VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018

A PROJECT REPORT **(15CSP85)**

ON

"FARMERS CROWD FUNDING"

A Portable Solution for Preventing Suicide Death's of Farmers in India.

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

M AISHWARYA (USN: 1CR16CS077)

SONU S BABU (USN: 1CR16CS163)

SUCHITRA I M (USN: 1CR16CS166)

SURAJ R BELLAD (USN: 1CR16CS168)

Under the Guidance of,

Dr. Shanthi M. B.

Associate Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"FARMERS CROWD FUNDING - A Portable Solution for Preventing Suicide deadth's of Farmers"** carried out by **M AISHWARYA (**USN:**1CR16CS077)**, **SONU S BABU (**USN:**1CR16CS163)**, **SUCHITRA I M (**USN:**1CR16CS166), SURAJ R BELLAD (**USN:**1CR16CS168),** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Dr. Shanthi M. B	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Associate Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners — Signature with date

1. _____

2. _____

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled "**FARMERS CROWD FUNDING - A Portable Solution for Preventing Suicide deadth's of Farmers**" has been successfully completed under the guidance of Dr. Shanthi M B, Associate Professor, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date:

Team members:

M AISHWARYA (USN:1CR16CS007) _____

SONU S BABU (USN:1CR16CS163) _____

SUCHITRA I M (USN:1CR16CS166) _____

SURAJ R BELLAD (USN:1CR16CS168) _____

ABSTRACT

India is a land of agriculture and farmers are indeed the backbone of our nation. The entire human cycle has to be dependent on this sector directly or indirectly. Over the past decades, Farmer's suicide in different parts of India has become a recurring phenomenon. The beginning of a such crisis led to enormous farmers' movement in various parts of India began demanding remunerative prices, writing off loans, etc. The recent years of advancement in Globalization, the problem faced my farmers and the agricultural sector has remained constant. In-order to approach the above problem statement with the help of technological tools, an Android based application is developed targeting the major concerns of farmers and their lifestyle. Kissan Rakshak application is designed to reduce the complexities of the issues faced by the farmers and those who are dependent on this sector. For the progress of our nation, the success and well-being of farmers is of utmost importance.

ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me a platform to pursue my studies and carry out my final year project

I have a great pleasure in expressing my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

I would like to thank **Dr. Prem Kumar Ramesh,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

I consider it a privilege and honor to express my sincere gratitude to my project guide **Dr. Shanthi M B,** Associate Professor **,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

I also extend my thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, I would like to thank my parents and friends for all their moral support they have given me during the completion of this work.

TABLE OF CONTENTS

	Page No.
Abstract	**i**
Acknowledgement	**ii**
Table of contents	**iii**
List of Figures	**v**
List of Abbreviations	**vi**
1 INTRODUCTION	**1**
1.1 Case Study of India	**2**
1.1.1 Case Study of Karnataka	**4**
1.1.2 Corona impact on Agriculture	**5**
1.2 Existing System	**6**
1.3 Proposed System	**6**
1.4 Problem Statement	**7**
1.5 Relevance of the Problem	**7**
1.6 Objective of Project	**7**
2 LITERATURE SURVEY	**8**
3 TECHOLOGIES USED	
3.1 Overview on Android Technology	**11**
3.2 Overview on Blockchain Technology	**12**
3.2.1 BlockChain for Transactions	**12**
3.2.2 BlockChain Terminologies	**13**
4 SYSTEM REQUIREMENTS	
4.1 Functional Requirements	**14**
4.2 Non Functional Requirements	**14**
4.3 System Configuration	**15**

5 SYSTEM ANALYSIS AND DESIGN

5.1 System Development Methodology 16

5.2 Data flow diagram 18

5.3 System Architecture 19

6 IMPLEMENTATION 20

6.1 Creating Farmers Profile 22

6.2 Creating Funders Profile 23

6.3 Creating Coorporative Profile 24

6.4 Integrating BlockChain for Transactions 25

7 TESTING

7.1 Testing Methodologies 26

7.1.1 Functional Testing 26

7.1.2 Compatibility Testing 26

7.1.3 Security Testing 27

7.1.4 Manual Testing 27

7.1.5 Interface Testing 27

8 BENEFITS AND OUTCOMES 28

9 RESULTS AND DISCUSSIONS 29

10 CONCLUSION AND FUTURE SCOPE 33

References

Appendix

LIST OF FIGURES

	Page No.
Fig 1.1 A Bar graph representing Suicide rates in Agriculture Sector in India over the years.	3
Fig 1.2 The graph plotted with year vs population which shows the decline employment rate from the year 2008 – 2020.	3
Fig 1.3 Newspaper article of the year 2018-19 drought	4
Fig 1.4 More than half (56%) of the farmers said that the lockdown has impacted their ability to prepare for the upcoming sowing season	5
Fig 3.1 Blockchain Technology	14
Fig 5.1 Project Plan	16
Fig 5.2 Methodology	17
Fig 5.3 Data Flow Diagram	18
Fig 5.4 System Architecture Diagram	19
Fig 6.1 The Basic Flow of Application in UI/UX view	20
Fig 9.1 Kisaan Rakshak Application Login Screen	29
Fig 9.2 Kissan Rakshak Application Register and Edit profile	30
Fig 9.3 Kissan Rakshak Farmer Screen	30
Fig 9.4 Kissan Rakshak Funders Screen	31
Fig 9.5 Policies, Insurance and Help Forum Screen	31
Fig 9.6 About Kissan Rakshak Screen	32
Fig 9.7 Blockchain integrated Banking Interface	32

LIST OF ABBREVIATIONS

ABI	**Application Binary Interface**
DFD	**Data Flow Diagram**
GIC	**General Insurance Corporation**
ICT	**Information and Communication Technologies**
QA	**Quality Assurance**
RCC	**Rural Credit Cooperatives**
SRS	**System Requirement Specifications**
UI	**User Interface**

CHAPTER 1

INTRODUCTION

India is a land of agriculture and farmers are indeed the backbone of our nation. Agriculture is the main occupation in India and about 70% of the population is engaged in this activity. All the other sectors depends directly or indirectly depend on agriculture for diverse reasons. The concept of farming and agriculture has undergone a sea change by adopting latest trends in technology over the years. With various amendments in agricultural schemes and policies, the availability of resources and productivity has improved over the years. A look back at Green Revolution, indicates a record output of 131 million tons in the year 1978 and 1979. This event has established India as one of the world's biggest agricultural producers. But on the darker side,the farmers suicide rate in India that ranged between 1.4 and 1.8 per 1,00,000 total population with an average of 10 suicides per day was witnessed. This led to a mammoth agricultural crisis in major parts on India. Solving this problem and identifying the root cause involved each stage consumed lot of time and tireless efforts. The research and reports depict the overall increment in productivity with latest advancement in technology. The current study showcases the effortsput to cut down the farmer's suicide rates but solving the problem from the root level is quite challenging even today. The unheard voice of farmers and their problems has begun to take newer shapes with the present scenario and the grass root level solution are left unsolved.

Approaching to this problem statement, Kissan Rakshak - a Crowd Funding Android based application that's developed to tackle the problem of farmers from intermediaries and raise the value of the crop yield.This project involves users like investors who are capable and interested to fund the farmers in need, supermarkets that takes care of buying and selling of yield and farmers who need financial support. This project definitely deals with one of the most critical issue in Agriculture Category.Further, this system simplifies the traditional way of lending money to farmers who are in need of financial support and ensures high value is obtained for the farmers yield.With various effective features and easy to use interfaces, the application is kept simple so that the

farmers can get the best possible help. It involves secure login for farmers who are registered with organization referred as Corporative Society which are established for the well-being of farmers. This application acts like a bridge between the farmers and funders who are ready to invest in agriculture sector. The main theme of Kissan Rakshak is to support both the farmers as well as funders in every single aspect and improve their lifestyle. The complexity is reduced a lot as farmers get the privilege to access money from Corporative Society which is funded by Funders. The rate of interest on loans granted by various banks for loans will not affect the farmer any more as the farmers will prefer to contact Corporative society rather than banks for financial aid. On the other end, Funders will be benefited through the cultivated crops given by farmers or by the direct exchange of money.

In order to reduce the complexities of the issues faced by the farmers, the idea of farmer crowd funding originated.Crowd Funding is coined for its amazing practical solution to generate and manage the funds.

1.1 Case Study of India

Over the past couple of years, the issue of farmers suicide in different parts of the world, including India has slowly become a major issue. The Central Governments, State Governments and the farming community have succeeded in achieving record production of 244.78 million tons of food grains during 2010-11 but it's not sufficient for the community to enhance their standard of living.This rightly indicates that there is an immense requirement of financial aid necessary for agriculture in India. Although India has attained self-sufficiency in food staples, the productivity of the farms is below that of Brazil, the United States, France and other nations.One study suggests that Indian agricultural policy should best focus on improving rural infrastructure primarily in the form of irrigation and flood control infrastructure, knowledge transfer of better yielding and more disease resistant seeds. Additionally, cold storage, hygienic food packaging and efficient modern retail to reduce waste that can improve output and rural incomes.

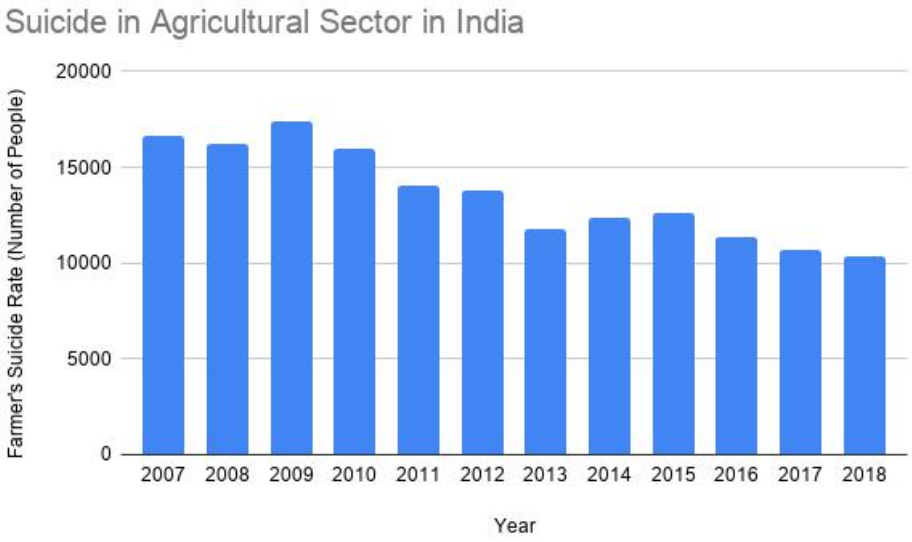


Fig 1.1 A Bar graph representing Suicide rates in Agriculture Sector in India over the years. (Source: plotted with available data from NCRB data)

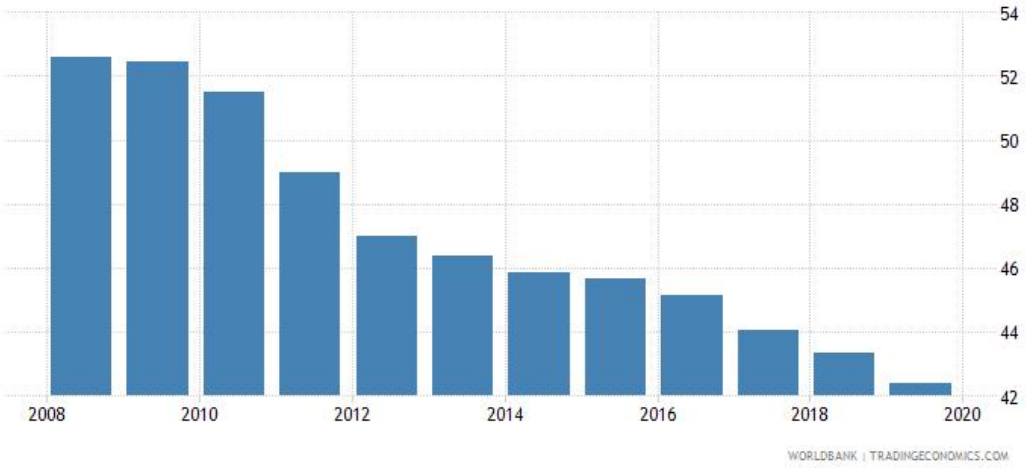
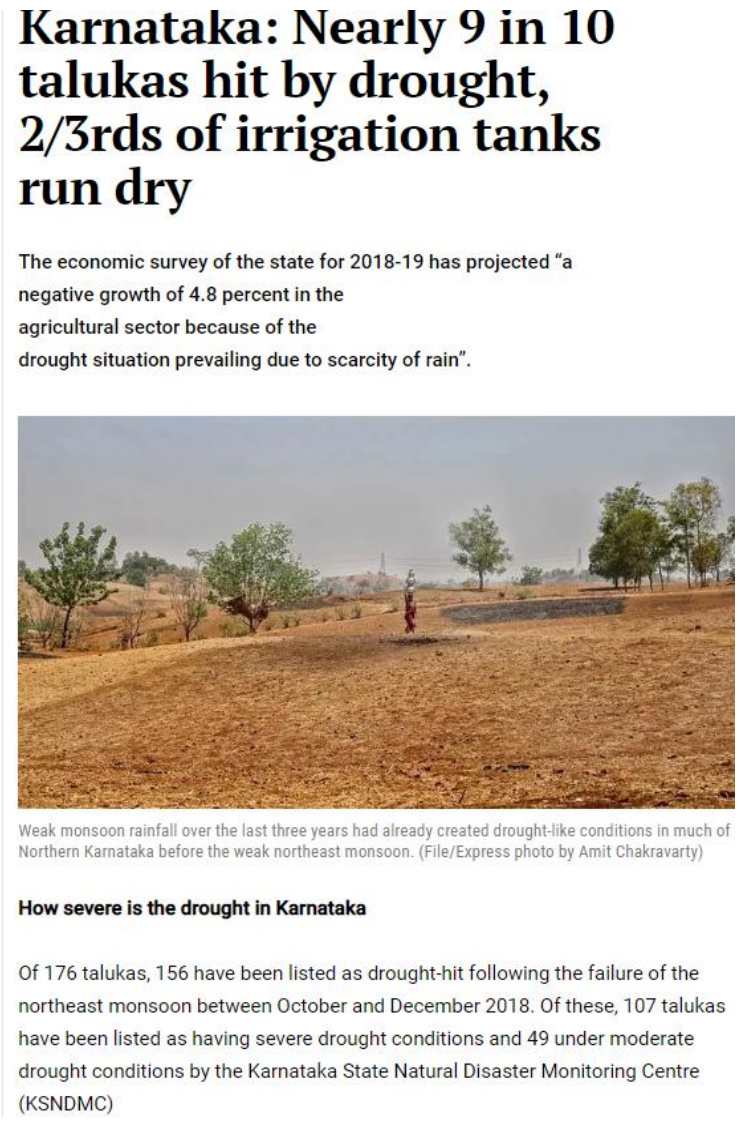


Fig 1.2 The graph plotted with year vs population which shows the decline employment rate from the year 2008 – 2020. (Source: Employment in agriculturein India was reported at 42.38 % in 2019, according to the World Bank collection of development indicators)

1.1.1 Case Study of Karnataka

Characteristics of Karnataka agriculture has changed over the past couple of years— a transition from non-capitalist path to agrarian capitalist path.Agrarian capitalism received a boost with the introduction of Green Revolution.Nonetheless, it is true that Karnataka is one of the fastest growing states in which the agrarian economy contributes 25 per cent of States' GSDP and employees more than 75 per cent of rural population. In 2018 alone 143 talukas, which went up to 107 in the subsequent year, were declared as drought areas.



Karnataka: Nearly 9 in 10 talukas hit by drought, 2/3rds of irrigation tanks run dry

The economic survey of the state for 2018-19 has projected "a negative growth of 4.8 percent in the agricultural sector because of the drought situation prevailing due to scarcity of rain".

Weak monsoon rainfall over the last three years had already created drought-like conditions in much of Northern Karnataka before the weak northeast monsoon. (File/Express photo by Amit Chakravarty)

How severe is the drought in Karnataka

Of 176 talukas, 156 have been listed as drought-hit following the failure of the northeast monsoon between October and December 2018. Of these, 107 talukas have been listed as having severe drought conditions and 49 under moderate drought conditions by the Karnataka State Natural Disaster Monitoring Centre (KSNDMC)

Fig 1.3 Newspaper article of the year 2018-19 drought

1.1.2 Corona Impact on Agriculture

Preliminary findings of a survey of more than 1,000 agricultural households across 12 Indian states show that 60% of those who did harvest reported a yield loss, and 1/10th of them could not harvest their crop in the past month. More than half (56%) of the farmers said that the lockdown has impacted their ability to prepare for the upcoming sowing season. The survey has found a "very strong association between food insecurity & farm size, with landless farmers 10 times more likely to skip a meal in the past month compared to large farmers". Half of them reported that they were "concerned about being able to afford inputs, particularly seeds and fertilizer," and more than one-third (38%) were concerned about labor shortages. About one-fourth said they were "storing their crops instead of selling them due to the lockdown". The study has found small/marginal farmers were significantly less likely to be able to sell their crops as compared to large farmers. The situation of farmers are really terrible.

Fig 1.4 More than half (56%) of the farmers said that the lockdown has impacted their ability to prepare for the upcoming sowing season.(Source from Indian Express newspaper update on May 22, 2020 6:50:09 am)

10% of farmers could not harvest their crop in the past month and 60% of those who did harvest reported a yield loss. Many reported that this was because of lockdown-related issues such as low market price or inability to access their land due to travel restrictions. Several farmers also reported severe weather and water scarcity/lack of irrigation, which are persistent challenges in the agricultural sector and will need to address as the slower-burning climate crisis continues in the midst of the pandemic.

1.2 Existing System

The existing systems includes the loans provided by banks where the rate of interest is high and eventually becomes difficult for the farmers to repayand the involvement of local landlords. The process of getting loan from the bank is very tedious. It requires a lot of paper work and they will have to submit proofs for mortgage. There is no direct interaction between the customers and the farmers. Therefore, the intermediates are the money makers. It is very time-consuming process and not every farmer is legible to apply.Documentation makes it a complicated task for many farmers who want to get benefits from Banks and other systems.

1.3 Proposed System

In the proposed system, i.e., through the application ,there is no intervention of banks and other local bodies. The entire concept of Interest Rates and Mortgage is eliminated.Indeed the Application makes it easier for serving this process.Farmers are paid for what they produce and the customers are satisfied with what they invest.Governmental Policies and Schemes eliminates the risk of investment and loses.The application helps Farmers to yield crops to maximum of the productivity with the help of Latest Techniques and Methods. It is completely user-friendly platform and requires very less efforts to operate.With the advancement in Block Chain technology, the transactions are made very secure and faster.Numerous Features enables smooth functioning of the application.The entire data will be maintained safe and securely in the database of the application.

In order to ease and reduce the complexities of the issues faced by the farmers, the idea of farmer crowd funding originated. The aim of crowd funding is to tackle the problem of intermediaries and raise the value of farmer's yield. The project involves users like investors who are capable and interested to fund the farmers in need, supermarkets that take care of buying and selling of yield and farmers who need financial support. This system simplifies the traditional way of lending money for farmers who are in need of financial aid and ensure high value for the farmers yield.

1.4 Problem Statement

Since times immemorial, farmers' suicide in different parts of India has become a recurring phenomenon,however it reflects the deep-rooted agrarian crisis. The beginning of such crisis can be traced back to the decade of 1980s when farmers' movement in different parts of India began demanding remunerative prices, writing off loans, etc. In fact, all these crises have translated in the form of farmers' committing suicide. Thus, the team is motivated to develop an Application, which acts as a platform that implements Crowdfunding for farmers using Block Chain - A portable solution for preventing suicide deaths of Farmers in India.

1.5 Relevance of the Problem

Implementing crowd funding to solve farmer suicide is the main objective of the system. The need of the application is high as it is capable of saving the lives of farmers as well as improve the lifestyle of farmers. To support Organic Farming methods and provide relevant updates to farmers on the latest techniques is an additional feature in the application.A prototype of the application, a web system has been developed that caters to the basic functioning of crowd funding. The prototype involves mainly three users, admin, farmer and funder. As a part of literature survey, it is found that the prototype will definitely outstand the existing .NET application and the result of project can contribute to the betterment of farmers.

1.6 Objective of Project

- To focus on eliminating the Intermediates who seemed to be more benefitted from the existingtrend.
- To provide common platform for farmers and funders across thecountry.
- To motivate farmers to indulge themselves in farming sector.
- To make farmers independent and ensure their dependency on traditional system is reduced.
- To provide financial support to famers.

Chapter 2

LITERATURE SURVEY

In order to get required knowledge about various concepts related to the present application, existing literature were studied. Some of the important conclusions were made through those are listed below.

The research put forward in [1], emphasis on the view that Farmers are called backbones of India and when backbone is damaged it is difficult for the country to withstand since food is the basic need for every human being to survive.The main objective of this application is to develop a site especially for farmers to provide interaction with the government. The application is designed with asp.net MVC.Here in this website before cultivating the crop the farmers should enter the details of the land, the crops to be grown and expected cost for the crop.The author hopes that from this application that there will be no farmer suicidesas there will be immediate support from the government. The author concept is to avoid the suicides of farmers as well as support the farmers.In the future, the author wants to include the problems or denote money related to the environmental crisis like floods, cyclones or tsunami occurs there will be a direct transfer of amount for immediate recovery of the city or state.

The author Muzaffar H. Assadi has discussed in [2], about the issue of farmers' suicide in different parts of the world.The path of development has not created debt free farmers. The budgetary allocation does not cross two percent of the total budget. Most interesting finding is the fact that the highest proportion of indebted farmers are belonging to backward communities with 42 percent with farm less than one hectare.Government of Karnataka has put forth many policy measure to avoid suicides like, avoiding money lending from private people.Secondly reducing the interest on cooperative lending to 6 percent.

The authors Valerie Busse and Michal Gregu have discussed in [3], focuses on a well-defined classification on crowdfunding compared to other financing methods. Since

the increasing evolution of information and communication technologies (ICT),the integration of peoples' private, social and professional lives is no longer bound to a certain time or location and is therefore nowadays often replaced through online interactions. Crowdfunding is a good example of this online interaction.Thus, the basic idea of crowdfunding is to raise external funds from a large audience, i.e. "the crowd". In contrast to some highly regulated traditional financing methods, such as Banks, and capital funding enterprises or less regulated traditional financing methods, for Instance venture capital or business angels, the digital platforms for crowdfunding via the Internet are accessible for everyone and therefore easy to reach.The scope is to improve the triadic relationship between entrepreneur, intermediary and crowd.Moreover, research could focus on specific analysis of decisions and empirically explore components and decision-drivers which support successful crowdfunding projects.

The researchers MunetaYokomatsu, Wei-Bin Yu, and Norio Okada have discussed in [4], elaborates the conditions of Rural China suffers from many natural disasters while there is an inadequate supply of disaster risk insurance for Chinese farmers. China suffers a lot from the natural disasters. Reference shows75% of industrial and agricultural production are exposed to serious meteorological, geological and oceanic disasters. Moreover, in 1996, the reform of rural financial system made Chinese Rural Credit Cooperatives (RCCs), a main function of which is to supply farmers with necessary loan for agricultural production. Thus, by making use of the private financing sectors' monitoring ability, farmers are less motivated to go into the moral hazard in disaster risk mitigation.As for the future work, more investigation on circumstantially practical conditions of the inter-linkage market. There are also concerns about the rural enterprises' ability to absorb the capital and surplus work force and try to involve the rural enterprises into the risk financing model in rural China.

The author Lili Zhang has discussed in[5], argued about how Blockchain is changingthe world as it approaches data. Blockchain could be used to store important data and ensure that the data is original as it is distributed and secured. It is a database for applications like managing physical and digital assets, recording internal transactions and verifying identities.As Blockchain technology matures, data management will be one of

the areas that will benefit the most.There are several ways by which blockchain manages data:

1. Data Integrity and Trust
2. Preventing Hacking
3. Performing predictive Analysis
4. Real Time Data Analysis
5. Data Sharing Management and audit Trials

Currently, existing peer to peer Blockchain networks can only process a small amount of data and a handful of transactions per second.Storage methods and processing capability of Blockchain networks will need to be improved, to deal with large volumes of data collected per second.Blockchain technology offers great data integrity advantages for big data including:

1. Greater Confidence in the integrity of data.
2. Immutable entries.
3. Consensus-driven timestamp.
4. Audit trials.
5. Certainty about the origin of data.

Chapter 3

TECHNOLOGIES USED

3.1 Overview on Android Technology

One of the most widely used mobile OS these days is ANDROID. Android Studio is Android's official IDE which is purpose built for Android to accelerate development and help build the highest-quality apps for every Android device. It is an operating system based on the Linux kernel. It was designed primarily for touch screen mobile devices, such as smart phones and tablet computers, with variants for television, cars and wrist wear. It offers tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools. Android Studio uses a Gradle-based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules.

3.2 Overview on Blockchain Technology

A blockchain is a public ledger of all bitcoin transactions that have ever been executed. A block is the current part of a blockchain, which records some or all of the recent transactions, and once completed, goes into the blockchain as permanent database. Each time a block gets completed, a new block is generated. Blocks are linked to each other (like a chain) in proper linear, chronological order with every block containing a hash of the previous block. The full copy of the blockchain has records of every bitcoin transaction ever executed. It can thus provide insight about the fact that banking transactions can be made much more secure and faster.

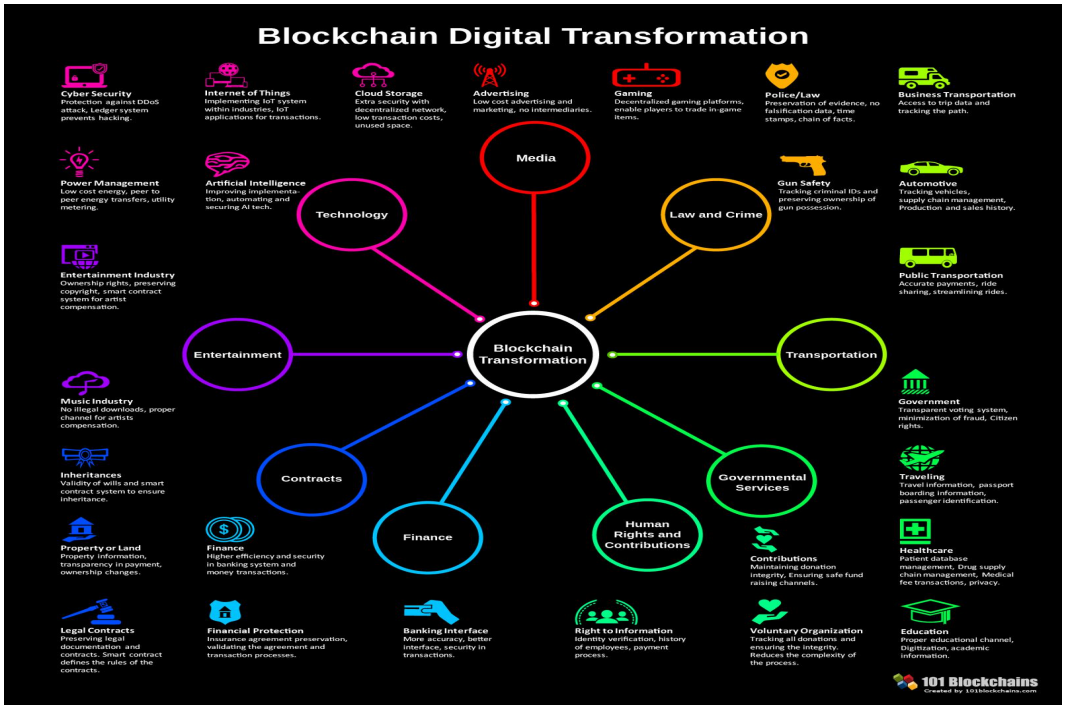


Fig 3.1 Blockchain Technology (Source: 101 Blockchains, Internet)

3.2.1 Blockchain for Transactions

A blockchain is made up of individual blocks of data involving a series of related transactions, linked together in consecutive order. Blockchain technology promises to facilitate fast, secure, low-cost international payment processing services through the use of encrypted distributed ledgers that provide trusted real-time verification of transactions without the need for intermediaries such as correspondent banks and clearing houses.A transaction could flow like this:

1. A wants to send money to B
2. The transaction is represented online as a “block”, a coin or a token in a database.
3. The block is broadcast to every party in the network.
4. After validation, the block is added to the chain, which provides an indelible and transparent record of transaction.
5. The money moves from A to B.

3.2.2 Blockchain Terminologies

Ethereum provides a decentralized virtual machine, the Ethereum Virtual Machine (EVM), which can execute scripts using an international network of public nodes.The virtual machine's instruction set, in contrast to others like Bitcoin Script, is Turing-complete. "Gas", an internal transaction pricing mechanism, is used to mitigate spam and allocate resources on the network.

- Connect to the Ethereum network
- Explore Ethereum's blockchain
- Create new transactions and smart contracts
- Run smart contracts
- Mine for new blocks

Solidity is an object-oriented programming language for writing smart contracts.It is used for implementing smart contractson various blockchain platforms, most notably, Ethereum. Solidity is compiled to bytecode that is executable on the EVM. With Solidity, developers are able to write applications that implement self-enforcing business logic embodied in smart contracts, leaving a non-reputable and authoritative record of transactions.Writing smart contracts in smart contract specific languages such as Solidity is claimed to be easy.

Ganache CLI is a fast and customizable blockchain emulator. It allows to make calls to the blockchain without the overheads of running an actual Ethereum node.

- Transactions are "mined" instantly.
- No transaction cost.
- Accounts can be re-cycled, reset and instantiated with a fixed amount of Ether (no need for faucets or mining).
- Gas price and mining speed can be modified.

Chapter 4

SYSTEM REQUIREMENTS

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

4.1 Functional Requirements

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions.Following are the functional requirements on the system:

1. Secure login to users who use Kissan Rakhsak application.

2. Based on the purpose of the users, various features are enabled

3. The transactions are made secured using Blockchain technology.

4.2 Non Functional Requirements

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours.

1. **Scalability:**It is capable of adding new users and various functionalities. The application also supports various languages which enhances understandability of the application.

2. **Reliability:** The application is more secured with the implementation of Blockchain technology.

3. **Availability:** The application is easily made available as Android phones are used by a greater number of users and is user-friendly.

4.3 System Configuration

Hardware system Configuration:

Processor: Intel Atom® processor or Intel® Core™i3 processor (minimum)

RAM :2 GB(min) or above

Hard Disk :5 GB of free space(minimum)

Display :1024x768 (minimum) screen resolution

Device : Android Phone

Software system Configuration:

Operating System : Windows 10/8

Coding Language : Java and APK above 16

Tools and Frameworks:

- **Android Studio** :Android Studio is the official Integrated Development Environment (IDE) for Android app development

- **jQuery Mobile** : It is a HTML5-based user interface system designed to make responsive web sites and apps that are accessible on all smartphone, tablet and desktop devices.

- **XML** : It stands for Extensible Markup Language, which is used to handle things like static user interface and to define layouts.

- **SQLite Database** : It is a freely available open source Relational Database Management System (RDBMS) that that stores data to a text file on a device.

Chapter 5

SYSTEM ANALYSIS AND DESIGN

Software design is a process through which the requirements are translated into a representation of software. Based on the user requirements and with the detailed analysis of the existing system, the new system must be designed.

5.1 System Development Methodology

System Development methodology is the development of a system or method for a unique situation. System design takes the following inputs:

- Statement of work - The application is designed to address one of the major problem in the agricultural sector.
- Requirement determination plan - Analyzing the importance of problem statement, resource requirements are carefully listed and planned.
- Current situation analysis - Based on previous research and by analyzing the present scenario, the application is designed flexibly.

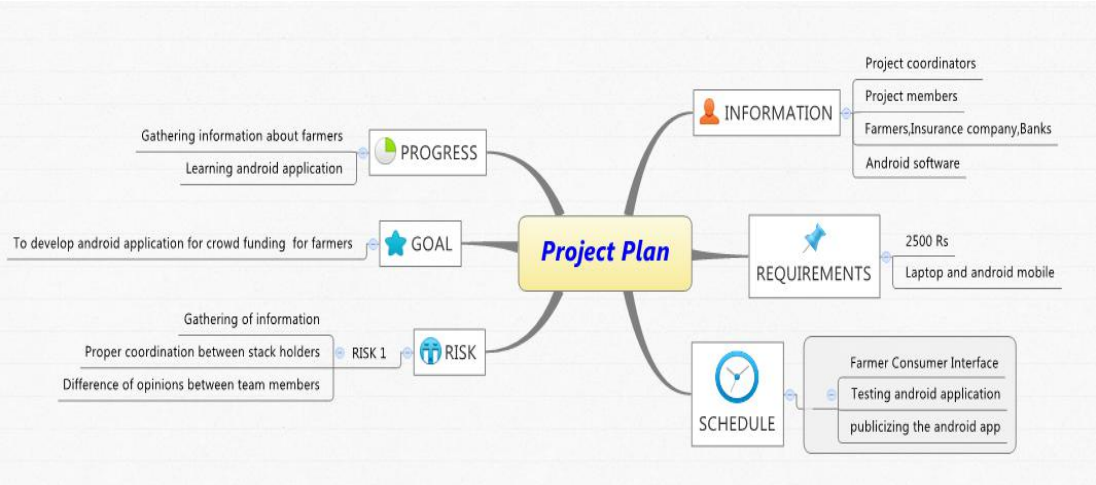


Fig 5.1 Project Plan

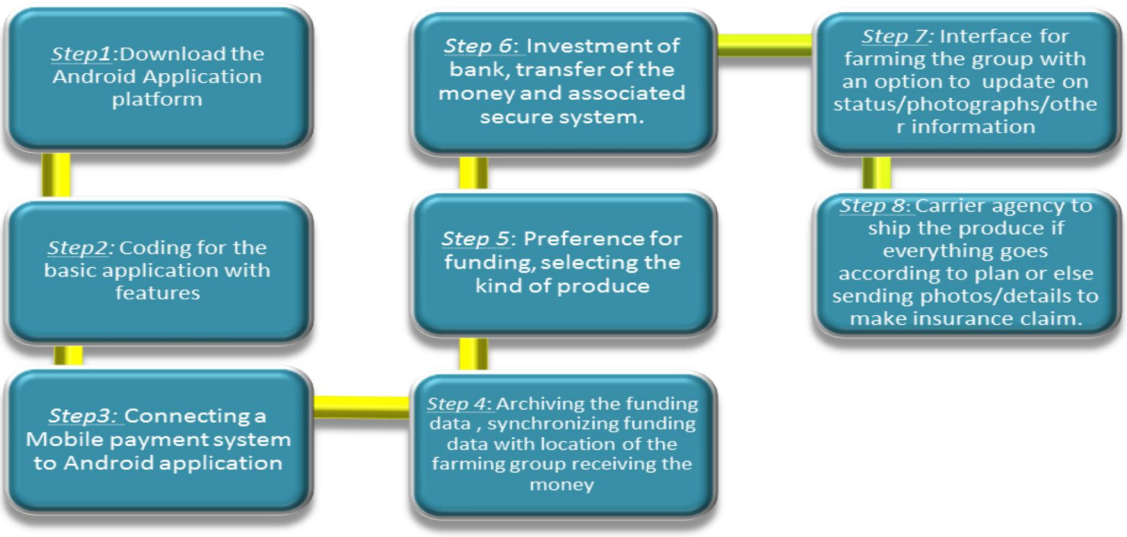


Fig 5.2 Methodology

• To develop a model for farmers crowd funding to assist the farmers for the development and progress of the country.

• To develop application in 'Android Studio' which includes features such as, Secured Authentication, Various Farming Techniques, Government Crop Policies, Organic farming etc.

• To ensure the money routed through Village Help Group through Bank, which provides interest free financing for farmers, and for the total farming activity, the bank takes a general insurance (offered by GIC).

• To ensure the transaction is secure, the application is developed using Block Chain Technology.

5.2 Data Flow Diagram

Data Flow Diagrams (DFD) are used to graphically represent the flow of data in a program or information system. Referring to the below figure initially all the functional, non-functional and resource requirements are collected and analyzed, followed by downloading and installing Android studio and its API. Java programming language is used for designing the interface for the front-end. The next phase is writing the contract code for integrating blockchain with the application. The last phase is testing the entire application with relevant data and verifying all its features.

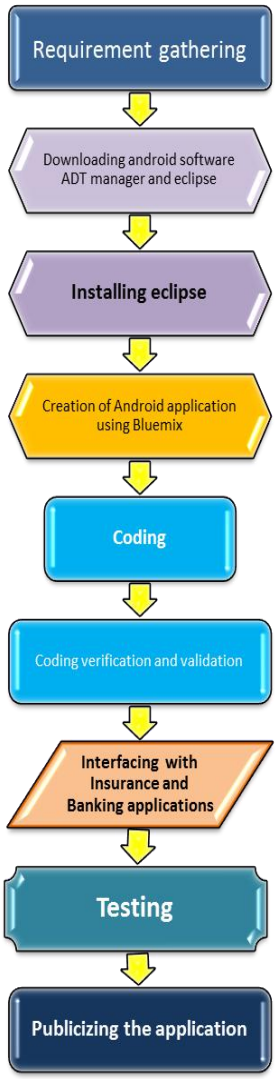


Fig 5.3 Data Flow Diagram

5.3 System Architecture

A system architecture is a conceptual model using which we can define the structure and behavior of that system.

The main ideology of creating farmers crowd funding was to create a centralized and a distributed system where all the other sectors will have to depend on this application. A dependency will be created on Kissan Rakshak application for satisfying the needs of farmers and funders thereby focusing on improving the present lifestyle. The below system architectural diagram illustrates the scenario of the developed application.

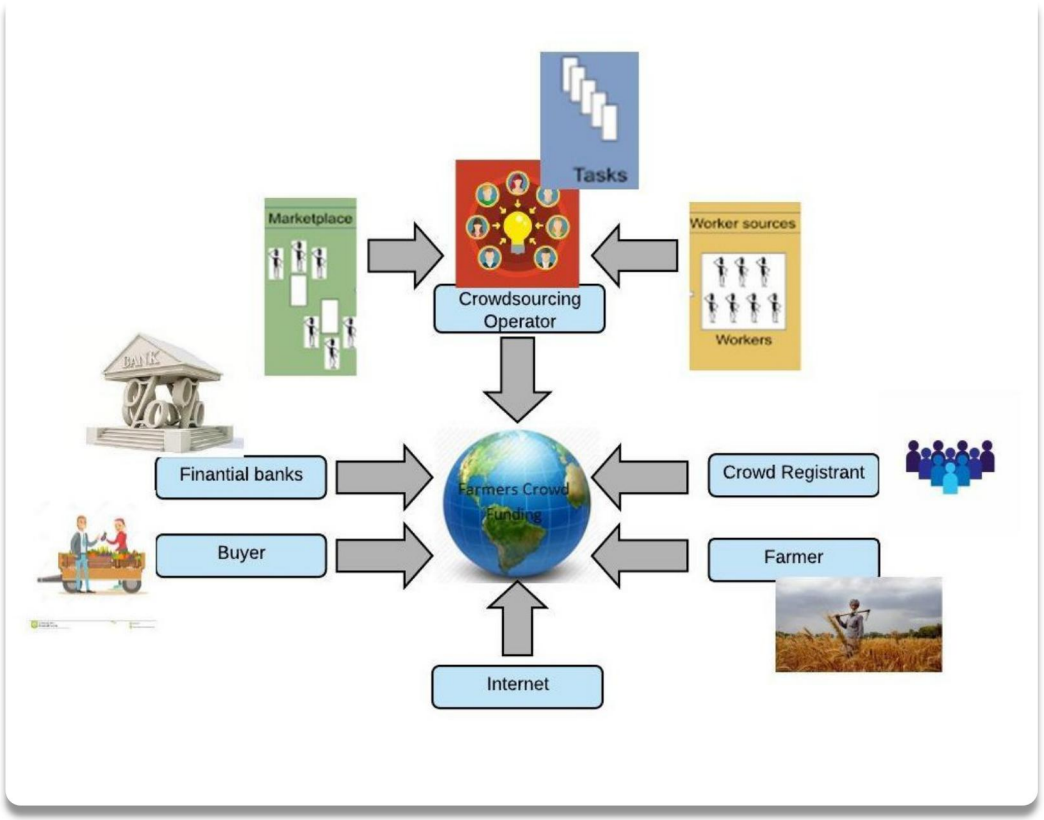


Fig 5.4 System Architecture Diagram

Chapter 6

IMPLEMENTATION

The Government of India is try to solve the financial issues of farmers through different schemes. But if the government or public sector banks try to waive off the loans then the Banking System would collapse. So the idea of crowd funding can help in solving this issue. The society being a part of the venture and rewarding the hardworking farmer, we are addressing one of the biggest social stigma's (being a farmer). If farmer society succeeds in farming, they ship the product as promised. If the farmer fails due to nature calamities like, lack of rain, cyclones, etc. The General Insurance Corporation (GIC) will reimburse the loan to Bank.

With the involvement of crowd funding interested people of India contribute a sum with an assurance of receiving the product. For example, if 'Coca Cola' can reach each and every corner of India and still sell beverage bottle at Rs. 8 to 10, they produce from nearby area and can reach the customers if a support system is provided. Money routed through Village Help Group through Bank, which provides interest free financing for farmers, and for the total farming activity, the bank takes a general insurance (offered by GIC).

Initially Adobe Xd tool is used to design the User Interface and to understand the user experience which includes the basic features and menus from which the final application should evolve.

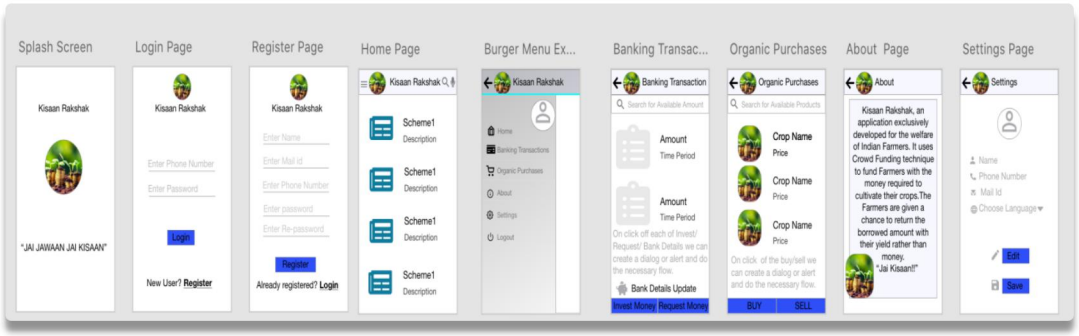
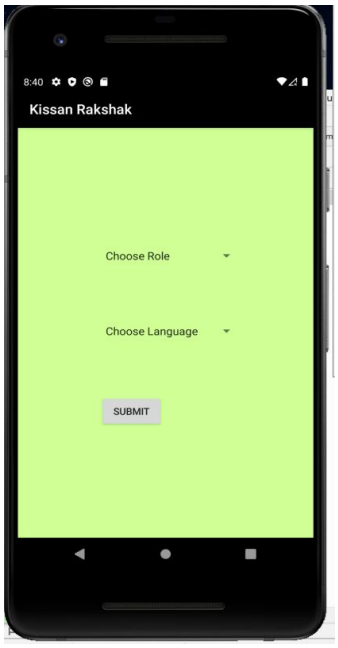
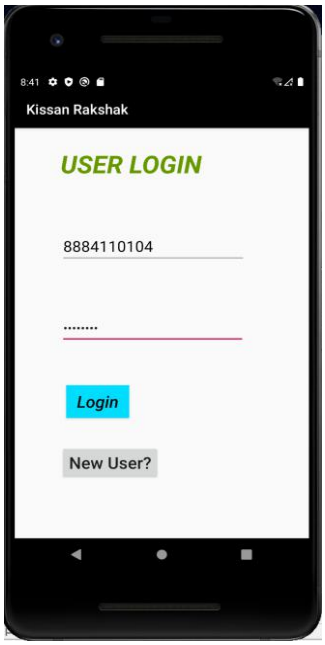


Fig 6.1 The Basic Flow of Application in UI/UX view





The interface in the above screenshot is the main screen of KissanRakshak application where user can choose their role and the language that they want the application to be displayed. The application includes three main roles: Farmer, Funder and Co-Operative Society. The xml is designed using the spinner tag which allows us to display the options available. Based on the role and the language chosen by the user the respective functional unit will get displayed once the submit button is clicked. Here the user login page will be displayed in two different language where the registered user has to login to access the various functionalities in the application.

The implementation of the application can be grouped into 3 major stages where each stage performs an important aspect of the application: Farmer, Funder and the Co-Operative Society.

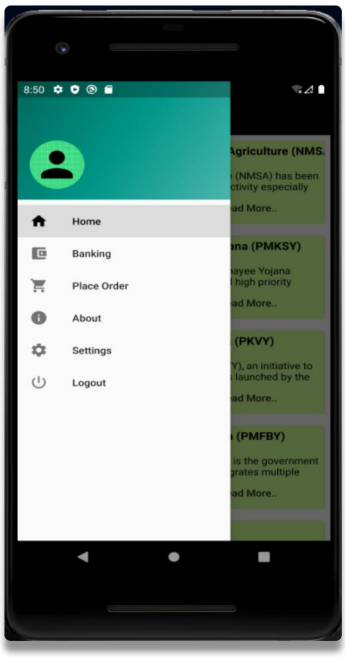
6.1 Creating Farmers Profile

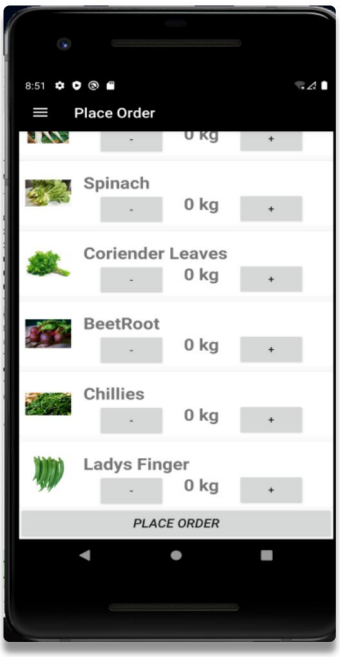
Through our app we are trying our best to extend our help to the backbone of our nation. The application is made user friendly by localizing in Kannada.

The list of various schemes by the government is made available for the benefit of the farmers. The application makes use of recycler view to display the schemes and has a read more button clicking on which the user can get to know more about the schemes in detail. It is important for the farmers to educate themselves for a good outcome and knowing about various crops is very important, thus Kissan Rakshak provides some useful information about the crops. The application also supports farmer to request money when they are in-need and also can add their bank account details and check for the balance. By this the application reduce the interest rates that they had to pay if they take loan from various banks

6.2 Creating Funder Profile

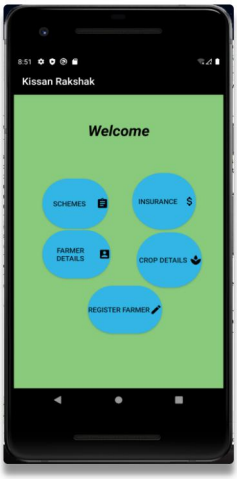
Funders are common people who want to fund some money and help farmers. They play a major role by extending their help and gratitude towards the backbone of our nation. The functional part for funders in Kissan Rakshak application, using which they can easily shed the helping hand for development of the farmers with the benefit of getting a some amount of good crop in return for the money invested.



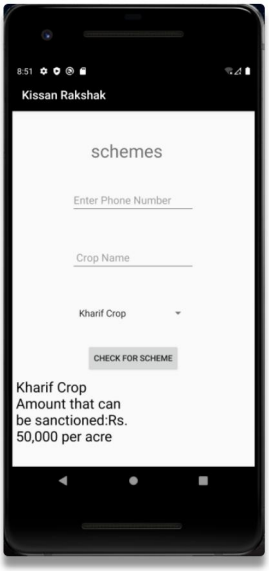


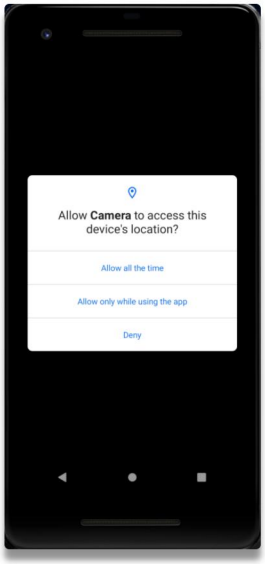
First the user has to register and then easily access the functionalities and this will allow the user to edit their profile. For example user can change the user name, mail id, phone number etc. Funder can see the schemes that government is setting for farmers so that they can spread their knowledge and educate the farmers. User can add their bank account and can raise the fund by making some contribution. Once the bank account is linked the user will be able to easily transact money. Funders can also order vegetables through the application which are cultivated by the farmers registered with Co-Operative Society.The various other features include Editing the profile, Banking (which will help them to easily transact their money) etc. The about interface gives a brief idea about the motto of Kissan Rakshak application.

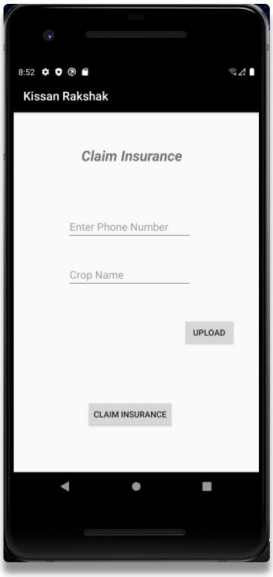
6.3 Creating Co-Operative Society Profile



They are the bodies which will keep records of the number of farmers that are registering and other details like claiming insurance etc.







The application helps the cooperative society to view the number of farmers registered and can also register the farmers. Co-Operative society can access the various schemes that are available when the farmer wants to claim for money.Based on the type of crop that they are cultivating different amount of money will be sanctioned. The Co-Operative society can also help the farmers to claim for insurance if their crops fails due to any natural calamity by helping them upload the picture of their field as a proof.

6.4 Integrating Blockchain for Transactions

With the increase in development of technologies the threat on security increases. So to have a secure and trust worthy system of transaction the app is integrated with blockchain a distributed system of network. Where the bank details are securely sent through blockchain. The blockchain is built using Ethereum network with Ethers. The system uses a test environment called Ganache-cli which provides 10 accounts with 100 ether each. Web3js is used to connect the app to the Ethereum network. The environment used is web provider with gas limit used is 3000000 to have the transactions. Here gas limit term means the maximum amount of gas that is spent on a particular transaction. A higher gas limit means the network must do more work to execute a transaction using ether or a smart contract. Whenever a transaction happens then each is associated with the hash value and securely transfer from app through Ethereum network. Each transaction is associated with a smart contract which is written in solidity language in Remix-ide which is then compiled and deployed to get the json file format. This json code is copied in .js code of app along with the address of deployed contract to have connection between app and smart contract.

Code Snippet:

```
pragma solidity ^0.4.18;

contract bankfund {

 string contact; string account;   string branch; string ifsc;

 function setInstructor(string _contact, string _account, string _bname, string _branch, string _ifsc) public {

        contact = _contact;

        account = _account; bname = _bname;

        branch = _branch; ifsc = _ifsc;

        }  }
```

Chapter 7

TESTING

7.1 Testing Methodologies

The program comprises of several algorithms which are tested individually for the accuracy. we check for the correctness of the program as a whole and how it performs.

7.1.1 Functional Testing

Functional software testing ensures that the application is well and functioning correctly. This type of testing focuses on the main purpose and flow of the app, ensuring that all its features are responsive and meet specifications. The Kissan Rakshak application is tested for its functional application and the comments are as below:

- The application installs and launches correctly
- The users can sign-up and login
- Text boxes and buttons function properly
- Push notifications render correctly

7.1.2 Compatibility Testing

Compatibility testing is a type of non-functional testing that is critical to ensure your mobile app works on various operating systems, devices and applications, network environments, and with particular internal hardware specifications. The findings of KissanRakshak application are as below:

- The app is compatible with various versions of android APK above 4.1
- The app performs well with varying networks and their parameters (bandwidth, operating speed, etc.)
- The app is compatible with different browsers (Google, Firefox, Safari, etc.)

7.1.3 Security Testing

The main focus in development of any android application is 80 percent of users would "uninstall an app due to security". With the implementation of Blockchain, the data and the transactions are made much more secure and faster. All the personal data, user authentications and payment gateways are ensured to be working perfectly, thus guaranteeing security.

7.1.4 Manual Testing

Mobile app testing is a complex process involving various mechanisms and different skill sets. To ensure the application is working fine under long runs, manual testing is performed to ensure that the final product really works as intended. Additionally, the application has undergone:

- Physical interface tests.
- Complex tests
- Exploratory testing (for finding bugs)

7.1.5 Interface Testing

This testing is done after all the modules of the app are completely developed, tested individually and all the bugs are fixed verified.

The KissanRakshak application interface is tested and the following observations are made:

- The funder is able to make transactions successfully.
- The bank details interface can be viewed by the users.
- The linked webpages are redirected easily.

Chapter 8

BENEFITS AND OUTCOMES

The expected result of our project is to make a tremendous impact on farmer's life, which are as follows -

- The developed tool will solve India's biggest problem of **'Farmers' Suicide'.**
- The solution will be the first of its kind where farming, supported by Society and farmer is supposed to pay only in the form of products.
- The project will help the Society and well-being of Farmers.
- Farmers are provided with user-friendly and Language choice feature will benefit those who are less educated and makes it more comfortable.
- The Farmers have opportunity to adopt latest techniques for cultivation, loan facilities by investors / lenders without interest, crop insurance schemes, adopting organic farming for higher productivity and quality, ensuring good return on investment, which will protect them from suicide.
- To involve young and upcoming professionals in the field of agriculture to motivate them to involve themselves in the farming.

Chapter 9

RESULTS AND DISCUSSIONS

The following screenshots depicts the working of Kissan Rakshak application with various features and interfaces.

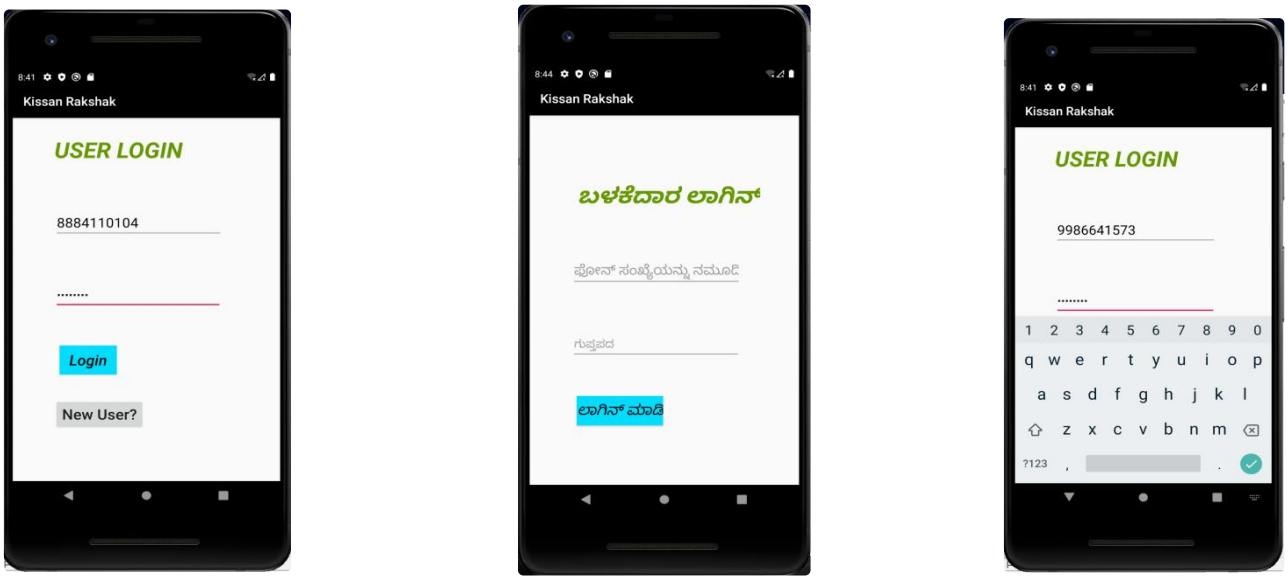


Fig 9.1 Kisaan Rakshak Application Login Screen is made available in the language selected before. An existing user has to fill the details and login to access the functionalities.

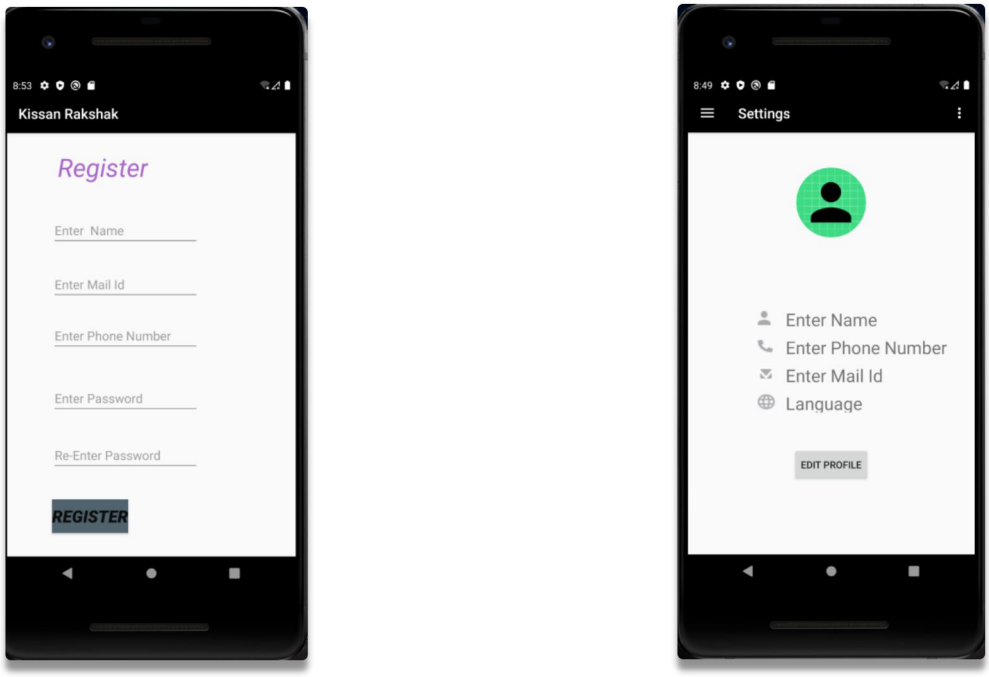


Fig 9.2 Kissan Rakshak Application Register and Edit profile for creating or editing the profile of users.

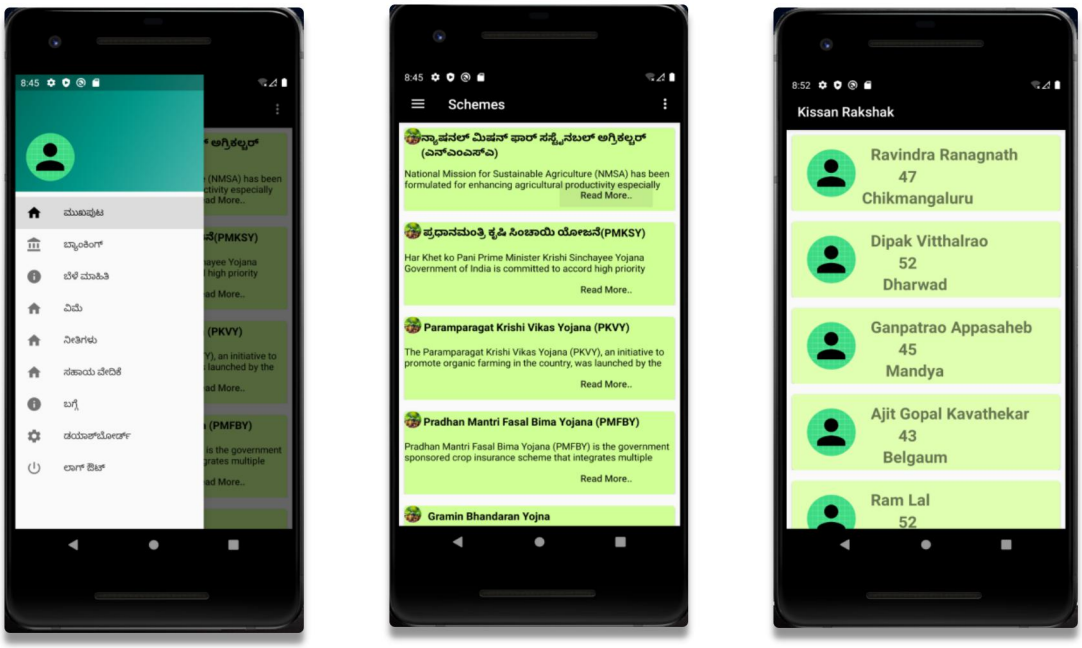


Fig 9.3 Kissan Rakshak Farmer Screen interface which consists of functionalities including various governmental schemes, crop details, emergency contact details and other information.

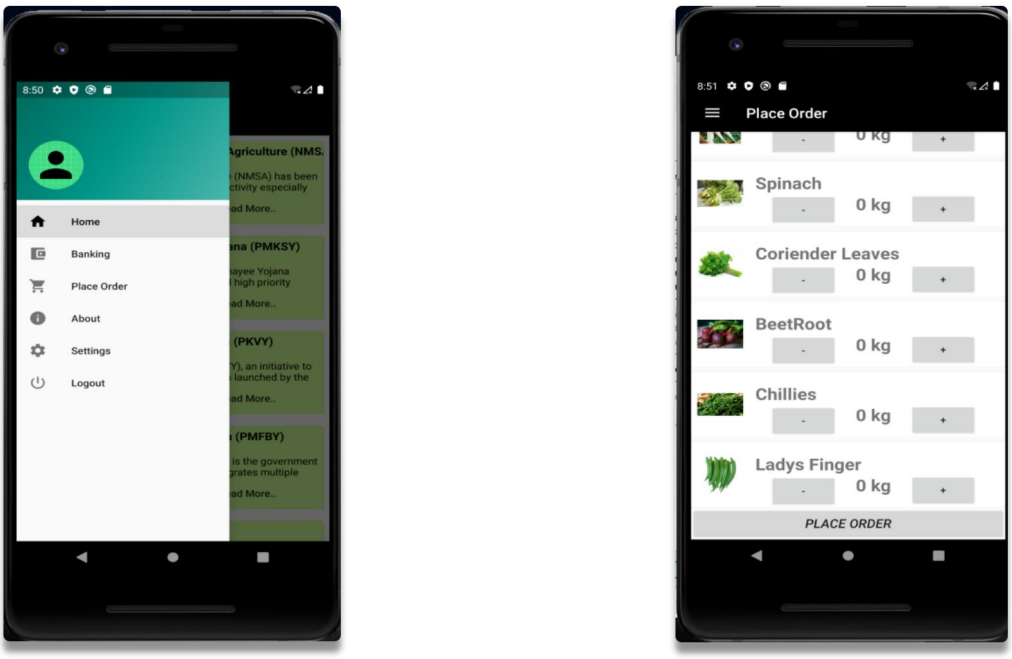


Fig 9.4 Kissan Rakshak Funders interface which includes features like home screen and place order where users can purchase the crops available and make payment for the purchase through the application.

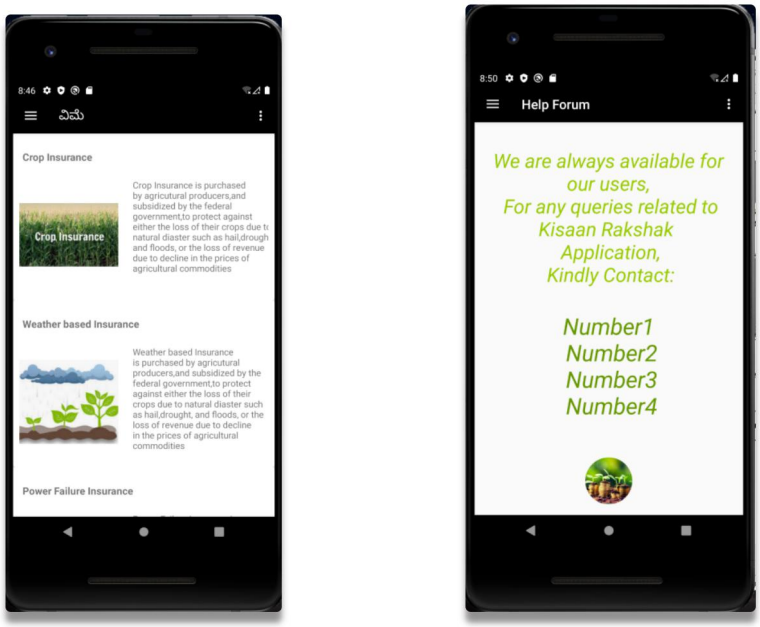


Fig 9.5 Policies, Insurance and Help Forum features included in KissanRakshak application helping farmers to grow their crops productively.

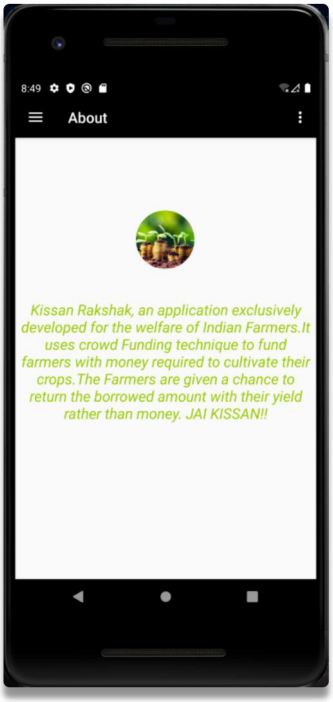


Fig 9.6 About Kissan Rakshak application interface which describes the main ideology and reason behind its creation.

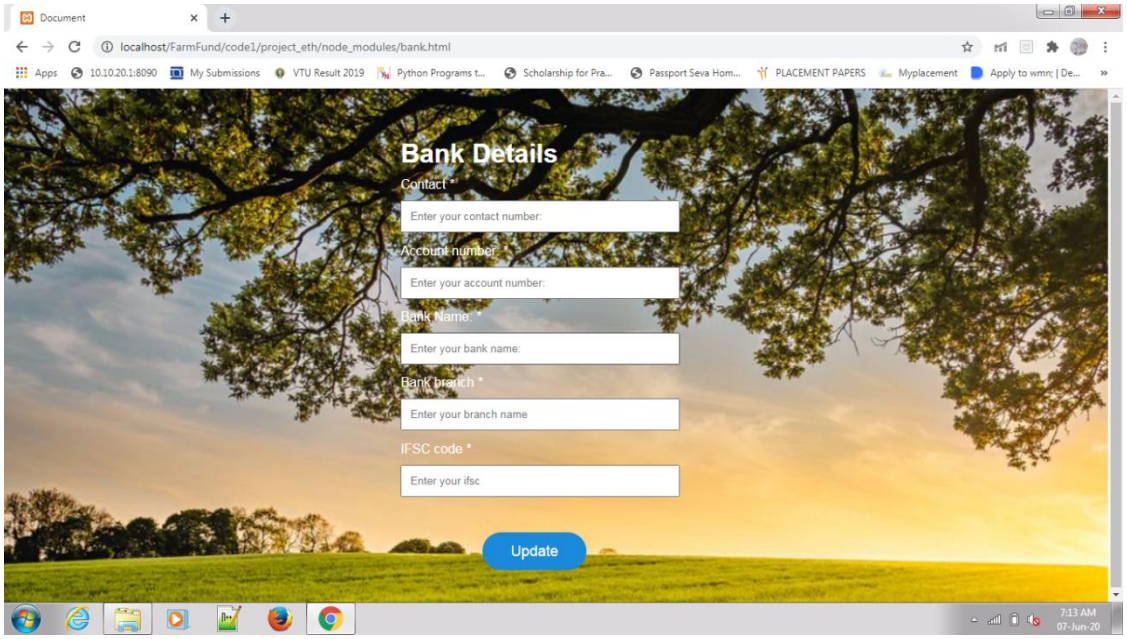


Fig 9.7 Blockchain integrated with Banking Interface where users have to fill in mandatory details for successful transactions.

Chapter 10

CONCLUSION AND FUTURE SCOPE

Enabling our developed tool will solve India's biggest problem i.e. 'Farmers' Suicide'. This solution will be first of its kind where in farming is supported by Society and Farmers are supposed to pay only in the form of product. The project will enable societal good and well being of Farmers.

The need to implement the application is very high as it is capable of saving the lives of farmers as well as to improve their lifestyle, improvising the application with latest updates will be ensured. The application extended with various features including from creating awareness of Resource Utilization to identifying the right crop for particular soil type and its conditions will be given importance. Voice Search will be incorporated with the application. The application can predict the possibility of natural calamities with latest trends.

Crowd Funding for Farmers using Kissan Rakshak will serve as a platform that can make a huge difference in the life of farmers. Hoping this application can be the change in the lives of famers towards a better side!

REFERENCES

[1] A.Yaganteeswarudu & Y.VishnuVardhan, Software application to prevent suicides of farmers with asp.net MVC, 2017 7th International Conference on Cloud Computing, Data Science & Engineering.

[2] Muzaffar H. Assadi, Agrarian Crisis and Farmer's Sucide in India : Dimension, Nature and Response of the State in Karnatka, 2016.

[3] Valerie Busse and Michal Gregu Crowdfunding – An Innovative Corporate Finance Method and Its Decision-Making Steps, 15 August 2019 Advances in Intelligent Systems and Computing book series (AISC, volume 1035).

[4] Muneta Yokomatsu, Wei-Bin Yu, and Norio Okada, Micro Credit, Transaction of Farmland Management Right and Diversification of Disaster Risk in Agricultural Villages in China, 2007 IEEE International Conference on Systems, Man and Cybernetics.

[5] Lili Zhang, Application of BlockChain as Data management, August 2019 in book of Big Scientific Data Management, pp.239-254.

[6] Assadi. "Agrarian Crisis and Farmers' Suicide in India: Dimension, Nature and Response of the State in Karnataka." The Indian Journal of Labour Economics Vol. 49 (4), 2006, 18-24.

[7] K. Nagraj "Farmers' Suicides in India Magnitudes, Trends and Spatial Patterns. Madras Institute of Development Studies 2008.

[8] Shrijit Mishra. "Suicide Mortality Rates across States of India. 1975-2001. A statistical note. Economic and Political Weekly. April 22, 2006, 1566-1569.

[9] P. Satish "Institutional Credit Indebtedness and Suicides in Punjab." Economicand Political Weekly. Vol. XLI No. 26, 2006.

[10] NSSO – "Indebtedness of Farmers Households 2003", 59th round, Report No. 498; Government of India 2005.

[11] Analysis of coordination mechanism of supply chain management information system from the perspective of block-chain https://link.springer.com/article/10.1007/s10257-018-0391-1.

[12] http://agricoop.nic.in/programmes-schemes-listing

[13] https://www.india.gov.in/

[14] https://docs.nethereum.com/en/latest/ethereum-and-clients/ganache-cli/

APPENDIX

Place order Code Snippet of **Farmers Profile** :

There are two java classes : 1) Fragment and 2) Adapter class

Fragment class

```
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.crowdfunding.R;

public class HomeFragment extends Fragment {

    private RecyclerView recyclerview;
    private String sname[];


    private String sdesc[];

    private String surl[]={"https://vikaspedia.in/agriculture/policies-and-
schemes/crops-related/krishi-unnati-yojana/national-mission-for-
sustainable-agriculture-1/national-mission-for-sustainable-agriculture",
    "https://pmksy.gov.in/","https://pgsindia-
ncof.gov.in/pkvy/index.aspx","https://pmfby.gov.in/","https://pib.gov.in/ne
wsite/mberel.aspx?relid=96201","https://vikaspedia.in/agriculture/agri-
insurance/livestock-insurance",
            "http://dahd.nic.in/related-links/centrally-sponsored-scheme-
fisheries-training-and-extension","http://dahd.nic.in/related-
links/centrally-sponsored-national-scheme-welfare-fishermen",
            "https://www.manifestias.com/2019/10/15/micro-irrigation-fund-
mif/"};



    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState)
{

        View root = inflater.inflate(R.layout.fragment_scheme, container,
false);

        recyclerview =  (RecyclerView) root.findViewById(R.id.scheme_view);

        RecyclerView.LayoutManager layoutmanager=new
LinearLayoutManager(getActivity());
        recyclerview.setLayoutManager(layoutmanager);
```

```
        sname=getActivity().getResources().getStringArray(R.array.schemes);

sdesc=getActivity().getResources().getStringArray(R.array.scheme_descriptio
n);
        MyAdapter myadapter = new MyAdapter(getActivity(),sname,sdesc,surl);
        recyclerview.setAdapter(myadapter);


        return root;
    }
}
```

Adapter class

```
import android.content.Context;
import android.content.Intent;
import android.net.Uri;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.core.content.ContextCompat;
import androidx.recyclerview.widget.RecyclerView;

import com.example.crowdfunding.R;

public class MyAdapter extends RecyclerView.Adapter<MyAdapter.MyViewHolder>
{

    Context c;
    String data1[];
    String data2[];
    String url[];
    String colors[];
    RecyclerView recyclerView;
   // int images[];

    public MyAdapter(Context c,String sname[], String sdesc[],String surl[])
{
        this.c=c;
        data1 = sname;
        data2=sdesc;
        url=surl;

       // images = simage;
    }

    @Override
```

```
    public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        View v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.scheme_row,parent,
false);
        return new MyViewHolder(v);
    }
```

Code snippet of **Funders profile :**

There are two java classes : 1) CartFragment and 2) Adapter class

Cart Fragment

```
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.TextureView;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import com.example.crowdfunding.R;


public class CartFragment extends Fragment {

    private RecyclerView recyclerview;
    private Button PlaceOrderBtn;
    private String
st1[]={"Potato","Tomato","Onion","Carrot","Beans","Brinjal","Cucumber","Rad
dish","Spinach","Coriender Leaves","BeetRoot","Chillies","Ladys Finger"};
    int[] image
={R.drawable.potato,R.drawable.tomato,R.drawable.onion,R.drawable.carrot,R.
drawable.beans,R.drawable.brinjal,R.drawable.cucumber,R.drawable.raddish,R.
drawable.spinach,R.drawable.coriender,R.drawable.beetroot,R.drawable.chilli
es,R.drawable.layds_finger};


    public static CartFragment newInstance(){
        return new CartFragment();
    }
    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState)
{

        View view=inflater.inflate(R.layout.fragment_cart,null);

        recyclerview =  (RecyclerView)
view.findViewById(R.id.recycler_view);
```

```
        PlaceOrderBtn = (Button) view.findViewById(R.id.place_order);

        RecyclerView.LayoutManager layoutmanager=new
LinearLayoutManager(getActivity());
        recyclerview.setLayoutManager(layoutmanager);

        My_Adapter myadapter = new My_Adapter(getActivity(),st1,image);
        recyclerview.setAdapter(myadapter);

        PlaceOrderBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Toast.makeText(getActivity().getApplicationContext(),
                        "Your Order has been placed successfully!!!",
Toast.LENGTH_LONG).show();
            }
        });

        return view;
    }

}
```

Adapter

```
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.crowdfunding.R;

class My_Adapter extends RecyclerView.Adapter<My_Adapter.MyViewHolder> {

    Context c;
    String data1[];
    int images[];

    String text;

    public My_Adapter(Context c,String st1[], int image[]) {
        this.c=c;
        data1 = st1;
        images = image;
    }


    @Override
    public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
```

```
        View v =
LayoutInflater.from(parent.getContext()).inflate(R.layout.cart_row,parent,f
alse);
        return new MyViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull final MyViewHolder holder, final
int position) {
        final int[] quantity = {0};
        holder.title.setText(data1[position]);
        holder.image.setImageResource(images[position]);
        holder.t1.setText("0 kg");
        holder.button1.setOnClickListener(new View.OnClickListener() {

            @Override
                public void onClick(View view) {
                    quantity[0] = quantity[0] + 1;
                    text = quantity[0] + "kg";
                    holder.t1.setText(text);
                }
            });
        holder.button2.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View view) {
                  if(quantity[0] ==0){
                      quantity[0] =0;
                      text = quantity[0] + "kg";
                      holder.t1.setText(text);
                  }
                  else {
                     quantity[0] = quantity[0] - 1;
                    text = quantity[0] + "kg";
                    holder.t1.setText(text);
                }

            }
        });
    }

    @Override
    public int getItemCount() {
        return images.length;
    }

    public class MyViewHolder extends RecyclerView.ViewHolder implements
View.OnClickListener {

        TextView title;
        ImageView image;
        Button button1,button2;
        TextView t1;
```

```
        public MyViewHolder(@NonNull View v) {
            super(v);
            title = v.findViewById(R.id.title);
            image = v.findViewById(R.id.image);
            button1=v.findViewById(R.id.increment_button);
            button2=v.findViewById(R.id.decrement_button);
            t1=v.findViewById(R.id.quantity);
        }

        @Override
        public void onClick(View view) {

        }
    }
}
```

Front End

There are two xml files.

Fragment_cart: holds the recycler view

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="0dp">
<androidx.constraintlayout.widget.ConstraintLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    tools:context=".ui.cart.CartFragment">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        tools:ignore="MissingConstraints" />

    <Button
        android:id="@+id/place_order"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Place Order"
        android:textSize="18sp"
        android:textStyle="italic"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.53"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/recycler_view"
        app:layout_constraintVertical_bias="1.0" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>
```

Code Snippet for **Corporative Society :**

Schemes java class

```
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

public class Cooperative_schemes extends AppCompatActivity {
    Button scheme_button,apply_button;
    String crop_type;
    TextView kharif_text,Rabi_text;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.cooperative_schemes);

         scheme_button=findViewById(R.id.scheme_button);
        kharif_text=findViewById(R.id.Kharif_scheme);
        Rabi_text=findViewById(R.id.Rabi_scheme);
        apply_button=findViewById(R.id.apply_scheme);

        Spinner spinner = findViewById(R.id.crop_type);
        ArrayAdapter<CharSequence> adapter =
ArrayAdapter.createFromResource(this, R.array.CropType,
android.R.layout.simple_spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_it
em);
        spinner.setAdapter(adapter);
        spinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> adapterView, View
view, int i, long l) {
                crop_type=adapterView.getItemAtPosition(i).toString();
            }

            @Override
            public void onNothingSelected(AdapterView<?> adapterView) {

            }
```

```
        });
        scheme_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                if(crop_type.equalsIgnoreCase("Kharif Crop")){
                    kharif_text.setVisibility(View.VISIBLE);
                    Rabi_text.setVisibility(View.INVISIBLE);
                }
                else if(crop_type.equalsIgnoreCase("Rabi crop")){
                    Rabi_text.setVisibility(View.VISIBLE);
                    kharif_text.setVisibility(View.INVISIBLE);
                }
                else if(crop_type.equalsIgnoreCase("Choose crop Type ")){
                    kharif_text.setVisibility(View.INVISIBLE);
                    Rabi_text.setVisibility(View.INVISIBLE);
                }
                else{
                    Toast.makeText(getApplicationContext(),"Choose correct
Option",Toast.LENGTH_SHORT).show();
                }
            }
        });
        apply_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Toast.makeText(getApplication(),"Successfully Submitted!!
We will Get Back To You!! ",Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

Banking Java Class

```
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import com.example.crowdfunding.R;
```

```
import com.google.android.material.bottomnavigation.BottomNavigationView;


public class BankFragment extends Fragment {

    private Button checkBalanecBtn, addBankDetailsBtn, transactMoneyBtn;
    AlertDialog dialog1;
    EditText editText1;


    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState)
{

        View root = inflater.inflate(R.layout.fragment_bank, container,
false);
        checkBalanecBtn = (Button) root.findViewById(R.id.CheckBalance);
        checkBalanecBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Toast.makeText(getActivity().getApplicationContext(),
                        "Sorry!, Currently you have 0 Balance",
Toast.LENGTH_LONG).show();
            }
        });
        addBankDetailsBtn = (Button) root.findViewById(R.id.AddBankAccount);
        addBankDetailsBtn.setOnClickListener(new View.OnClickListener() {
```