A
Project Report
on

"KISAAN HELPER"

By

Trilokyanath Santosh Wagh
Abhishekh Ravindra Chaudhari
Rushikesh Padmakar Narkhede
Kapil Mohan Deore
Sagar Sudhakar Patil
Pratik Arun Chavan
Krutika Mahendra Gosavi

The Shirpur Education Society's

Department of Computer Engineering

R. C. Patel Institute of Technology Shirpur - 425405.

Maharashtra State, India

[2016-17]

A
Project Report
on

"KISAAN HELPER"

Submitted By

Trilokyanath Santosh Wagh
Abhishekh Ravindra Chaudhari
Rushikesh Padmakar Narkhede
Kapil Mohan Deore
Sagar Sudhakar Patil
Pratik Arun Chavan
Krutika Mahendra Gosavi

The Shirpur Education Society's

Department of Computer Engineering

R. C. Patel Institute of Technology Shirpur - 425405.

Maharashtra State, India

[2016-17]

Chapter 1

Introduction

Kisaan Helper platform is java based website use to provide proper guidence to farmer in case of their difficulties regarding farming. It is developed to work towards empowerment of farmers and development of villages, the website design is neat and offers a user-friendly interface. It provides information on current weather and also the forecast for the next five days, market prices of commodities/crops in the nearest town, knowledge on fertilizers, seeds, machinery etc. The option to use the website in different languages makes it more widely accessible. The app also offers helpline numbers to get in touch with Kisan Call Centre Services.The user can access a variety of informative modules including agricultural advisory, weather, market prices, agriculture information library in the form of text, imagery, audio and videos in the selected language at profiling stage. It Provides a subsidiary of Indian Farmers' Fertilizer Cooperative Ltd. Its aim is to help Indian farmers make informed decisions through customized information related to their needs.

1.1 Need for the new system

In an attempt to make crop residue management and other agro machinery available to state farmers at a click of a button,we have proposed a website that would help farmers have ready information about the availability of the agro machinery in nearby areas that can be hired for agricultural purposes. Comprise text message services, helplines and apps that provide information on training or weather forecasts, as well as accessing markets, financing and inputs such as fertilizers. It's difficult to determine what plants need without an accurate picture of what's going on in the ground. A soil test can paint a picture of what's going on and indicate if any nutrients are lacking. A common misconception is that gardeners fertilize plants. But fertilizer amends the soil that feeds plants, according to the soil-testing lab professionals.

1.2 Detailed Problem definition

Seeds:
Seed is a critical and basic input for attaining higher crop yields and sustained growth in agricultural production. Distribution of assured quality seed is as critical as the production of such seeds. Unfortunately, good quality seeds are out of reach of the majority of farmers, especially

small and marginal farmers mainly because of exorbitant prices of better seeds.

Manures, Fertilizers and Biocides:
Indian soils have been used for growing crops over thousands of years without caring much for replenishing. This has led to depletion and exhaustion of soils resulting in their low productivity. The average yields of almost all the crops are among t e lowest in the world. This is a serious problem which can be solved by using more manures and fertilizers.

Irrigation:
Although India is the second largest irrigated country of the world after China, only one-third of the cropped area is under irrigation. Irrigation is the most important agricultural input in a tropical monsoon country like India where rainfall is uncertain, unreliable and erratic India cannot achieve sustained progress in agriculture unless and until more than half of the cropped area is brought under assured irrigation.

1.3 Proposed system

By proposing this projet we conclude that the most of problems that farmer faced due to illitercy will be solved by our platform. The chatbox in our project will help to get communication with our agriculture technical experts. Weather display column will help them to understand weather. Fertilizer section is given to buy and search for fertilizer which is suitable for crop soil testing division is given to give the report regarding fertility of soil, it's NPK values and ferture crop information. By using our plateform there is no need to be worry. We have provided the optimum platform for farmers that will definately help them in future now there is no need to go in market for fertilizes and seeds. No need to visit govt. offices for insurance every thing can be done just by sitting at a place. There will be no any kind of lacking of technical. Their problems with the experts and also among other farmers from their area.

Chapter 2

Requirement Analysis

Regarding our project work the current problem is related to Farmers. If any of the grieviences is their, then solving all those problems of farmer is not an easy job because at that time we should consider the maximum workload and we needs an another platform for keeping that track of record so that the problem will be solved. Requirement analysis, also called as requirement engineering, is the process of determining user expectations for a new or modified product. Requirement analysis is critical to the success of system or software project. The requirement should be documented, actionable, measurable, testable, traceable, related to the identified business or opportunities and defined to a level of detail sufficient for system design. These features, called requirements, must be quantitative, relevant and detailed. Conceptually, requirements analysis includes three types of activity: Eliciting requirements: The task of communicating with customers and users to determine what their requirements are. This is sometimes also called requirements gathering. Analyzing requirements: Determining whether the stated requirements are unclear, incomplete, ambiguous, or contradictory, and then resolving these issues. Recording requirements: Requirements might be documented in various forms, such as natural-language documents, use cases, user stories, or process specifications.

2.1 Data Requirement

Data requirement define the specific data item/data structure that must be included as a part of software product. For example, in this system the data requirement is the farmer reviews about a services that would be provided for as input to the system. .

2.2 Functional Requirements

In software engineering, a functional requirements defines a function of software as a set of input, the behavior and output functional requirements may be calculations of technical details, data manipulation, processing and other specific functionality that defines would be what system is supposed to accomplish. Behavior requirements describing all the cases where the system used the functional requirements are capture in use cases.

- Generally functional requirements are express in the form of "System must do", while non-functional requirements are "System shall be".
- The plan for implementing functional requirements is detailed in the system design.
- The plan for implementing functional requirements is detailed in the system architecture.

2.3 System Specifications

2.3.1 Hardware Requirement

1. RAM : 2 GB
2. Storage : 16 GB
3. OS :Windows XP or above

2.3.2 Software Requirement

1. Operating system : Windows
2. Programming Package : Java Ecllipse Neon J2EE Edition and SQL Server
3. Coding Language :Java, XML, HTML, JSP, Ajax, Json and CSS
4. Database : MySQLi

Chapter 3

Planning and Scheduling

Goal is to establish a programmatic strategy for authenticating an user in the technical project. The purpose planning is to ensure that end result is to avoid any intruder in the system and provide a security for user to login in the system.

3.1 Project Planning

Project plan is important to understand software in a system context and to review the software scope that was used to generate planning estimates. Next, communication for analysis must be established so that problem recognition of basic problem elements is perceived by the customer user. For example, an inventory control system is required for a major supplier of auto parts. Analysts and the problem with the current manual system. Inability to obtain the status of a component rapidly, two or three days to update a caedle, multiple records of the same vendor with components and so forth. Once the problem is identified, the analysis determine what information is to be produced by the new system and what data will be provided to the system. For instance, the customer desires daily report that indicates what parts have been taken from inventory and how many similar parts re- main upon evaluating current problem and desired information(input and output).

Project planning must deal with:

- Project Complexity: Has a strong effect but is heavily influenced by past practitioner experience.

- Project Size: As size increases the inter dependencies of element also grows.

- The degree of uncertainty: The degree to which requirements are solidi?ed and the ease of function decomposition. The plan provides the basic for control without that plan there is no basic for determining when variances occur and no basis for any corrective action.

3.2 Project Scheduling(Cost and Effort)

Regarding our project work scheduling in a project refers to road map of all activities to be done with specified order and within time slot allotted to each activity. Project managers tend to defines various tasks, and project milestones and them arrange them keeping various factors in mind. They look for tasks that lie in critical path in the schedule, which are necessary to complete in specific manner (because of task interdependency) and strictly within the time allocated. Arrangement of tasks which lies out of critical path are less likely to impact over all schedule of the project. For scheduling a project, it is necessary to

- Break down the project tasks into smaller, manageable form.
- Find out various tasks and correlate them.
- Estimate time frame required for each task.
- Divide time into work-units.t.
- Assign adequate number of work-units for each task.
- Calculate total time required for the project from start to finish.

3.2.1 Effort Estimation

Our development schedule consists of following steps:

- Estimate the size of the product..
- Estimate the effort(man-months)..
- Estimate the schedule(calender-months)..

Critical Path:The chain of task,that determines the duration of the project.

Earliest Time:Earliest Time that a task can begin if all the preceeding tasks are

Latest Time:Latest Time for the task initialized that will not delay the project.

3.2.2 Cost Estimation

Software project management begins with a set of activities that are collectively called project planning. Before the project can begin ,team must estimate the work to be done, the resources that will be required, the time that will elapse from start to finish. Whenever estimates are made, they will look into future and except some degree of uncertainty as a matter of source. Although estimating is as much arts as it is science, this activity needs not be conducted in a hazard manner. There are three parameters involved in computing the total cost of software development project:

Hardware and software cost including maintenance.

Testing cost.

Effort cost.

Chapter 4

Software Requirement Specification

The System Requirements Specification (SRS) is a formal statement of the application functional and operational requirements. It serves as a contract between the developer and the customer for whom the system is being developed. The developers agree to provide the capabilities specified. The client agrees to and the product satisfactory if it provides the capabilities specified in the SRS. A brief description of SRS functions and characteristics includes the following:
The SRS provides the following functions:

- Designing and developing the system.

- Evaluating the product in all subsequent phases of the life cycle..

- Determining the success of the project..

The SRS has the following characteristics:

- Demonstrates that system provides value to FNS in terms of the business objectives and business processes

- Contains a complete set of requirements for the system..

- Is solution independent. The SRS is a statement of what the application is to do not of how it works. The SRS does not commit the developers to a design. For that reason, any reference to the use of a specific technology is inappropriate in an SRS, unless the technology is listed as a system constraint.

- The SRS provides the following requirements, where a requirement is defined as a condition the system must meet for the customer to and it satisfactory. A requirement has the following characteristics: Provides a benefit to the organization. That benefit is directly traceable to the business objectives and business processes of the FNS.

4.1 Data flow diagram

The DFD takes an input-process-output. View of the system that is, data object flow into the software are transform by processing elements and resultant data object flow out of the software. Data object represented by label, arrows and transformation are represented by circle. Also called bubbles. DFD is presented in hierarchical fashion that is first data flow model represents the system as a whole subsequent DFD refine the context diagram (Level 0 DFD), providing increasing details with each subsequent level.

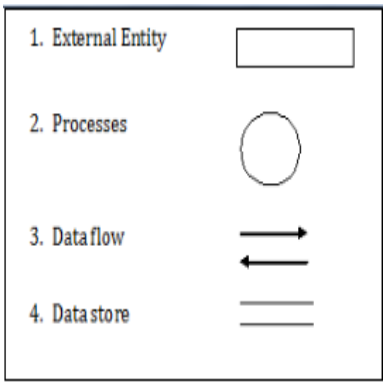

Figure 4.1: Basic Symbols of DFD

A context level DFD for the system. The primary external DFD produce information for used by the system and user information generated by system. The label arrow represent a data object or data hierarchy. The DFD generally contains the following elements and there representation given below:

4.2 UML Diagrams

4.2.1 Use Case Diagram:

A use case diagram is representation of a user's interaction with the system and depicting the specification of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in Conjunction with the textual use case and will often the accompanied by other type of diagram as well.

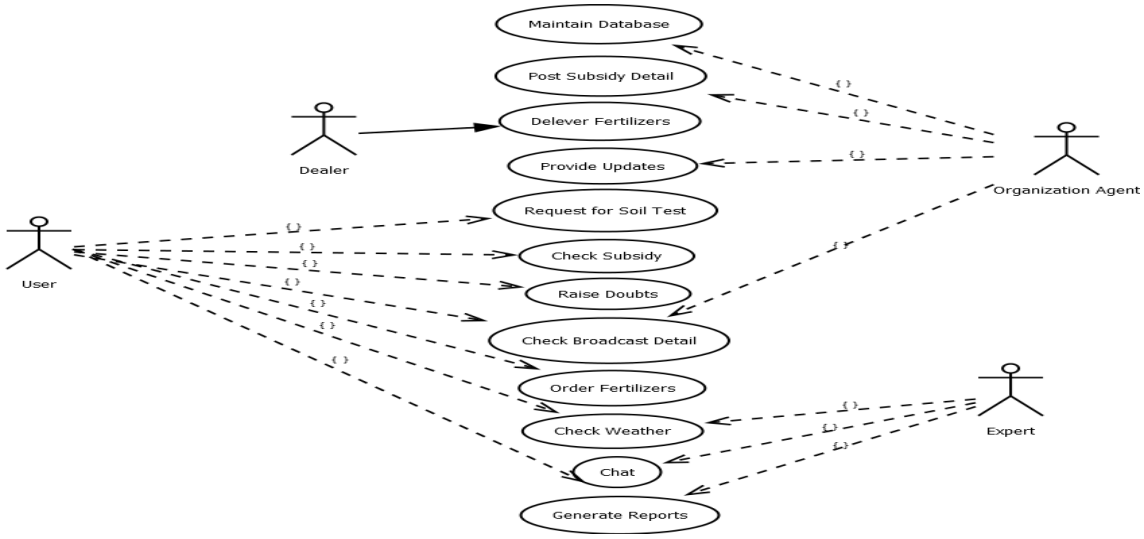

Figure 4.2: Usecase Diagram KISAAN HELPER

4.2.2 Class Diagram:

In software engineering, a class diagram in Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or method) and the relationships among objects. Class diagram can be used for data modeling.

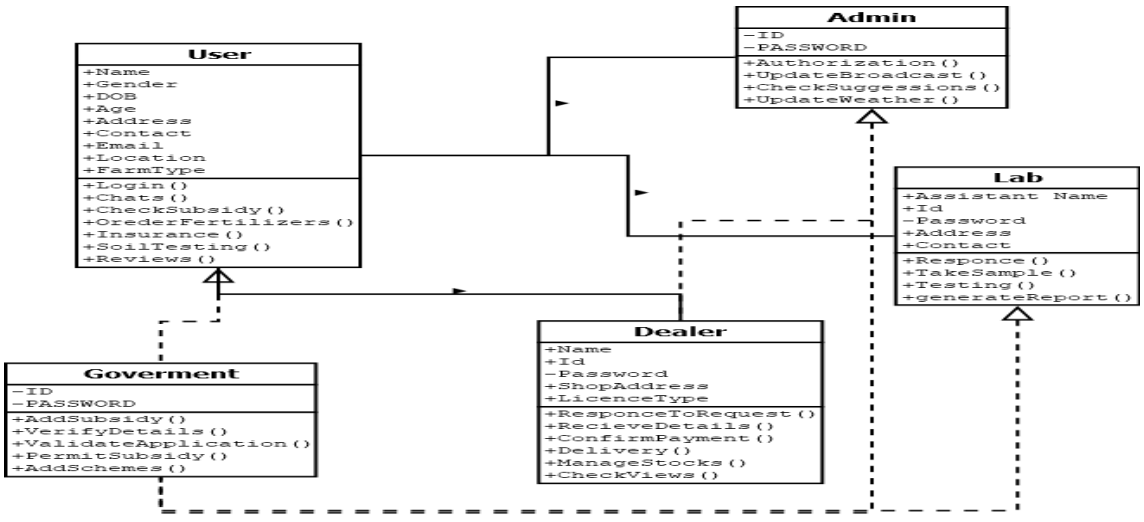

Figure 4.3: Class Diagram KISAAN HELPER

4.2.3 Sequence Diagram:

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interaction arranged in time sequence. A sequence diagram shows, as parallel vertical lines (lifeline), different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the speci?cation of simple runtime scenarios in a graphical manner.

4.2.4 Collaboration Diagram:

A collaboration diagram describes interaction among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence and use case diagrams describing both the static structure and dynamic behavior of a system.

4.2.5 State Diagram:

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

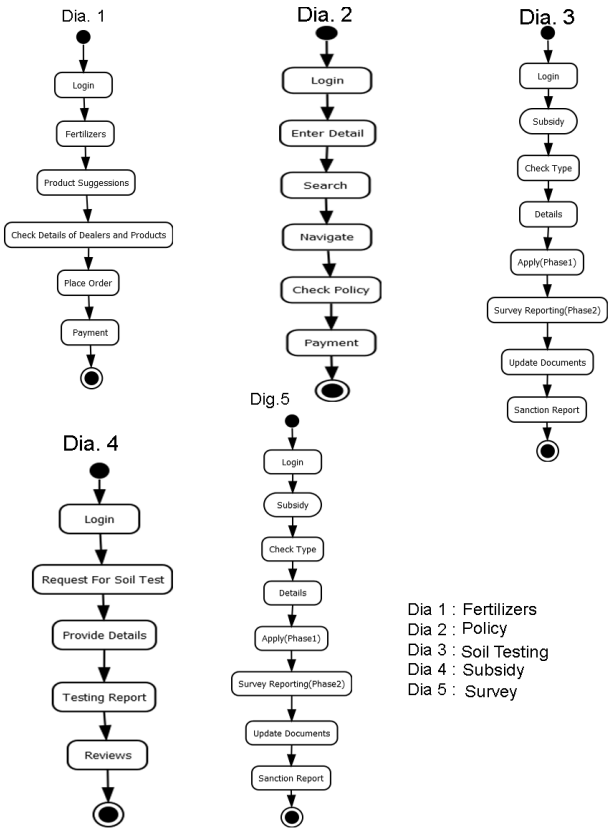

Figure 4.4: State Diagram

4.2.6 Activity Diagram:

Activity diagram are graphical representation of workflow of stepwise activities and actions with the support for choice, interaction and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes. Activity diagram shows the overall flow of control.

4.2.7 Component Diagram:

A component is something required to execute a stereotype function. Examples of stereotypes in components include executables, documents, database tables, files, and library files. Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer-service provider relationship between the two components. An assembly connector is a "connector between two components that defines that one component provides the services that another component requires. An assembly connector is a connector that is defined from a required interface or port to a provided interface or port.

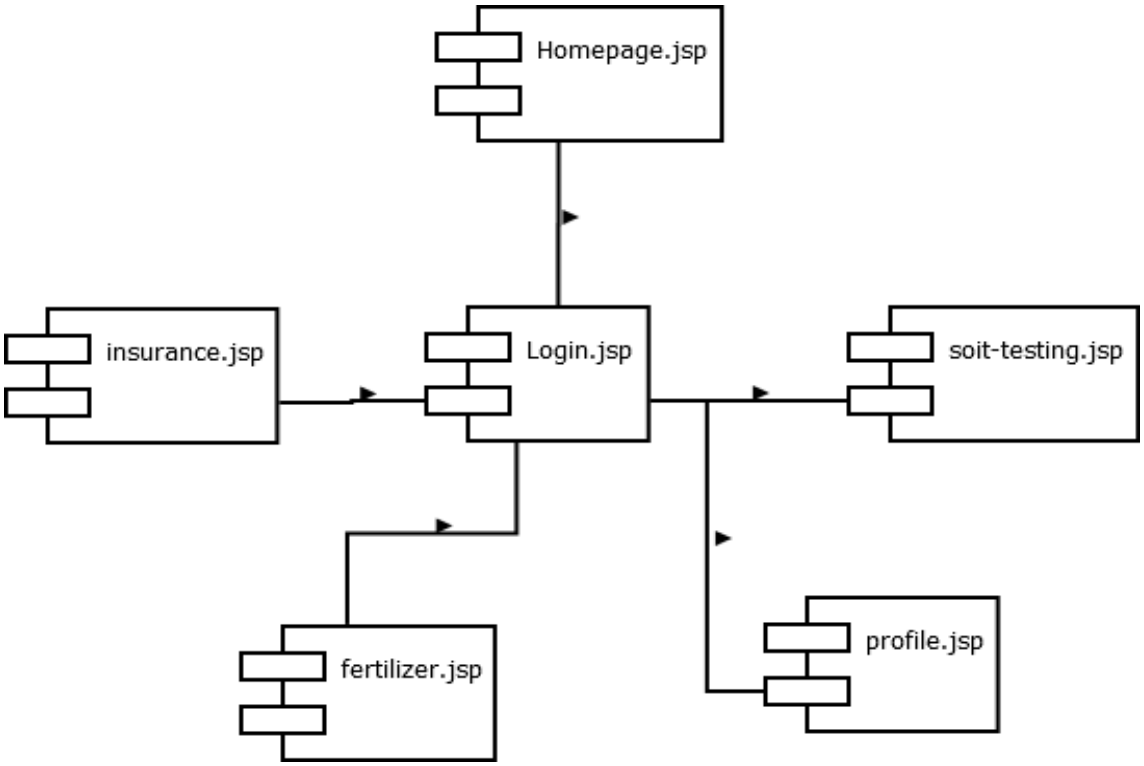

Figure 4.5: Component Diagram

4.2.8 Deployment Diagram:

A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist like a web server, an application server, and a database server, what software components ("artifacts") run on each node such as web application, database, and how the different pieces are connected like JDBC, REST, RMI. The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have subnodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

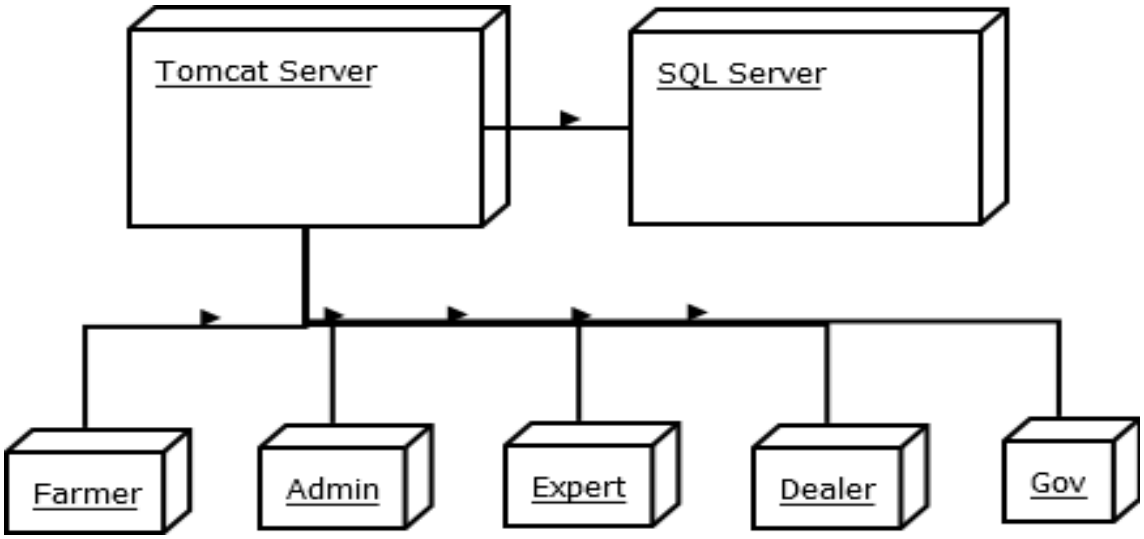

Figure 4.6: Deployment Diagram

Chapter 5

Coding

5.1 Platform

JAVA: Java is a general-purpose computer-programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.
The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Classpath (standard libraries), and IcedTea-Web (browser plugin for applets).

HTML:
In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system. Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed "some of the many areas in which hypertext is used" and put an encyclopedia first.

HTML is a markup language that web browsers use to interpret and compose text, images, and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced

by the web page designer's additional use of CSS.

Markup:
HTML markup consists of several key components, including those called tags (and their attributes), character-based data types, character references and entity references. HTML tags most commonly come in pairs like ¡h1¿ and ¡/h1¿, although some represent empty elements and so are unpaired, for example ¡img¿. The first tag in such a pair is the start tag, and the second is the end tag (they are also called opening tags and closing tags).

Another important component is the HTML document type declaration, which triggers standards mode rendering.

The following is an example of the classic "Hello, World!" program, a common test employed for comparing programming languages, scripting languages and markup languages.

The Document Type Declaration ¡!DOCTYPE html¿ is for HTML5. If a declaration is not included, various browsers will revert to "quirks mode" for rendering.

JSP:
Jackson structured programming (JSP) is a method for structured programming based on correspondences between data stream structure and program structure. JSP structures programs and data in terms of sequences, iterations and selections, and as a consequence it is applied when designing a program's detailed control structure. The method applies to processing of any data structure or data stream that is describable as a hierarchical structure of sequential, optional and iterated elements. This could be a stream of messages that a process reads to invoke and coordinate other modules or objects, or it could be a string of characters in parameters passed to a single operation on an "object" coded in an object-oriented programming language. In other words, it could be either above or below the level where object-oriented methods are applied

Michael A. Jackson originally developed JSP in the 1970s. He documented the system in his 1975 book Principles of Program Design. In a 2001 conference talk, he provided a retrospective analysis of the original driving forces behind the method, and related it to subsequent software engineering developments. Jackson's aim was to make COBOL batch file processing programs easier to modify and maintain, but the method can be used to design programs for any programming language that has structured control constructs, languages such as C, Java and Perl. Despite its age, JSP is still in use and is supported by diagramming tools such as Microsoft's Visio and CASE tools such as Jackson Workbench.

Jackson Structured Programming was seen by many as related to Warnier structured programming, but the latter method focused almost exclusively on the structure of the output stream. JSP and Warnier's method both structure programs and data using only sequences, iterations and selections, so they essentially create programs that are parsers for regular expressions which simultaneously match the program's input and output data streams.

Because JSP focuses on the existing input and output data streams, designing a program using JSP is claimed to be more straightforward than with other structured programming methods, avoiding the leaps of intuition needed to successfully program using methods such as

top-down decomposition.

Another consequence of JSP's focus on data streams is that it creates program designs with a very different structure to the kind created by the stepwise refinement methods of Wirth and Dijkstra. One typical feature of the structure of JSP programs is that they have several input operations distributed throughout the code in contrast to programs designed using step wise refinement, which tend to have only one input operation. Jackson illustrates this difference in Chapter 3 of Principles of Program Design. He presents two versions of a program, one designed using JSP, the other using "traditional" methods.

The method begins by describing a program's inputs in terms of the four fundamental component types. It then goes on to describe the program's outputs in the same way. Each input and output is modelled as a separate Data Structure Diagram (DSD). To make JSP work for compute-intensive applications, such as digital signal processing (DSP) it is also necessary to draw algorithm structure diagrams, which focus on internal data structures rather than input and output ones.

The input and output structures are then unified or merged into a final program structure, known as a Program Structure Diagram (PSD). This step may involve the addition of a small amount of high level control structure to marry up the inputs and outputs. Some programs process all the input before doing any output, whilst others read in one record, write one record and iterate. Such approaches have to be captured in the PSD.

5.2 Runtime Snapshots

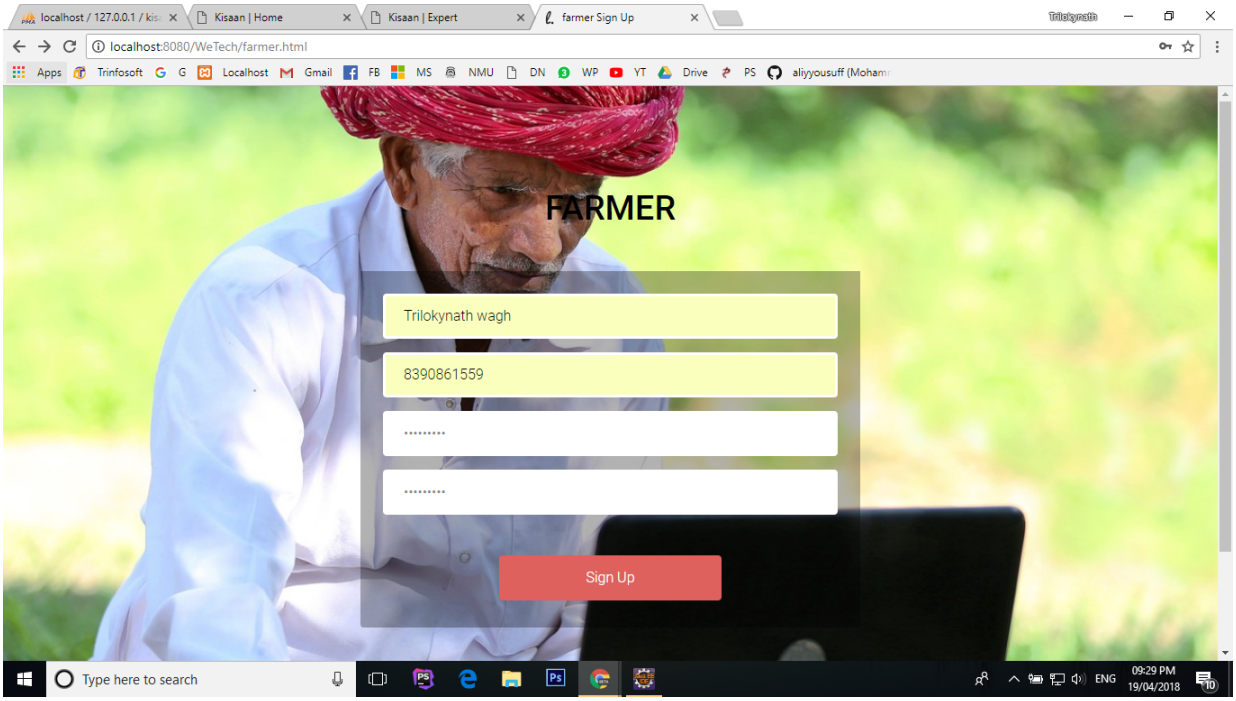

Figure 5.1: Registration Form

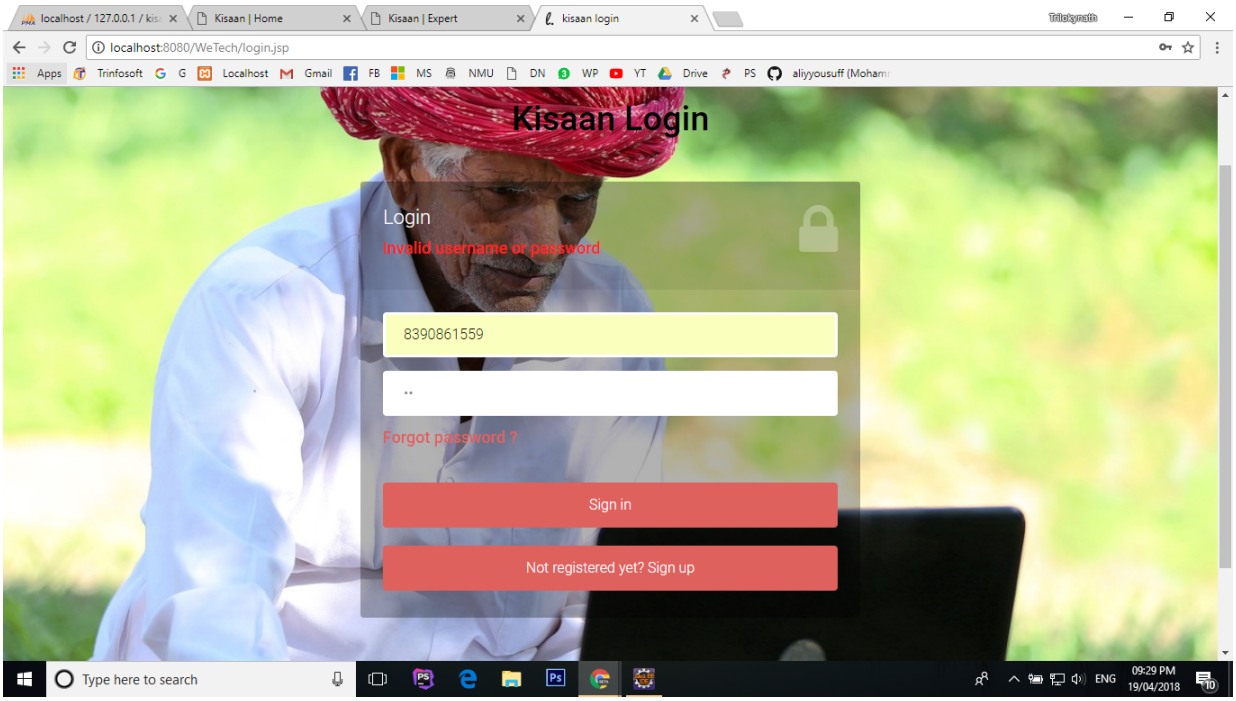

Figure 5.2: Login Form

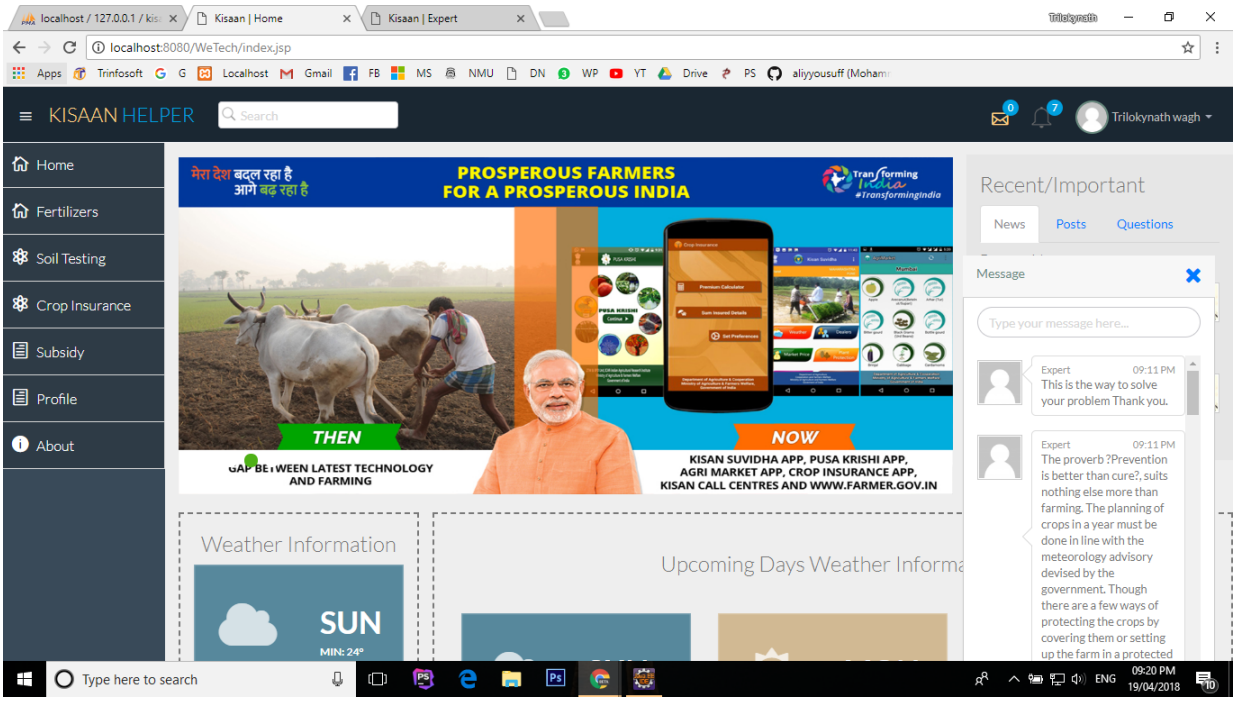

Figure 5.3: Home page

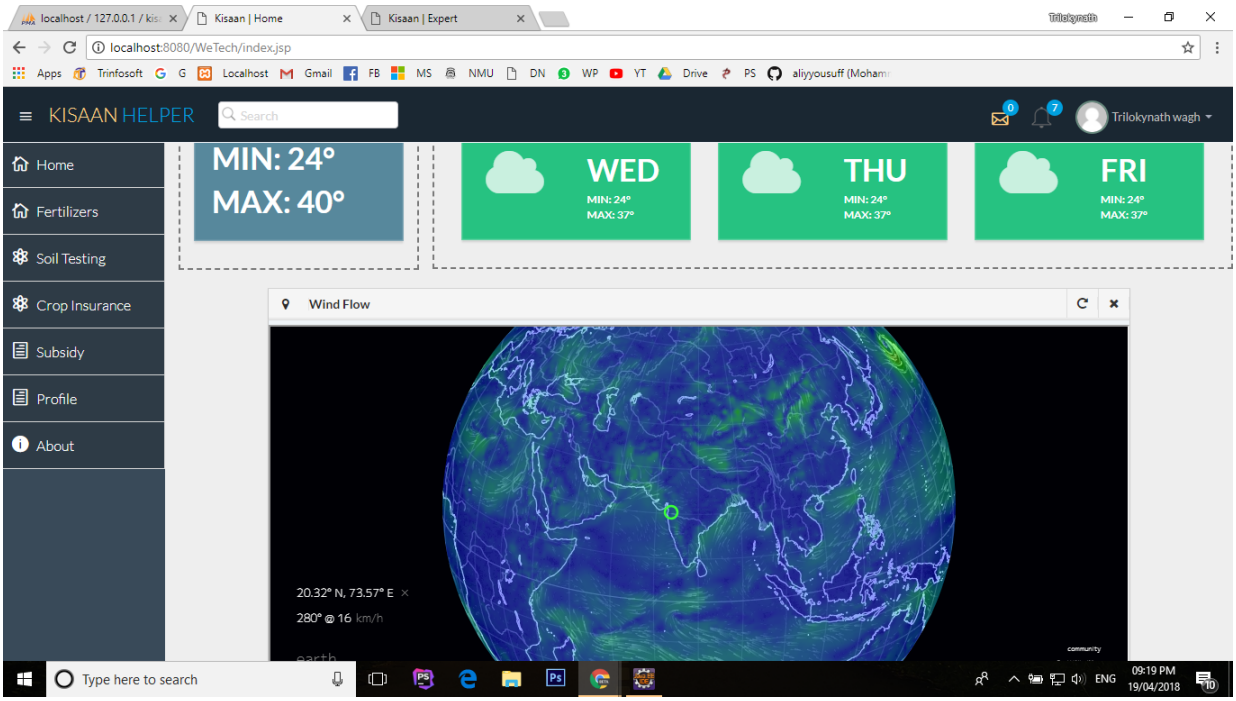

Figure 5.4: Weather Info

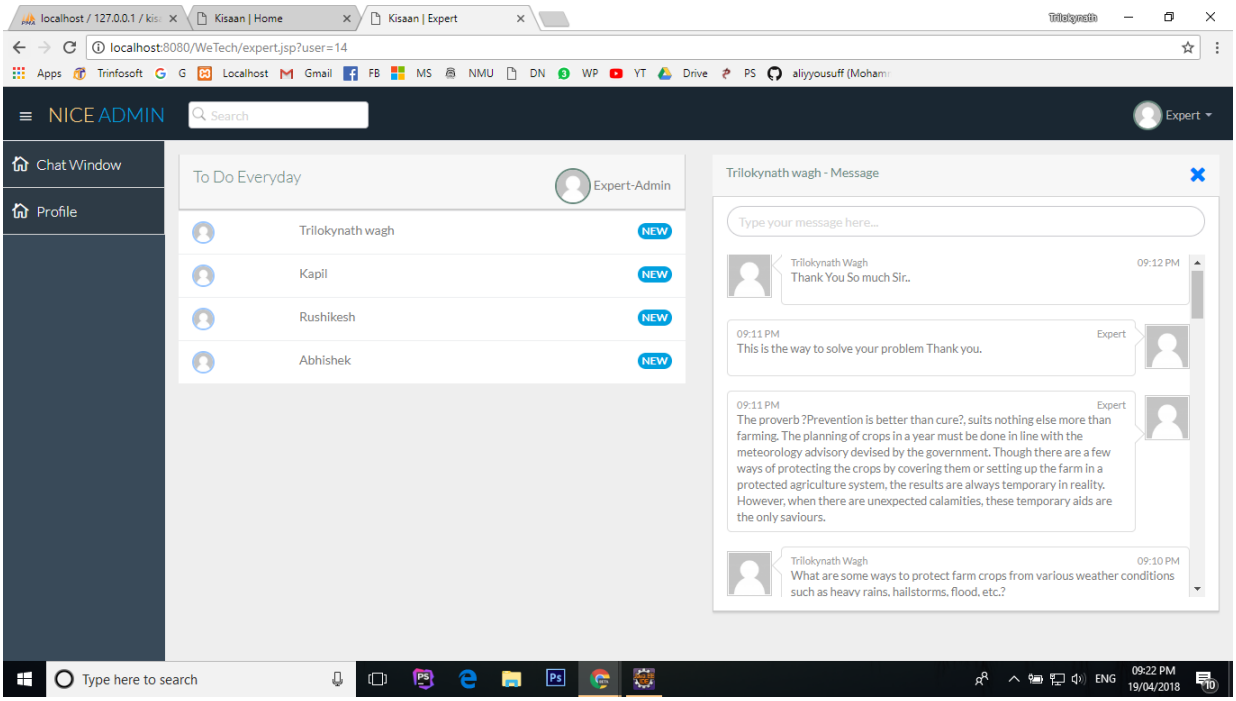

Figure 5.5: Expert Chatbox

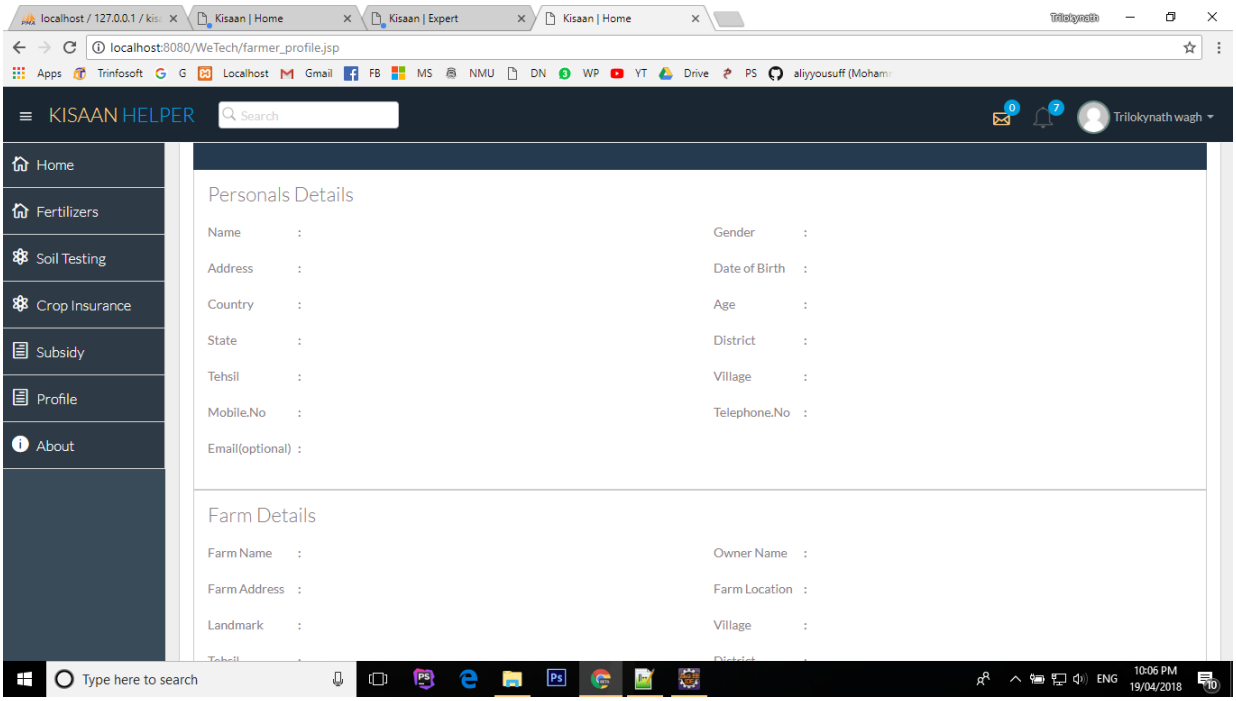

Figure 5.6: Profile Page

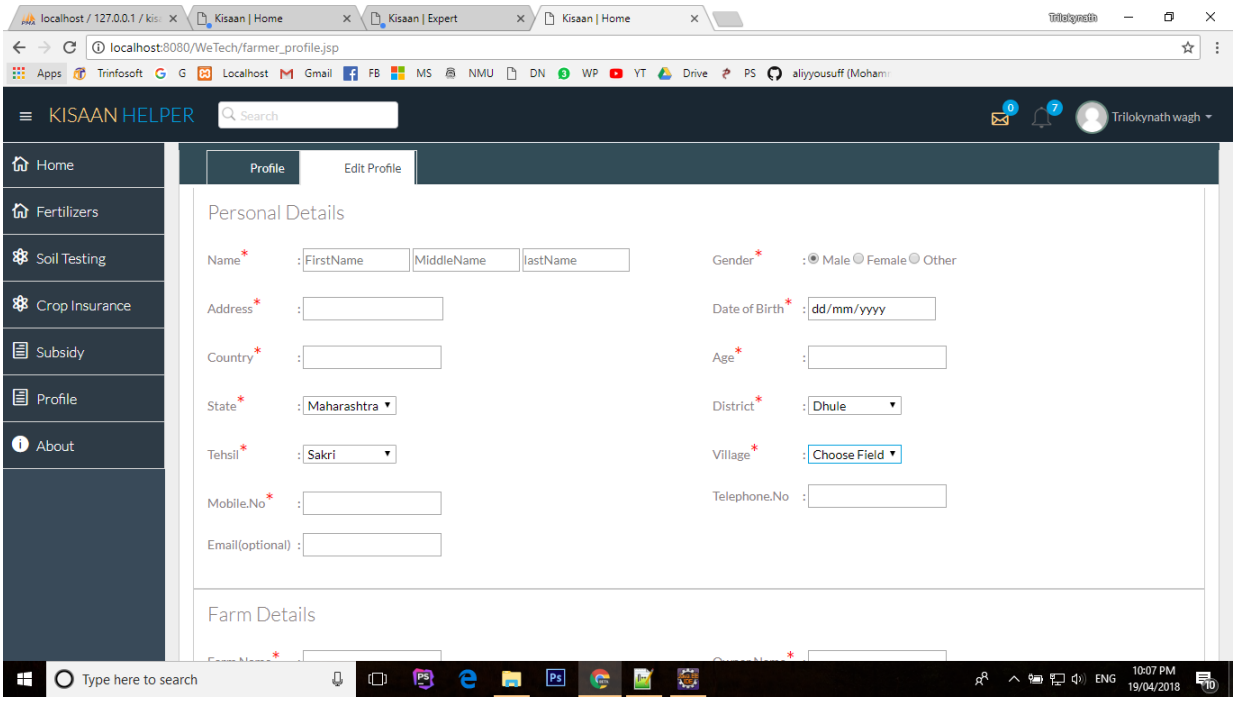

Figure 5.7: Edit Profile page

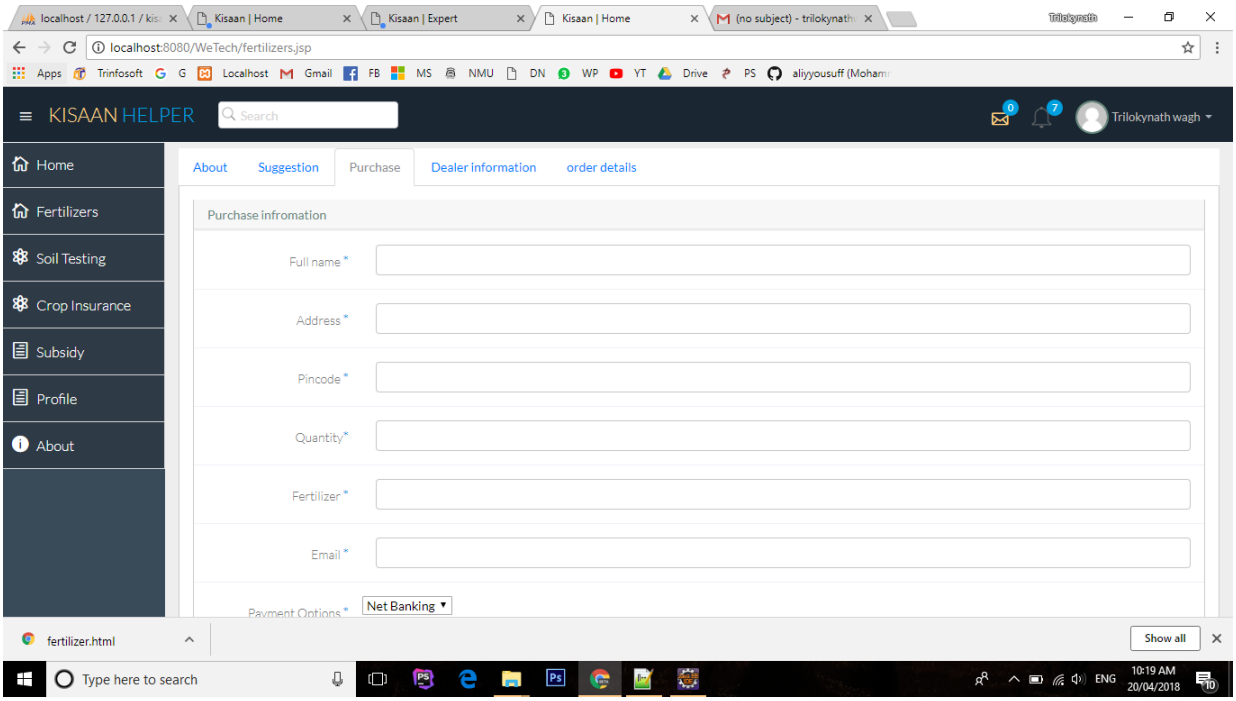

Figure 5.8: Dashboard

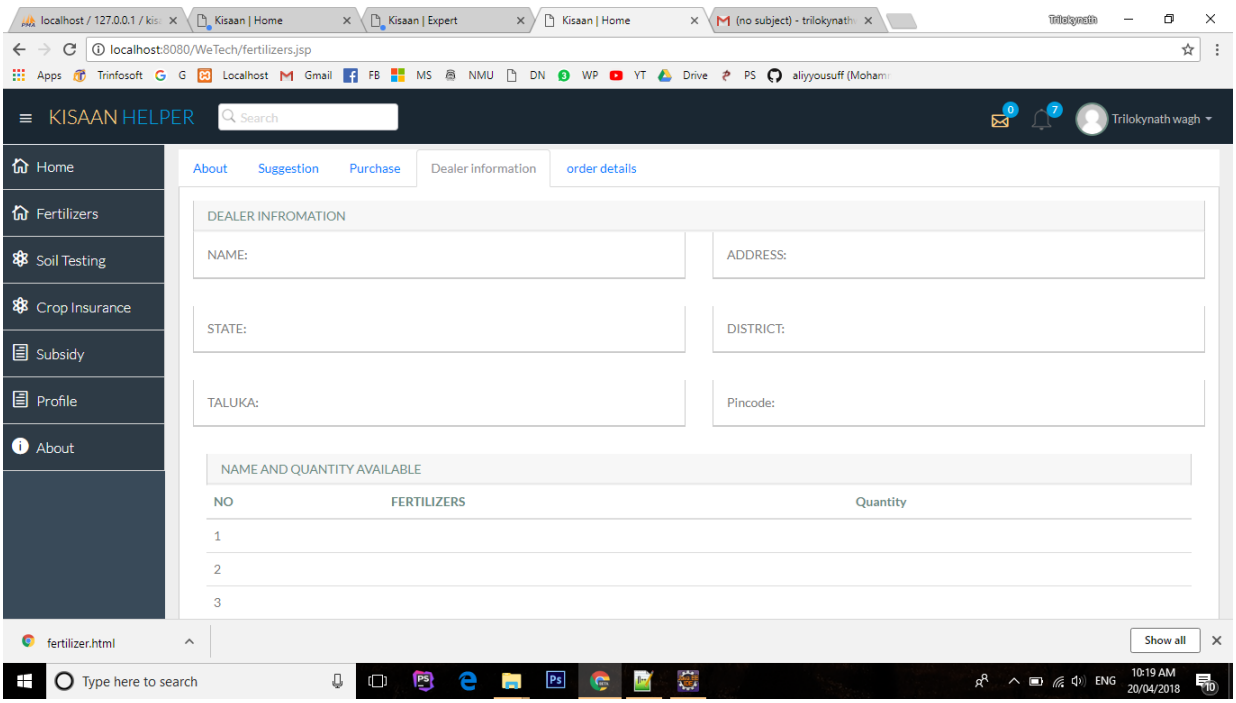

Figure 5.9: Purchase Fertilizers

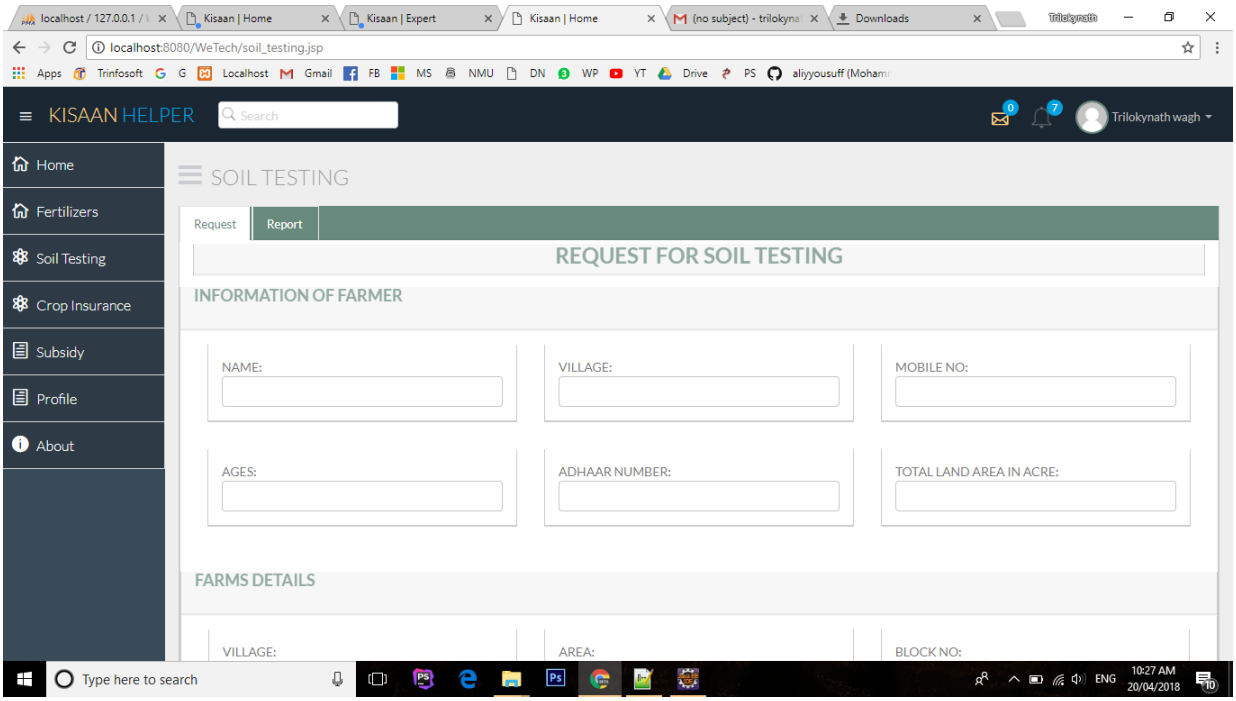

Figure 5.10: Dealer Information

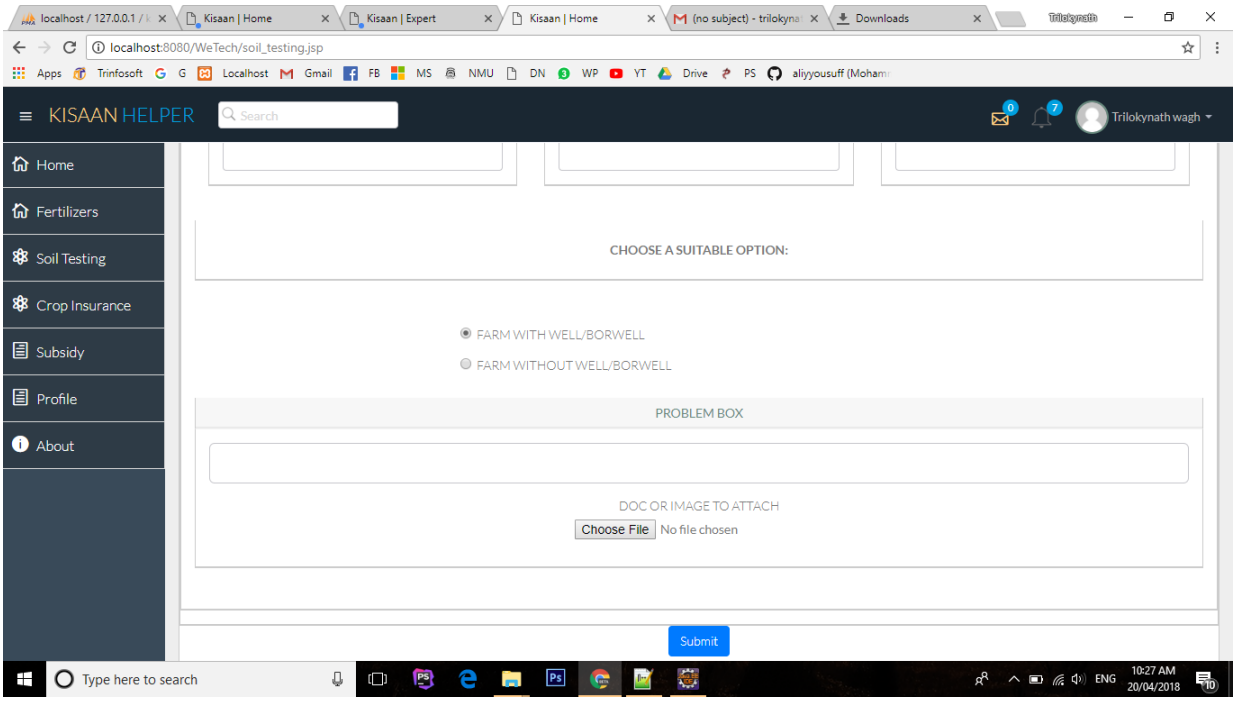

Figure 5.11: Soil Testig Request

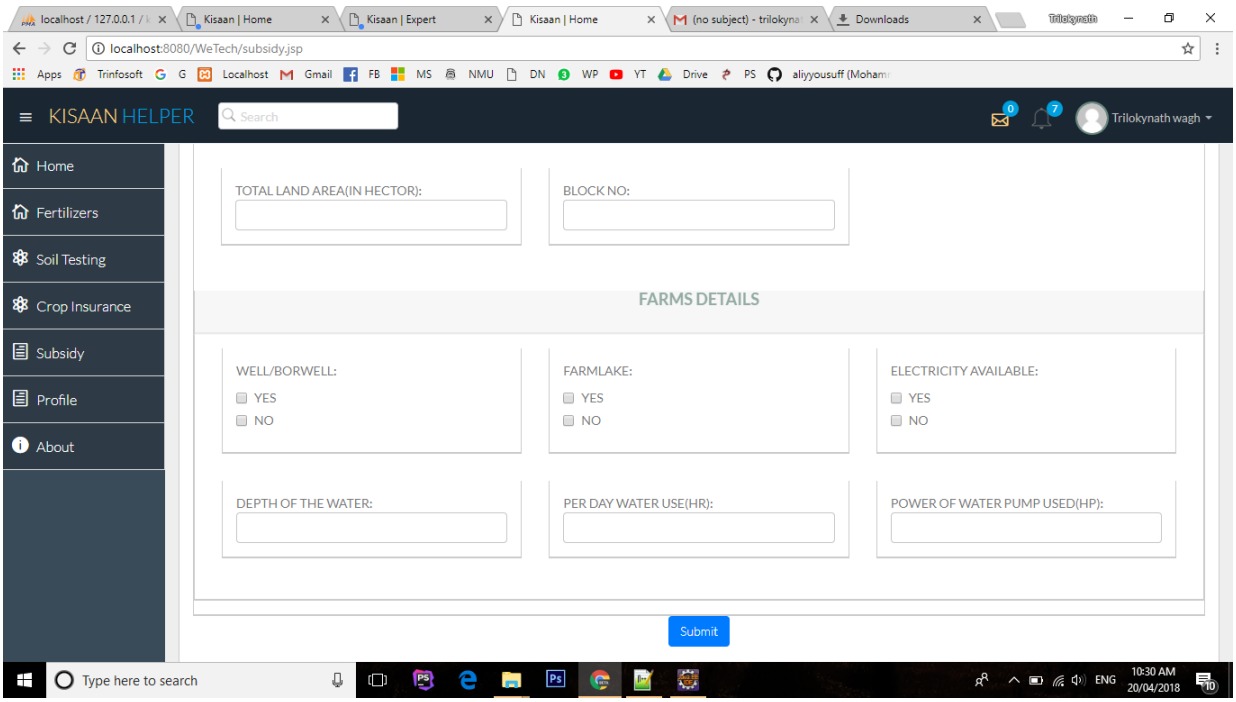

Figure 5.12: Subsidy Form

Chapter 6

Conclusion

6.1 Conclusion

By using our plateform there is no need to be worry. We have provided the optimum platform for farmers that will definately help them in future now there is no need to go in market for fertilizes and seeds. No need to visit govt. offices for insurance every thing can be done just by sitting at a place. There will be no any kind of lacking of technical. Their problems with the experts and also among other farmers from their area.