  |
| --- |
| Using Decision Tables for Clear, Well-Designed Testing | StickyMinds |

**Integration Testing**

     A screenshot of a computer

Description automatically generated

1. **Resource & Environment Needs**
   1. **Testing Tools**

##### List of Tools like

##### Selenium

##### Android Studio

##### Postman

##### LoadRunner

* 1. **Test Environment**

##### It mentions the minimum **hardware** requirements that will be used to test the Application. Following **software’s** are required in addition to client-specific software.

##### Windows 10 and above preferred

##### VSCode 2022 or above preferred

##### Android Phone Preferred