**ASSISTED DRIVING**

Submitted in partial fulfillment of the requirements

of the degree of

**B. E. Computer Engineering**

By

| **Acvin Gonsalves** | **14** |
| --- | --- |
| **Praveen Fernandes** | **63** |
| **Ashley Mascarenhas** | **67** |
| **Sanford Mascarenhas** | **68** |

Supervisor:

**Ms. Nidhi Gaur**

Assistant Professor

Department of Computer Engineering

St. Francis Institute of Technology

(Engineering College)

University of Mumbai

2018-2019

**CERTIFICATE**

This is to certify that the project entitled **“Assisted Driving using Reinforcement Learning”** is a bonafide work of **“Acvin Gonsalves” (14),“Praveen Fernandes” (63), “Ashley Mascarenhas” (67), “Sanford Mascarenhas” (68)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of B.E. in Computer Engineering.



Ms. Nidhi Gaur

Guide



Dr. Kavita Sonawane

Head of Department

Dr. Sincy George

Principal

**Project Report Approval for B.E.**

This project report entitled ***Assisted Driving*** by **Acvin Gonsalves, Praveen Fernandes, Ashley Mascarenhas, Sanford Mascarenhas**is approved for the degree of ***B.E. in Computer Engineering.***

Examiners

1.---------------------------------------------

2.---------------------------------------------

Date:

Place:

**Declaration**

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

-----------------------------------------

Acvin Gonsalves(14)

-----------------------------------------

Praveen Fernandes(63)

-----------------------------------------

Ashley Mascarenhas(67)

-----------------------------------------

Sanford Mascarenhas(68)

Date:

**Abstract**

One serious road accident occurs every minute and 16 die on Indian roads every hour. Automobile collision avoidance systems operate under the principle that even if the human makes an error and creates a circumstance where a crash is unavoidable, with the right corrective measures the severity of an accident can be reduced. Collision avoidance systems use sensors and other parameters that are capable of detecting obstructions in front of a moving vehicle. It may then issue a warning to the driver or take any number of direct, corrective actions. The collision avoidance system may also pre-charge the brakes in conjunction with an automatic braking oremergency brake assist system. That can provide the driver with a substantial amount of braking power. In case the user does not respond to the warnings and the systems determines that a collision is imminent, it will actually engage the brakes. Monitoring and effective use of this data can reduce a potentially dangerous drivers effect on his surroundings. With real-time analysis and auditory alerts of these factors, we can increase a driver's overall awareness to maximize safety.

**Table of Contents**

| **Chapter** | **Contents** | **Page No.** |
| --- | --- | --- |
| **1** | **INTRODUCTION** | **1** |
|  | **1.1 Description** | **1** |
|  | **1.2 Problem Formulation** | **1** |
|  | **1.3 Motivation** | **2** |
|  | **1.3 Proposed Solution** | **2** |
|  | **1.4 Scope of the project** | **2** |
| **2** | **REVIEW OF LITERATURE** | **4** |
|  | **2.1 Review of Algorithms** | **4** |
|  | **2.2 Significance of NPK pH and Rainfall** | **4** |
| **3** | **SYSTEM ANALYSIS** | **5** |
|  | **3.1 Functional Requirements** | **5** |
|  | **3.2 Non Functional Requirement** | **6** |
|  | **3.3 Specific Requirements** | **6** |
|  | **3.4 Use-Case Diagrams and description** | **7** |
| **4** | **ANALYSIS MODELING** | **11** |
|  | **4.1 Data Modeling** | **11** |
|  | **4.2Class Diagram** | **14** |
|  | **4.3 Functional Modeling** | **14** |
|  | **4.4 TimeLine Chart** | **15** |
| **5** | **DESIGN** | **17** |
|  | **5.1 Architectural Design** | **17** |
|  | **5.2 User Interface Design** | **19** |
| **6** | **IMPLEMENTATION** | **21** |
|  | **6.1 Algorithms / Methods Used** | **21** |
|  | **6.2 Working of the project** | **23** |
| **7** | **TESTING** | **25** |
|  | **7.1 Test cases** | **27** |
| **8** | **RESULTS AND DISCUSSIONS** | **33** |
| **9** | **CONCLUSIONS & FUTURE SCOPE** | **41** |

Appendix

Literature Cited

Acknowledgements

**List of Figures**

| **Fig. No.** | **Figure Caption** | **Page No.** |
| --- | --- | --- |
| 3.4.1 | Use Case | 8 |
| 4.1.1 | Activity Diagram | 12 |
| 4.2 | Level 1 DFD | 13 |
| 4.3 | Sequence Diagram | 14 |
| 4.5 | Class Diagram | 14 |
| 4.6 | Timeline chart | 15 |
| 5.1 | System Architectural Flow | 17 |
| 5.2 | Flow of Soil Profile Based Agricultural System | 18 |
| 5.3 | Main Screen | 19 |
| 5.4 | Login Screen | 19 |
| 5.5 | Registration Screen | 19 |
| 5.6 | Input Screen | 19 |
| 5.7 | Output Screen | 19 |
| 6.1 | Algorithm for SVM | 26 |
| 8.1 | Crop Details | 33 |
| 8.2 | User Details | 33 |
| 8.3 | Soil \_Info Details | 33 |
| 8.4 | Rainfall Details | 39 |
| **Fig. No.** | **Figure Caption** | **Page No.** |
| 8.5 | Input case 1 | 35 |
| 8.6 | Output case 1 | 35 |
| 8.7 | Less Nutrients Case 1 | 37 |
| 8.8 | After adding nutrients case 1 | 37 |
| 8.9 | Input case 2 | 38 |
| 8.10 | Output case 2 | 38 |
| 8.11 | Less Nutrients Case 2 | 39 |
| 8.12 | After adding nutrients case 2 | 39 |
| 8.13 | History | 40 |
| 8.14 | Accuracy of SVM model | 40 |

**List of Tables**

| **Table No.** | **Table Title** | **Page No.** |
| --- | --- | --- |
| 3.4.1 | Description of Use Case | 8 |
| 5.2 | User Details | 20 |
| 5.3 | Soil\_Info Details | 20 |
| 5.4 | Rainfall Details | 20 |
| 7.1 | Security Test Details | 26 |
| 7.2 | Data flow test test | 28 |
| 7.3 | Mining test | 29 |
| 7.4 | Test case for Authorization Procedure | 30 |
| 7.5 | Test case for Retrieval Procedure | 32 |

# Chapter1 Introduction

The aim of the project is to create an Assisted Driving System (ADS). Assisted Driving System (ADS) have the potential to optimize safety and efficiency in road traffic. In order to reach this objective, human-centered design principles have to be considered. Therefore, the effects of such devices on driver behavior and emotion are often analyzed quantitatively by using Driving simulator studies that measure effects on individual level. The system senses obstacles and uses auditory signal to warn the driver of a crash and pre charges the charges the brakes to provide more braking power.

## Description

This project describes the design and implementation of an system that provides assisted deceleration to the driver in order to reduce the severity of an accident by using a continuous monitoring system. The bot will run a stimulation of car which will sense obstacles in virtual environment and it perform the necessary action it should take to counter various obstacles that Driver may face in real world. It will warn the driver of a potential crash and charge the brakes for faster response. If the driver cannot respond in time the bot takes over and applies the brakes.

## Problem Formulation

The key requisite of the bot is to calculate how much force should be applied to the brakes and time required for the car to decelerate enough in order to avoid a crash. We decided to work on this topic so that we can make help curb the carnage on Indian roads caused by negligent driving.

## Motivation

Human negligence is one of the leading cause of road accident [5]. We could reduce human errors by offloading tasks. Autonomous driving is a good example, however it is not feasible in our country as the road lanes are not marked and road signs are not visible or broken. Partial control provides a better alternative. The bot controls the deceleration of the car only if the driver does not respond to the warning of the potential accident within a given time. Reinforcement learning is a much better way of teaching a car to prevent accidents than supervised and unsupervised guidance. It can automatically teach itself without human help to find new solutions which humans otherwise could not think of.

* 1. **Proposed Solution:**

Our solution to reducing road accidents is by predicting a crash. A car will be able to predict a crash based on certain parameters. Weight, traction, speed as well as relative distance and speed from adjoining vehicles. Using these parameters, the car decides to brake with a certain magnitude. For this, we will be using the concept of Reinforcement Learning. Reinforcement learning is a learning algorithm beyond supervised guidance. In short, it gives rewards successful and penalises unsuccessful tasks. Comparable to parenting.[6]

# Chapter 2 Review of Literature

In order to design the system we need to plan all aspects of the system. We therefore, need to study various techniques and technologies that can be used to develop our system. This literature survey aims to find out how data mining works, there are a lot of researches that have been conducted in the previous paper, through which the further studies can be processed on.

1. **Review of Algorithms**

Much has been accomplished in the field of autonomous automation. We see a drastic change in autonomous vehicle technology since 1920s, when the first radio controlled vehicles were designed. In the subsequent we see fairly autonomous electric cars powered by embedded circuits in the roads. 1980s saw vision guided autonomous vehicles, which was a major milestone in technology and till date we use similar or modified forms of vision and radio guided technologies. Various semi-autonomous features introduced in modern cars such as lane keeping, automatic braking and adaptive cruise control are based on such systems. Extensive network guided systems in conjunction with vision guided features is the future of autonomous vehicles. It is predicted that most companies will launch fully autonomous vehicles by the advent of next decade. The future of autonomous vehicles is an ambitious era of safe and comfortable transportation.[3] However a country like India lacks the road infrastructure in order to implement fully autonomous cars. So we instead we are implementing Assisted Driving.

The Assisted driving agent monitors and detect an obstacle in the path of the car. If the car and steers around the obstacle. If the obstacle in front of the car is stationary the car has to stop and steer around it if there is space. If the obstacle is mobile the car has decelerate enough to avoid the crash , in most cases it means matching the speed of the obstacle in front while keeping a safe distance. One of the principal braking parameters of a vehicle is the settled longitudinal deceleration ax. If cohesion of tyres of all wheels with the road coating is fully

used, it may be theoretically calculated according to the following formula:

axn >= [0.1+0.85(φxmax-0.2)].g

where axn is longitudinal deceleration, φxmax is maximum coefficient of friction and g is acceleration due to gravity.[2]

In the proposed solution of Assisted driving , an intelligent agent has make decisions in case the user does not respond in an emergency situation. In order to train the intelligent agent we are using Deep Q learning. The agent in a given state chooses the action that gives it the maximum Q-value. Q-Values essentially define how ‘good’ a certain state is and how much potential reward is offered by moving to that state. Initially all the Q-values are set to zero. So the agent starts by taking random actions, since it has no experience yet. However, after taking random actions over many iterations, it slowly learns to accurately predict rewards i.e Q-values for each action. It does this by adjusting its predicted reward for specific state-action pairs towards the received reward every time. In Q-learning algorithm a table is maintained which maps the Q-value for each state-action combination. The agent can simply look for the highest Q-Value, and take that action. However in a case of driving, the parameters values are continuous. It would not be practical or efficient to create a table for each action and state. Instead Deep-Q Learning uses Neural Networks to learn the patterns between state and q-value, using the reward as the expected output. In Deep Q-Networks, instead of defining every single value

(angle, speed), we correlate actions to q-values.[1]

CARLA is an open-source simulator for autonomous driving research. In addition to open-source code and protocols, CARLA provides open digital assets (urban layouts, buildings, vehicles) that were created for this purpose and can be used freely. The simulation platform supports flexible specification of sensor suites and environmental conditions. An end-to-end model via reinforced learning is evaluated in controlled scenarios of increasing difficulty, and their performance is examined via metrics provided by CARLA.[4]

# Chapter 3

# System Analysis

# 3.1 Functional requirements

The functional requirement describes the core functionality of the system. According to the functions of the robot it’s divided into three subsystems as listed below. The functional requirements of each subsystem are mentioned along with it. This section includes the data and functional process requirements.

## 3.1.1 Input subsystem

* The weight of the car.
* The speed of the car.
* The traction of the wheels.
* relative speed to front objects

## 3.1.2 Processing subsystem

* Python is used for programming the the bot in simulation.
* A logarithmic function is used to find the distance of the obstacle ahead, during the simulation.
* Arduino IDE for the Arduino on the RC car.
* Reinforcement learning algorithm to find the correct deceleration rate for various situations.
* Circuit Ninja App is used for controlling the RC

## 3.1.3 Output subsystem

* The algorithm has to find the best solution for the problem, from the data obtained from the simulation.
* The RC car stops when an immovable obstacle is in front of it.
* If a vehicle in front of the car is slower, the car decelerates in order to match the speed of the car ahead.
* It stops if no space is available sideways to overtake the front car or object

## 3.2 Non-Functional requirements

## 3.2.1 Performance requirements

The system should function efficiently as per the performed simulation training. For the various input cases it should perform the necessary tasks, such that output produced is of the car predicting occurrence of crash and avoiding it.

### 3.2.2 Software quality attributes

* **Safe Software**

The software used is Carla Simulator, which is programmed using Python programming language. Once simulated, the software will contain all the training information for our system. It is then connected to our hardware system using Raspberry Pi. Once integrated with hardware, it will safely perform necessary tasks and provide desired output.

* **Durability And Maintainable hardware**

Hardware components used for system include ultrasound sensor, auditory signal to produce beeping sound and visionary signal(LED bulb) for alerting user. These components can be easily available. Also, they are highly reliable, giving us the desired results when used. They are easy to maintain and can be replaced without causing any problems in our system.

* **Portability**

Once software of our system is developed, it can be connected to hardware system of any vehicle using a Arduino. Thus, when fully developed our simulated training information can be integrated to any vehicle and it will be possible to predict crashes for any vehicle.

* **Resistance:** Resistant enough to Withstand changing environment and as well as new functionality

# 3.3 Specific requirements

## Hardware Requirements

1) Laptop or PC : Minimum 8GB and above

2) Internet : Wired/Wireless

3) Ultrasound Sensor : Module HC SR04

4) Wires : Normal and Jumper Wires

5) LED Light : Green and Red Color

6) Battery : Lipo Battery

7) Mobile : 2GB

8) Speed Sensor : LM2983N

9) Wheels : 4 Rubber wheels with cylindrical orifice

10) Arduino : GENO/UNO

11) Bread boards : Mini And Large

12) Motor : 300rpm (4 pieces)

13) Motor Driver : LM298N

## Software Requirements

1) Front end : Circuit Ninja App for hardware purpose

2) Browser : Google Chrome 1.0 and above

3) Back End : Python and Arduino IDE

4) Software -Package : Pygame for simulation , Tensorflow

## 3.4 Use-Case Diagrams and Description

A use case is a list of actions or event steps typically defining the interactions between a role, known as an actor, and a system to achieve a goal. The Use Case diagram for Assisted Driving System consists of two actors - User and Bot is shown as below :

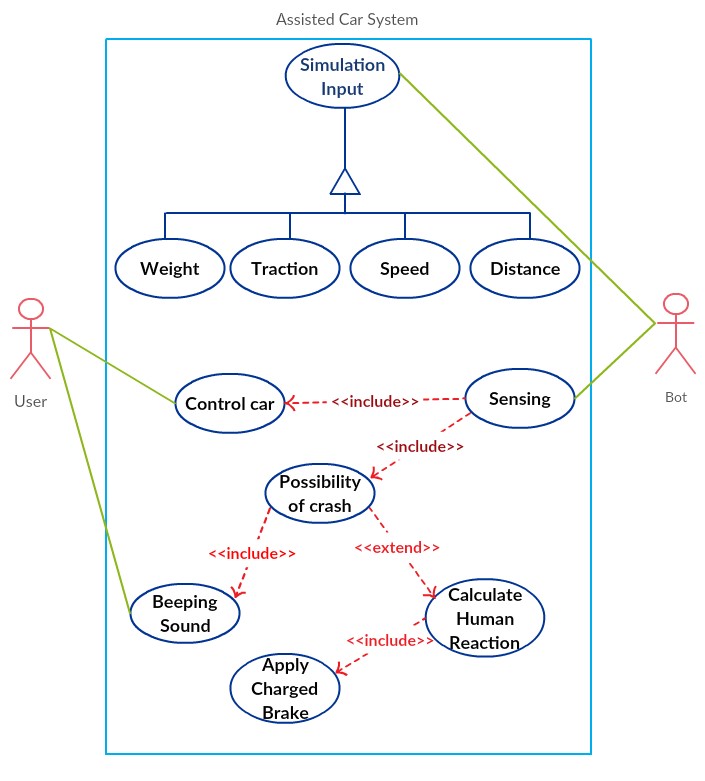


Table 3.4.1: Description of Use Case for Assisted driving system

| Actors | Functions |
| --- | --- |
| User | 1) Control Car  2) Beeping Sound |
| Bot | 1. Simulation Input  - Calculate Weight, traction, speed and distance  2. Sensing module  - Controlling car  - Finding possibility of crash  - Process the parameter received from Simulation input  - Calculating the human reaction time  - Beeping Sound  - Apply charged brakes |

# Chapter 4

# Analysis Modeling

In level 0 DFD we have the components as the environment and the crash avoidance , the major processes are the apply breaks and to match speed with the car ahead.

4.1.2 DFD Level 1:

Level 1 of the DFD shown in fig 4.2, consists of details of the system. These are the various processes:

1. Environment
2. Signal User
3. Apply brake if user does not respond
4. Crash Avoidance

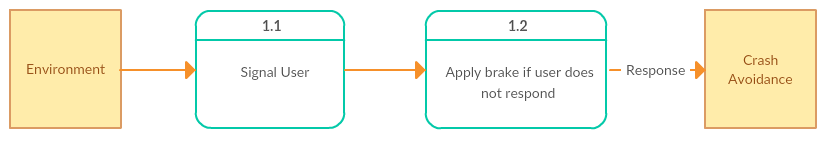


Figure 4.2: Level 1 DFD

**4.2 Activity Diagram**

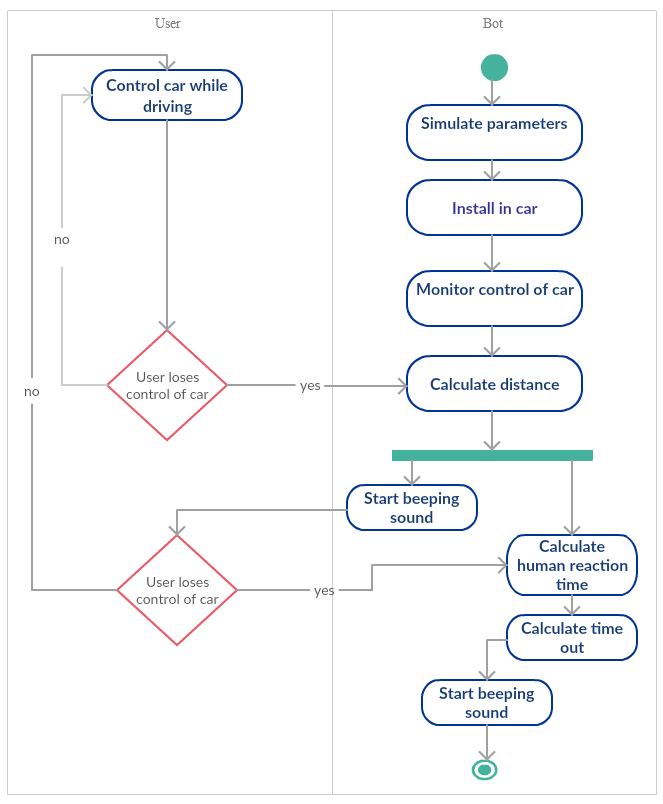
****

Figure 4.3: Activity Diagram

The sequence of the project as shown in fig 4.3 is as follows:

User Process:

* + - 1. The farmer will first register into the application.
      2. Once he registers, he can login to the system
      3. He enters the values and the views the output produced by the application.
      4. He can then logout of the application.

Application:

The application validates the farmer details.

It also is responsible for the training and testing of the dataset i.e. running the algorithm.

Once the training and testing is done, its displays the result to the farmer.

**4.3 Class Diagram**

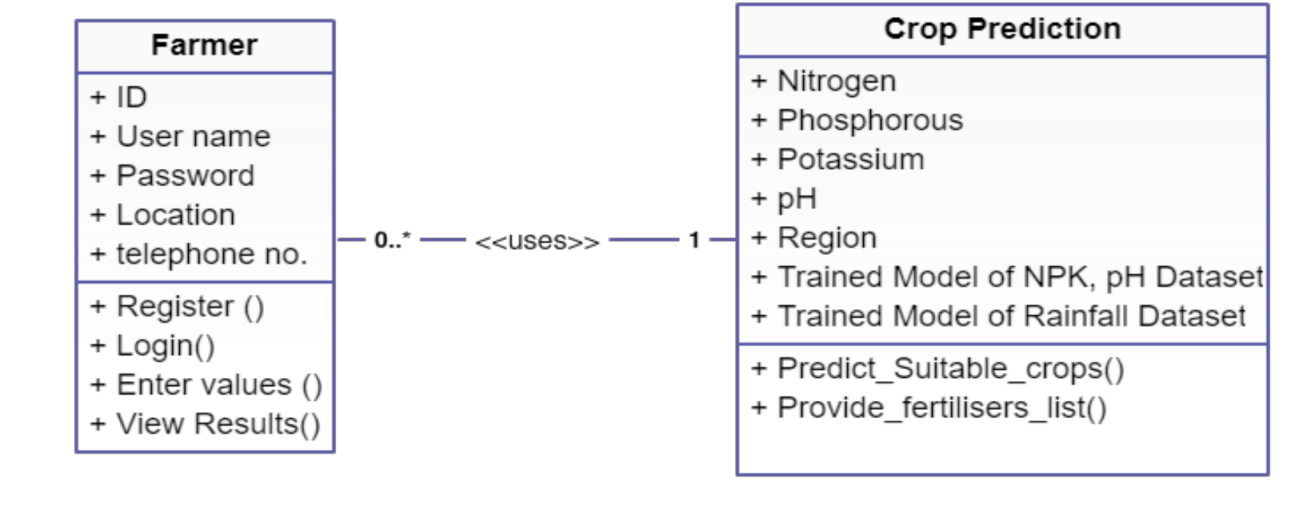
****

Figure 4.5: Class Diagram

Our application consists of two classes i.e. farmer and crop prediction. Each farmer would have a user ID, user name, password, location as attributes. The farmer can register, login, enter values i.e. NPK and pH as well as location and also view results.

The crop prediction class will deal with all the prediction attributes and function i.e. training dataset , predicting crops and fertilizers required

**4.4 Timeline Chart**

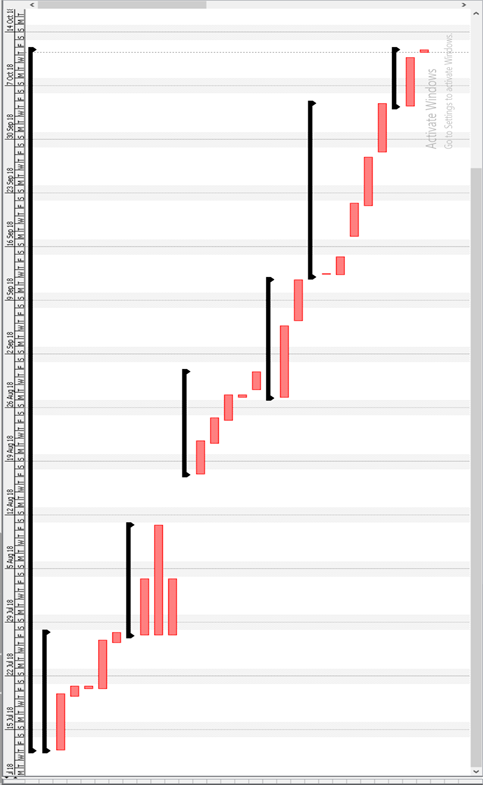


Figure 4.6: Timeline Chart

# 

# 

# Chapter 5

# Design

## Component Diagram

## The architecture of the system is depicted in the fig 5.1 which is explained in the subsequent sections.

**5.1.1 Requirements Gathering:**

In this application we will use Python API , Python Client, Component and Hardware are the components that are included in the Architectural Diagram. The components included are weight, traction and speed.

**5.1.2 Input:**

The input would be processed from the python API and the Python Client to

**5.1.3 Processing and output:**

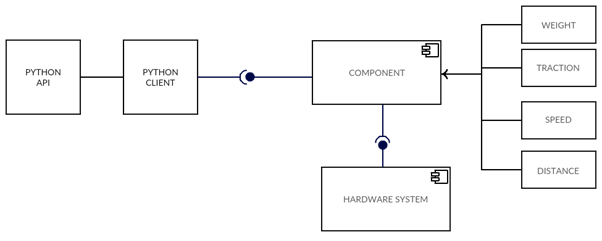
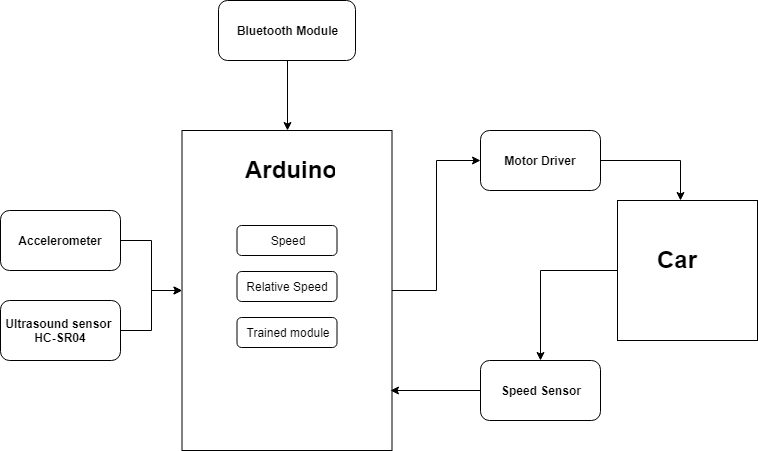
After getting a testing data system will start processing of that data with the training data and after comparing the both data i.e. testing data with training data system will provide the output as the soil fertility and according to soil fertility level predict crops.  ****

Fig 5.1 Component Diagram



### System Flow diagram

Figure 5.2: Flow of Soil Profile Based Agricultural System

The execution flow of the system is depicted in the figure 5.2. The farmer first registers in the system , he then enters the NPK and pH values as well as region of production. Once he enters the details the list of possible crops are generated. Along with it the list of fertilizers that can be applied are also generated. After that the farmer can view his previous results.

## 5.2 User Interface Design

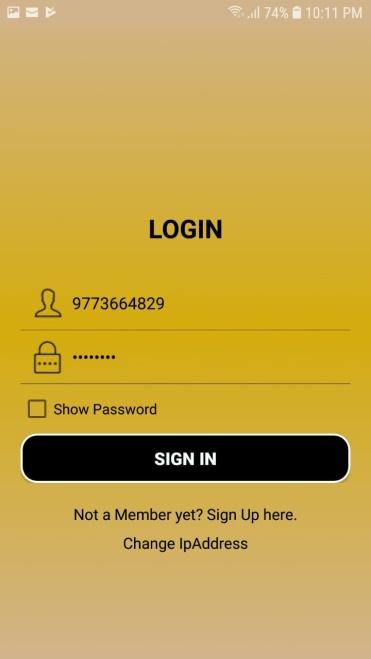
  

Fig 5.3 Main screen Fig 5.4 Login Fig 5.5 Register

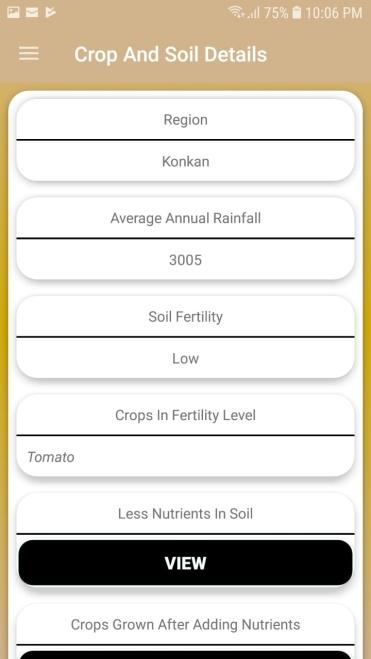
**** 

Fig 5.6 Input Screen Fig 5.7 Output Screen

## Database Design

All the crops details will be stored such as NPK, pH, rainfall, user history, etc. There are different databases for different users in the system.

Crop: The Crop database will have information about crop such as their name, N, P, K, pH, Fertility etc.

Table 5.1: Crop Details

| **Column** | crop\_id | Crop\_name | N,P,K,Ph | Fertility | Region |
| --- | --- | --- | --- | --- | --- |
| **Datatype** | Int | Varchar2 | Varchar2 | Varchar2 | Varchar2 |
| **Size** | 10 | 30 | 10 | 10 | 30 |

Rainfall: It consists of annual rainfall, region of rainfall, etc.

Table 5.2: Rainfall Details

| **Column** | Rainfall\_id | Region | Annual Rainfall |
| --- | --- | --- | --- |
| **Datatype** | Int | Varchar2 | Int |
| **Size** | 10 | 30 | 20 |

Soil\_info: Information about the soil including N, P, K, Ph, Fertility, Less Nutrients, Region, Rainfall, etc. It stores all the soil history.

Table 5.3: Soil\_info Details

| **Column** | Soil\_id | User\_id | N | P | K | ph | Fertlity | Date |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Datatype** | Int | Varchar2 | Int | Int | Int | Int | Varchar2 | Date |
| **Size** | 10 | 20 | 10 | 10 | 10 | 10 | 20 | 10 |

Table 5.4: Soil\_Info

| **Column** | Time | Status | Less\_nutrients | Nutrients | Add\_nutrients | Region |
| --- | --- | --- | --- | --- | --- | --- |
| **Datatype** | Time | Varchar2 | Varchar2 | Varchar2 | Varchar2 | Varchar2 |
| **Size** | 10 | 20 | 20 | 20 | 20 | 20 |

Users: The user database will have information about user name, id, password, age, etc.

Table 5.5: User details

| **Column** | User\_id | User\_name | Mobile no | Password | Age | City |
| --- | --- | --- | --- | --- | --- | --- |
| **Datatype** | Int | Varchar2 | Number | Varchar2 | Int1 | Varchar2 |
| **Size** | 10 | 30 | 10 | 20 | 10 | 20 |

# 

# Chapter 6

# Implementation

## Algorithms and Methods Used.

### Algorithm used for Learning: Q Learning

Q-learning is a model-free reinforcement learning algorithm. The goal of Q-learning is to learn a policy, which tells an agent what action to take under what circumstances. It does not require a model of the environment, and it can handle problems with stochastic transitions and rewards, without requiring adaptations.

1. **Q-Values or Action-Values:** Q-values are defined for states and actions. Q(S, A) is an estimation of how good is it to take the action A at the state S. This estimation of Q(S, A) will be iteratively computed using the TD- Update rule which we will see in the upcoming sections.
2. **Rewards and Episodes:** An agent over the course of its lifetime starts from a start state, makes a number of transitions from its current state to a next state based on its choice of action and also the environment the agent is interacting in. At every step of transition, the agent from a state takes an action, observes a reward from the environment, and then transits to another state. If at any point of time the agent ends up in one of the terminating states that means there are no further transition possible. This is said to be the completion of an episode.
3. **Temporal Difference or TD-Update:** The Temporal Difference or TD-Update rule can be represented as follows :

Q(st,at) = (1-α) . Q(st,at) + α ( rt + **ℽ**.maxQ(st+1,at+1) ) .. [7]

st : current state of the agent

at: current action picked according to some policy

st+1: next state where the agent ends up

at+1: next best action to be picked using current Q-value estimation i.e pick the action with the maximum Q-value in the next state.

rt: current Reward observed from the environment in Response of current action.

**ℽ:** Discounting Factor for Future Rewards. Future rewards are less valuable than current rewards so they must be discounted. Since Q-value is an estimation of expected rewards from a state, discounting rule applies here as well. The value of ℽ lies between 0 and 1.

α: Learning rate

1. **Choosing the Action to take using ε-greedy policy:** ε-greedy policy of is a very simple policy of choosing actions using the current Q-value estimations. It goes as follows :

* With probability (1-ε) choose the action which has the highest Q-value.
* With probability ε choose any action at random.

**6.1.2 How Q-Learning Works?**

In Q-learning and related algorithms, an agent tries to learn the optimal policy from its history of interaction with the environment. A history of an agent is a sequence of state-action-rewards.

We treat history of interaction as a sequence of experiences. These experiences will be the data from which the agent can learn what to do. As in decision-theoretic planning, the aim is for the agent to maximize its value, which is usually the discounted reward. Q-learning uses temporal differences to estimate the value of Q\*(s,a). In Q-learning, the agent maintains a table of Q[S,A], where S is the set of states and A is the set of actions. Q[s,a] represents its current estimate of Q\*(s,a).

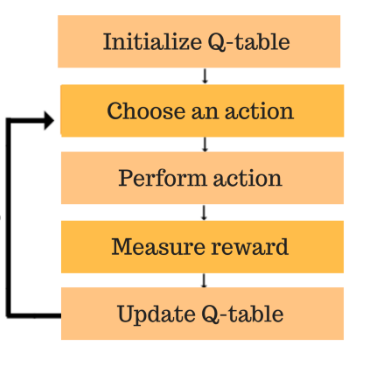


Fig 6.1.2.1 Q-Learning Algorithm

Variants of Q-Learning algorithm:

1. Deep Q-Learning

2. Double Q-Learning

Deep Q-Learning:

The DeepMind system used a deep convolutional neural network, with layers of tiled convolutional filters to mimic the effects of receptive fields. Reinforcement learning is unstable or divergent when a nonlinear function approximator such as a neural network is used to represent Q. This instability comes from the correlations present in the sequence of observations, the fact that small updates to Q may significantly change the policy and the data distribution, and the correlations between Q and the target values.[8]

The technique used experience replay, a biologically inspired mechanism that uses a random sample of prior actions instead of the most recent action to proceed. This removes correlations in the observation sequence and smooths changes in the data distribution. Iterative update adjusts Q towards target values that are only periodically updated, further reducing correlations with the target.

Double Q-learning

Because the future maximum approximated action value in Q-learning is evaluated using the same Q function as in current action selection policy, in noisy environments Q-learning can sometimes overestimate the action values, slowing the learning. A variant called Double Q-learning was proposed to correct this. Double Q-learning is an off-policy reinforcement learning algorithm, where a different policy is used for value evaluation than what is used to select the next action.[9]

In practice, two separate value functions are trained in a mutually symmetric fashion using separate experiences, QA  and QB. The double Q-learning update step is then as follows:

QAt+1(st,at) = QAt(st,at) + αt (st,at) (rt +**ℽ**QBt (st+1,arg max QAt(st+1,a)) - QAt(st,at))

and

QBt+1(st,at) = QBt(st,at) + αt (st,at) (rt +**ℽ**QAt (st+1,arg max QBt(st+1,a)) - QBt(st,at))

Now the estimated value of the discounted future is evaluated using a different policy, which solves the overestimation issue.

**6.1.3 Q-Learning used in our project:**

In our application we use Deep Q-Learning in order to train our driving agent. Deep Q-Learning uses Neural Networks to learn the patterns between state and q-value, using the reward as the expected output. These are called Deep Q-Networks. In Deep Q-Networks, instead of defining every single value (Steering wheel angle, speed), we correlate actions to q-values. For example, turn steering wheel 5 degrees to the left is a potential action. So is 4 degrees to the right, and change to 50kph, and change to 30kph. The rewards are defined as follows:

We adopted a +-1 reward system. Each frame the agent is awarded a maximum of +1 if no collision is detected. In an effort to encourage the agent to observe the speed limit the +1 reward for not crashing is multiplied by the factor below.

rewardFactor = **(** speedLimit - abs(speedLimit - agentSpeed) **) /**speedLimit

This factor decreases the reward as the agents velocity diverges from a predetermined speed limit.

* 1. **Codes:**

**Python File:**

for i in range(num\_r\_to\_l):

obs= obstacle.Obstacle(random.randint(CONST.OBS\_LN\_RtoL\_MIN,CONST.OBS\_LN\_RtoL\_MAX), "r\_to\_l", art.cars['gray'], car.rect.centerx, car.rect.centery, other\_obs = obstacles)

all\_sprites.add(obs)

obstacles.add(obs)

def initSimulation(car, state, filling\_buffer = False, x\_dist = 0, lane=2):

global obstacles

global all\_sprites

obstacles.empty()

all\_sprites.empty()

all\_sprites.add(car)

state.reset()

obs\_x = [220,220]

obs\_lanes = [2,3]

x\_dist = int(random.uniform(0,0.5)\*CONST.SCREEN\_WIDTH)

if filling\_buffer:

lane = random.randint(1,3)

for obst\_data in range(len(obs\_x)):

while (lane == obs\_lanes[obst\_data] and abs(x\_dist - obs\_x[obst\_data]) < CONST.CAR\_SAFE\_BUBBLE):

x\_dist = int(random.uniform(0,0.5)\*CONST.SCREEN\_WIDTH)

car.reInit(x\_dist, lane)

initStaticObstacles(xPos=obs\_x, lanes=obs\_lanes)

car.updateSensors(obstacles)

total\_frames = 0 # Frame Counter

epochs = 50000 # Repeat how ,many times

epoch\_cnt = 0 # Epoch counter

gamma = 0.9 # Learning rate for replay scenarios

epsilon = 1 # Greedy or decisive action, will reduce overtime

leave\_program = False # Leave program

batch\_size = 30 # Batch size for replay

buffer = 30000 # Size of buffer

replay = [] # List of stored experiences

reward = 0 # Did the car perform good or bad? keep track

\_\_console\_data\_print\_frequency = 240

for i in range(epochs):

initSimulation(car, state, filling\_buffer = True if len(replay) < buffer else False)

pigs\_fly = False

frames\_this\_epoch = 0

while not pigs\_fly:

\_\_console\_string = ""

\_\_console\_string += "FRAME: {0} -- ".format(frames\_this\_epoch)

clock.tick(CONST.SCREEN\_FPS)

screen.fill(CONST.COLOR\_BLACK)

qMatrix = dqnn.getQMat(state.state)

if (random.random() < epsilon):

action\_idx = random.randint(0,len(CONST.ACTION\_AND\_COSTS)-1)

\_\_console\_string += "random action: {0} -- ".format(CONST.ACTION\_NAMES[action\_idx])

else:

action\_idx = np.argmax(qMatrix)

\_\_console\_string += "selected action: {0} -- ".format(CONST.ACTION\_NAMES[action\_idx]

car.updateAction(action\_idx)

all\_sprites.update()

\_\_console\_string += "speed: {0} -- ".format(car.speed)

collisions = pygame.sprite.spritecollide(car, obstacles, False)

car.updateSensors(obstacles) # Sensor update

state.update(car.sensor\_data) # Update state with new data

next\_qMatrix = dqnn.getQMat(state.state)

reward = car.reward

if (collisions or car.out\_of\_bounds):

pigs\_fly = True

reward=CONST.REWARDS['terminal\_crash'] print("terminal\_crash\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

if car.isAtGoal():

pigs\_fly = True

reward = CONST.REWARDS['terminal\_goal']

car.terminal = True

print("terminal\_goal!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!")

if frames\_this\_epoch > CONST.TAKING\_TOO\_LONG:

pigs\_fly = True

print("RESET!!!!! TAKING TOO LONG!!!!!!!!!!!!!!!!!")

\_\_console\_string += "Reward: {0} -- Epsilon: {1} -- Epoch: {2} -- Total\_Frames: {3}".format(reward, epsilon, epoch\_cnt, total\_frames)

if len(replay) < buffer:

replay.append((copy.deepcopy(state\_0), copy.deepcopy(action\_idx), copy.deepcopy(reward), copy.deepcopy(state.state))

if log\_data:

states0.append(copy.deepcopy(state\_0))

actions.append(copy.deepcopy(action\_idx))

rewards.append(copy.deepcopy(reward))

states1.append(copy.deepcopy(state.state))

epoch\_cnt = 0# keep epochs at zero until buffer if full

else:

CONST.SCREEN\_FPS = 50

#h = (h+1)%buffer

replay.append((copy.deepcopy(state\_0), copy.deepcopy(action\_idx), copy.deepcopy(reward), copy.deepcopy(state.state)))

# print("action: {0}, reward: {1}, ".format(action\_idx, reward))

batch = random.sample(replay, batch\_size)

target\_batch = []

for element in batch:

replay\_old\_state, replay\_action\_idx, replay\_reward, replay\_new\_state = element

q\_mat\_old = dqnn.getQMat(replay\_old\_state)

y = copy.deepcopy(q\_mat\_old[0]

q\_mat\_new = dqnn.getQMat(replay\_new\_state)[0]

q\_val\_new = max(q\_mat\_new)

if (replay\_reward == -1):

q\_update = replay\_reward

else:

q\_update = replay\_reward + (gamma\*q\_val\_new)

y[replay\_action\_idx] = q\_update

target\_batch.append(y)

if total\_frames % 100 == 0:

dqnn.fitBatch([row[0] for row in batch], target\_batch, save=False, verbose=True, iteration\_count=total\_frames-buffer)

elif total\_frames % 10001 == 0:

dqnn.fitBatch([row[0] for row in batch], target\_batch, save=True, verbose=False, iteration\_count=total\_frames-buffer)

else:

dqnn.fitBatch([row[0] for row in batch], target\_batch)

if epsilon > 0.1:

epsilon -= 1/100000

if moving\_obstacles:

for obs in obstacles:

if obs.out\_of\_range:

obs.reInitObs(0, CONST.LANES[random.rand(CONST.CAR\_LANE\_MIN,CONST.CAR\_LANE\_MAX)], obstacles)

if car.rect.x > CONST.SCREEN\_WIDTH + CONST.SCREEN\_PADDING:

pigs\_fly = True

all\_sprites.draw(screen)

center\_guard = CONST.LANES[3] + CONST.LANE\_WIDTH//2

color = CONST.COLOR\_ORANGE

for lane in CONST.LANES:

pygame.draw.line(screen, color, (0, lane-CONST.LANE\_WIDTH//2), (CONST.SCREEN\_WIDTH, lane-CONST.LANE\_WIDTH//2))

color = CONST.COLOR\_WHITE

pygame.draw.line(screen, CONST.COLOR\_ORANGE, (0, CONST.LANES[len(CONST.LANES)-1] + CONST.LANE\_WIDTH//2), (CONST.SCREEN\_WIDTH, CONST.LANES[len(CONST.LANES)-1] + CONST.LANE\_WIDTH//2)

pygame.draw.circle(screen, CONST.COLOR\_ORANGE, (car.carrot), 5)

pygame.draw.circle(screen, CONST.COLOR\_ORANGE, (300, int(CONST.LANES[3] + CONST.LANE\_WIDTH//2)), 4)

for beam in car.lidar.beams:

pygame.draw.line(screen, beam.color, (beam.x1, beam.y1), (car.rect.centerx, car.rect.centery))

if state.setLivePlot:

print("PLOTTING")

state.plotState(True)

for event in pygame.event.get():

# Check for closing window

if event.type == pygame.QUIT:

pigs\_fly = True

leave\_program = True

dqnn.session.close()

if event.type == pygame.KEYDOWN:

if event.key == pygame.K\_p:

state.setLivePlot = not state.setLivePlot #toggle live plotting

action\_idx = 1

if event.key == pygame.K\_UP:

\_\_console\_data\_print\_frequency += 1

print("Print Frequ every {0} frames".format(\_\_console\_data\_print\_frequency))

if event.key == pygame.K\_DOWN:

\_\_console\_data\_print\_frequency -= 1

print("Print Frequ every {0} frames".format(\_\_console\_data\_print\_frequency))

if event.key == pygame.K\_RIGHT:

\_\_console\_data\_print\_frequency += 10

print("Print Frequ every {0} frames".format(\_\_console\_data\_print\_frequency))

if event.key == pygame.K\_LEFT:

\_\_console\_data\_print\_frequency -= 10

print("Print Frequ every {0} frames".format(\_\_console\_data\_print\_frequency))

if event.key == pygame.K\_q:

epsilon += 0.05

if epsilon > 1: epsilon = 1

print("Epsilon now: {0}".format(epsilon))

if event.key == pygame.K\_a:

epsilon -= 0.05

if epsilon < 0.1: epsilon = 0.1

print("Epsilon now: {0}".format(epsilon))

if \_\_console\_data\_print\_frequency <= 0: \_\_console\_data\_print\_frequency = 1

if total\_frames % \_\_console\_data\_print\_frequency == 0:

print(\_\_console\_string, os.linesep)

print("q\_matrix: {0} -- ".format(qMatrix))

print("\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_", os.linesep)

frames\_this\_epoch += 1

total\_frames += 1

# After everything, flip display

pygame.display.flip()

epoch\_cnt += 1

if epoch\_cnt == epochs-1: epoch\_cnt = epochs - 2

if leave\_program: break

if log\_data:

log.logData(fileNames, toLog)

# Data Logging

states0.clear()

actions.clear()

rewards.clear()

states1.clear()

dqnn.session.close()

pygame.quit();

**Arduino File:**

**Code:-**

int motorLpin1=2;

int motorLpin2=3;

int motorRpin1=4;

int motorRpin2=5;

int motorLpwm=10;

int motorRpwm=11;

const int trigPin = A1;

const int echoPin = A0;

int linear\_dist;

int left\_speed=6;

int right\_speed=7;

int rps;

int motorSpeed=125;

int turn=50;

long duration;

int distance;

void setup() {

Serial.begin(9600);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(13, OUTPUT);

pinMode(12, OUTPUT);

pinMode(motorLpin1,OUTPUT);

pinMode(motorLpin2,OUTPUT);

pinMode(motorRpin1,OUTPUT);

pinMode(motorRpin2,OUTPUT);

pinMode(motorLpwm,OUTPUT);

pinMode(motorRpwm,OUTPUT);

}

void loop() {

String input="";

while(Serial.available()){

input+=(char)Serial.read();

delay(5);

}

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance= duration\*0.034/2;

if(input=="n"){

stp();

}

else if(input=="F"){

fwd();

}

else if(input=="R"){

rev();

}

else if(input.indexOf("TL")>-1){

lft();

}

else if(input.indexOf("TR")>-1){

rght();

}

else if(input!=""){

motorSpeed=input.toInt();

}

*//Speed Sensor*

if( left\_speed > right\_speed )

{

rps = left\_speed;

}

else if( left\_speed < right\_speed )

{

rps = right\_speed;

}

else

{

rps = (left\_speed+right\_speed)/2;

}

*//RPM TO Linear velocity*

*=>Linear distance as it calculated per sec*

linear\_dist=(0.35\*rps/(2\*9.8\*0.8))+(0.35\*rps\*1.5)+10;

*//braking dist +stopping dist+safe dist*

*//Test cases*

if(distance <= linear\_dist)

{

digitalWrite(13, HIGH);

digitalWrite(12, LOW);

//object infront of car is detected

if(motorSpeed!=0)

{

motorSpeed=motorSpeed-45;

if(distance <= linear\_dist)

{

motorSpeed=0;

}

}

}

else

{

digitalWrite(13, LOW);

digitalWrite(12, HIGH);

*//no object infront of the car*

if(motorSpeed==0)

{

motorSpeed=125;

}

}

}

void fwd(){

analogWrite(motorLpwm,motorSpeed);

analogWrite(motorRpwm,motorSpeed);

digitalWrite(motorLpin1,0);

digitalWrite(motorLpin2,1);

digitalWrite(motorRpin1,1);

digitalWrite(motorRpin2,0);

}

void rev(){

analogWrite(motorLpwm,motorSpeed);

analogWrite(motorRpwm,motorSpeed);

digitalWrite(motorLpin1,1);

digitalWrite(motorLpin2,0);

digitalWrite(motorRpin1,0);

digitalWrite(motorRpin2,1);

}

void lft(){

analogWrite(motorLpwm,motorSpeed-turn);

analogWrite(motorRpwm,motorSpeed+turn);

digitalWrite(motorLpin1,1);

digitalWrite(motorLpin2,0);

digitalWrite(motorRpin1,1);

digitalWrite(motorRpin2,0);

}

void rght(){

analogWrite(motorLpwm,motorSpeed+turn);

analogWrite(motorRpwm,motorSpeed-turn);

digitalWrite(motorLpin1,0);

digitalWrite(motorLpin2,1);

digitalWrite(motorRpin1,0);

digitalWrite(motorRpin2,1);

}

void stp(){

analogWrite(motorLpwm,0);

analogWrite(motorRpwm,0);

digitalWrite(motorLpin1,1);

digitalWrite(motorLpin2,1);

digitalWrite(motorRpin1,1);

digitalWrite(motorRpin2,1);

}

# Chapter 7

# Testing

The testing is performed based on the algorithms used in the system. The main flow of the system depends upon whether the used algorithms function properly or not.

White Box testing is performed to test the functionality of the system. Functional testing is done on three main steps to guarantee proper working of the system.

1. Security (Identification of the user)
2. Data Flow
3. Mining (crop prediction)

## Test Module 1: Security

The security of the database is maintained by avoiding unauthenticated access by any attacker. Whenever person wants to access the data or view history of soil he first needs to login which is the matched with the values stored in database. If the values match the person is authenticated and granted an access.

Table 7.1: Security Test Table

| Validation Module | Security |
| --- | --- |
| Scenario | Securing User’s history |
| Triggering Event | Attempt for login |
| Brief Description | To avoid any data theft via identity spoofing, the login details is considered. |

| Related Modules | User, Soil\_info | |
| --- | --- | --- |
| Flow of Activities | Actor | System |
| 1.User is assigned with user name and password  . | 1.1.System accepts the details |
|  | 1.2. The received details are verified |
|  | 1.3. If details match with stored details grant an access |
| 2. Soil information is visible to the user | 2.1 After getting an access the user requests for soil information via View history tab.  2.2 The system then collects the data from server and displays it to user. |
| Conclusion | If tries to steal the soil information with mobile number or user id he cannot get access to database for password which is secured by hiding it while entering. So unauthorized authentication is prevented. | |

## 

## Test Module 2: Data Flow

The flow of data into the system is must to show proper information when asked. All modules are interlinked therefore changes made in one module must be reflected in the other without any inconsistency.

1. User -> Crop ->Soil\_info
2. Crop ->rainfall

One of the flows is tested below where the user checks for crops to be cultivated and then view his history.

Table 7.2: Data Flow Test Table

| Validation Module | Data Flow | |
| --- | --- | --- |
| Scenario | Making proper data available to the user | |
| Triggering Event | User uses crop prediction | |
| Brief Description | When the crop list is displayed during prediction should be updated in database and should be shown in history as well.  . | |
| Related Modules | User, Crops, Soil\_info | |
| Flow of Activities | Actor | System |
| 1. User enters the required parameters for crop prediction | 1.1.System gets all the details of soil and uses data mining algorithm to predict the list of crops |
| 2. List of crops to be grown is displayed | 2.1. The list of crops is shown to the user. |
|  | 2.2. Same changes are updated on database |
|  | 3. User clicks on view history tab | 3.1.Displays soil history |

| Conclusion | With the help of view history feature it becomes easy for user to keep of track of previous grown crops with the parameter values of soil and also gets to know which year he had cultivated the specific crop with rainfall details. |
| --- | --- |

## Test Module 3: Mining

The system is developed in order to help farmer with cultivation decisions. Our system aims are predicting crops and also suggesting the additional nutrients that can be added. Thus proper prediction of crops is necessary based on certain conditions.

Table 7.3: Mining Test Table

| Validation Module | Mining | |
| --- | --- | --- |
| Scenario | Predicting crops | |
| Triggering Event | User clicks on suggest crops | |
| Brief Description | User will be showed a list of crops that he can grow based on NPK pH and rainfall/region. | |
| Related Modules | User, Crops, Rainfall | |
| Flow of Activities | Actor | System |
| 1. User enters all the parameters | 1.1.System receives all the details including NPK &ph, Rainfall, Region  . |
|  | 2. User clicks on suggest crop | 2.1. The list of crops predicted are shown to the user with fertility level |
|  | 2.2. It also suggest additional nutrients for more crops |

| Conclusion | Mining is performed to predict crops and fertility level of soil with additional nutrients required to grow more crops. |
| --- | --- |

## Miscellaneous Tests

Test the working of some of the features provided by the system. To carry out this testing the test cases are developed for each of the modules in the system.

Table 7.4: Test Case Template for authorization procedure

| Project Name: Soil Profile Based Agricultural System | |
| --- | --- |
| Test Case Template | |
| Test Case ID:001 | Test Designed by: . |
| Test Priority (Low/Medium/High): High | Test Designed date: 8thMarch, 2019 |
| Module Name: User | Test Executed by: Samiksha R. |
| Test Title: To check whether user is authorized. | Test Execution date: 9thMarch, 2019 |
| Description: Test the authentication procedure | |
| Pre-conditions: User must have internet, user name/mobile number and password | |
| Dependencies: Internet | |

| Step | Test Steps | Test Data | Expected Result | Actual Result | Status (Pass/Fail) |
| --- | --- | --- | --- | --- | --- |
| 1 | Start | ------ | ------ | ------ | ------ |
| 2 | Is Internet on? | ------ | The Internet must be enabled | The Internet is enabled | Pass |
| 3 | Is user registered? | ------ | User must be registered | User is registered | Pass |
| 4 | User enters username and password | User name received (samiksha)  Password received  (12345678) | Login details should be correct and password should be hidden with \* | Login details are correct and password is hidden with \* | Pass |
| 5 | Validation of details | ------ | Matching the credentials with the ones stored in database. | Matching the credentials with the ones stored in database. | Pass |
| 7 | Data retrieved |  | Login successful | Login Successful | Pass |

Table 7.5: Test Case Template for data retrieval procedure

| Project Name: Soil Profile Based Agricultural System | |
| --- | --- |
| Test Case Template | |
| Test Case ID:002 | Test Designed by: . |
| Test Priority (Low/Medium/High):  High | Test Designed date: 8thMarch, 2019 |
| Module Name: User | Test Executed by:NISHITA D. |
| Test Title: To check whether user can view the crops predicted | Test Execution date: 10thMarch, 2019 |
| Description: Test the retrieval procedure | |
| Pre-conditions: User must have Internet. | |
| Dependencies: Internet, soil reports for NPK & pH | |

| Step | Test Steps | Test Data | Expected Result | Actual Result | Status (Pass/Fai l) |
| --- | --- | --- | --- | --- | --- |
| 1 | Start | ------ | ------ | ------ | ------ |
| 2 | Is Internet on? | ------ | The Internet must be enabled | The Internet is enabled | Pass |
| 3 | Is user registered? | ------ | user must be registered | user is registered | Pass |
| 4 | user logs in the system | Username: Samiksha Password:  12345678 | User should be able to gain access | Usergains access | Pass |

| 5 | User enters all the parametrs | ------ | All parameters(NPK, pH, region) must be filled and accepted | Parameters are accepted. | Pass |
| --- | --- | --- | --- | --- | --- |
| 8 | User clicks on suggest crop |  | List of predicted crops should be shown with fertility level and additional nutrients. | List of predicted crops is shown with fertility level and additional nutrients. | Pass |

# 

# Chapter 8

# Results and Discussion

## Experimental Setup

To execute and evaluate the system with all the features we have designed the experimental set up with following integral components of the system.

### Database

It is an integral part of the project. Since the project deals with huge amount of data, maintaining a well-designed database is a must. The table in fig 8.1 stores information about all the crop details. The table in fig 8.2 stores information about the users registered in our system. The table in fig 8.3 stores information about the soil history and prediction history of our system. The table in fig 8.4 depicts the annual rainfall based on 5 regions i.e Konkan, Vidharbha, Marathwada, Khandesh and Pashchim Maharashtra.

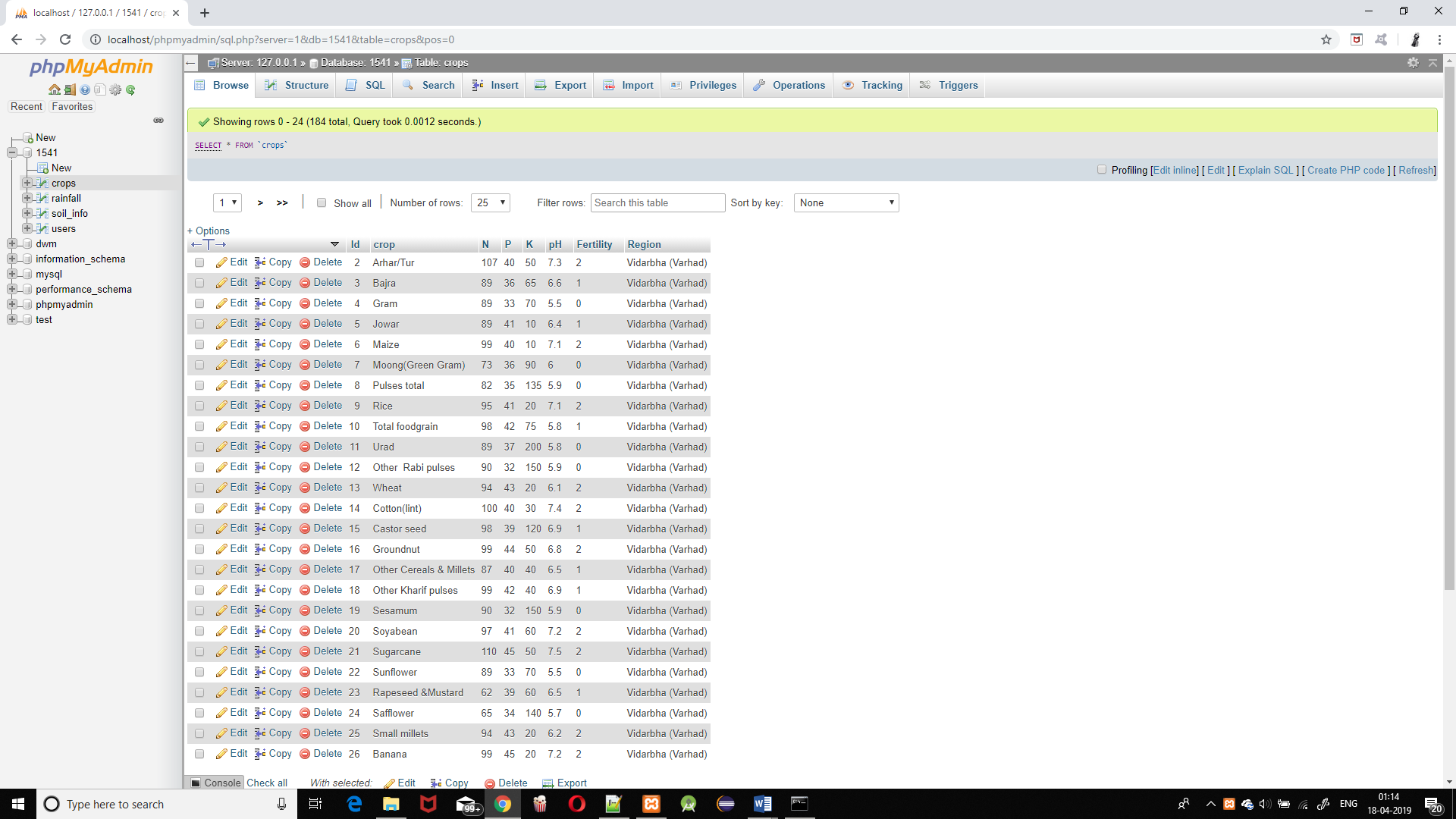


Figure 8.1: Crop Details

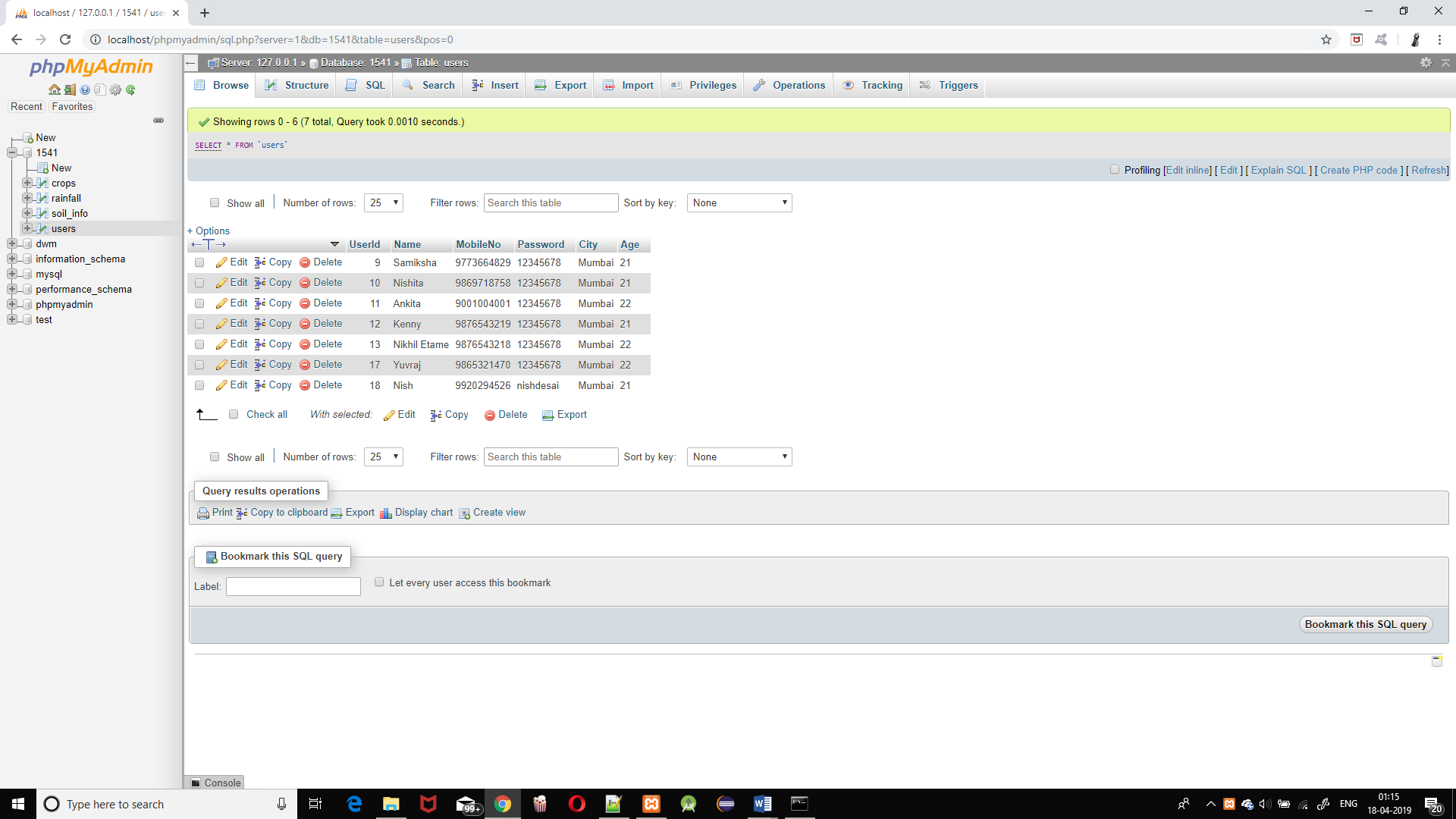


Figure 8.2: User Details

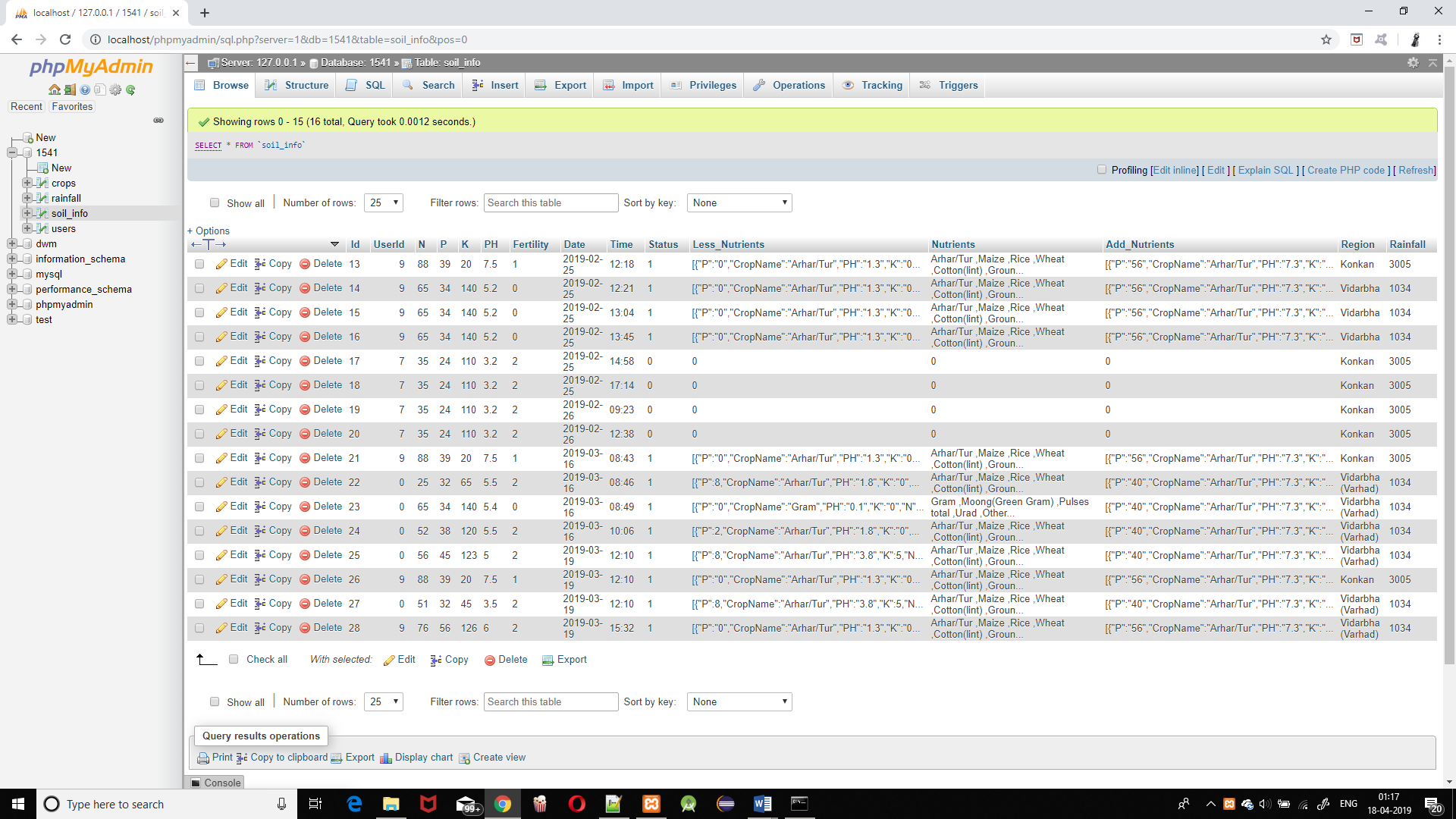


Figure 8.3: Soil\_info Details

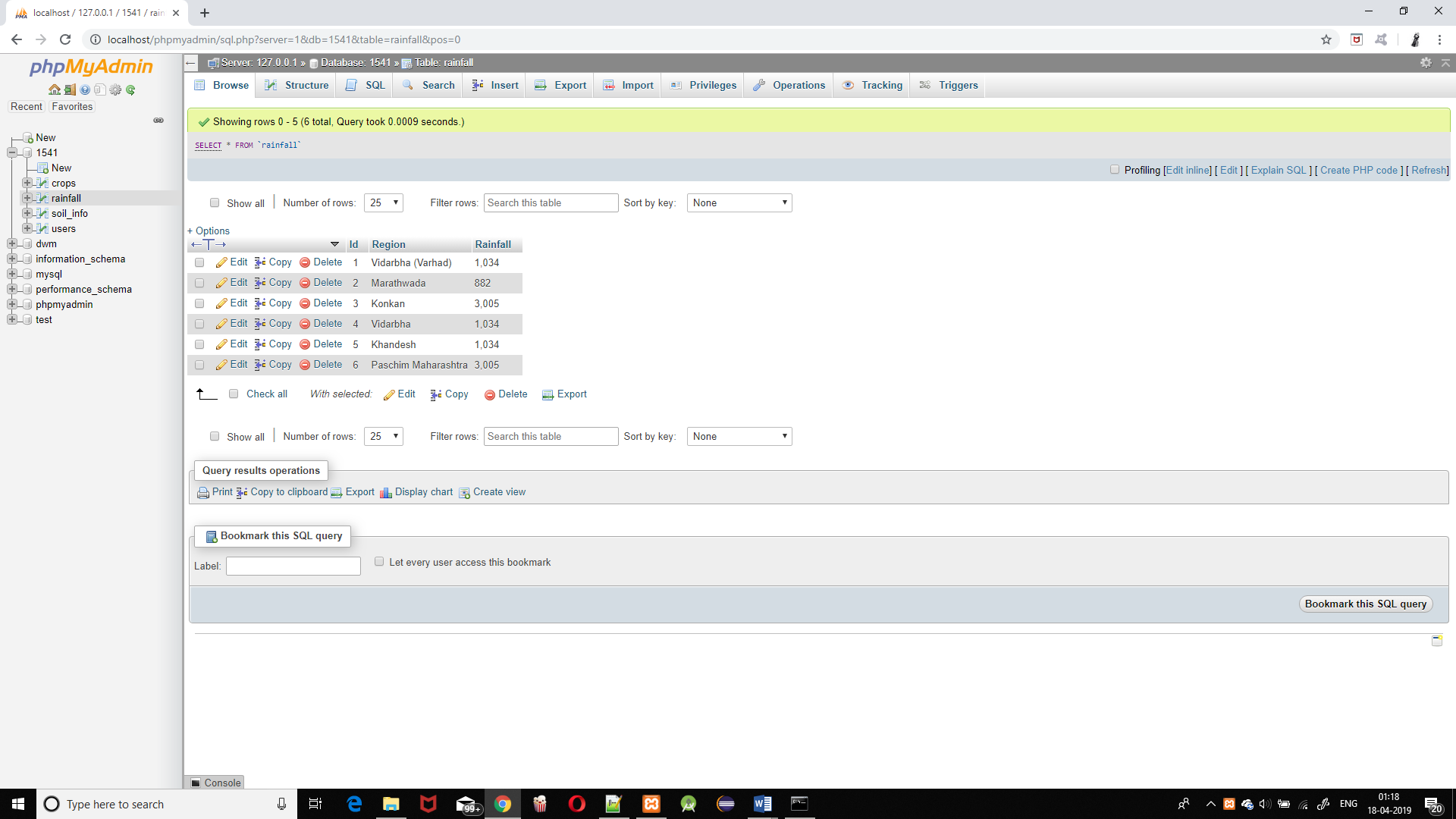


Figure 8.4: Rainfall details

## 

## 8.2 Results and Discussion with all Features of System

**Case 1:Low fertility**

****

Fig 8.5 Input case 1

In the above image the farmer inputs following values: N,P,K & pH (65,35,0,5.5 respectively). After entering the values he selects the region as Konkan from the regions specified in dropdown list.

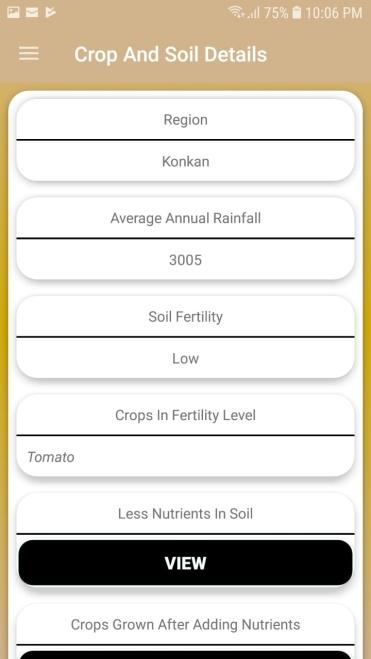


Fig 8.6 Output case 1

The following image describes the output as Region which was selected by farmer,

The annual rainfall of the region selected by farmer, Fertility level based on NPK & pH values, list of crops that can be grown in that fertility range and region, with additional tabs of Less Nutrients and Crops grown after adding nutrients.



Fig 8.7 Less nutrients case 1

The above image describes the nutrients(NPK & pH) which are less in soil based on standard values for specific crop.

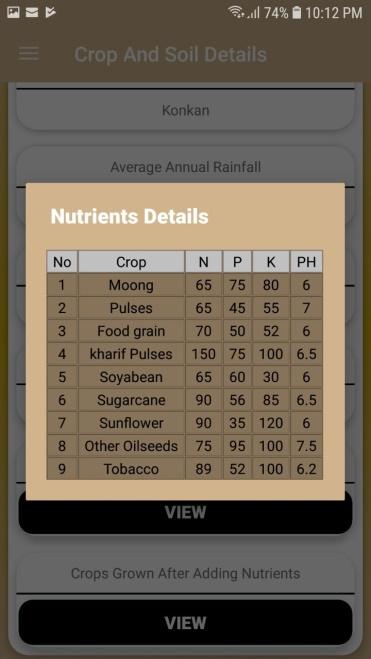


Fig 8.8: After adding nutrients case 1

The above image describes the additional crops that can be added after adding the less nutrients.

**Case 2:High fertility:**

Fig 8.9 Input case 2

In the above image the farmer inputs following values: N,P,K & pH (72,38,135,6.9 respectively). After entering the values he selects the region as Konkan from the regions specified in dropdown list.

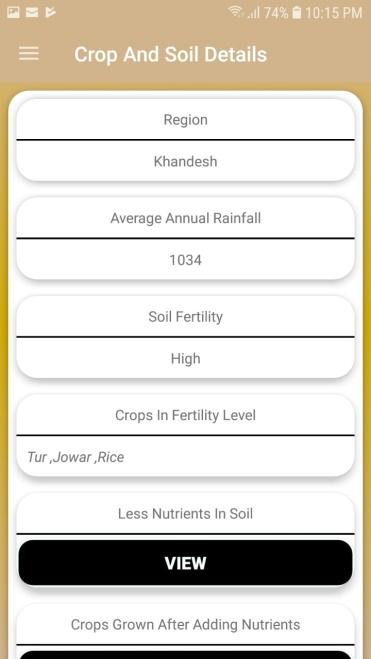


Fig 8.10 Output Case 2

The following image describes the output as Region which was selected by farmer,

The annual rainfall of the region selected by farmer, Fertility level based on NPK & pH values, list of crops that can be grown in that fertility range and region, with additional tabs of Less Nutrients and Crops grown after adding nutrients.



Fig 8.11 Less nutrients case 2

The above image describes the nutrients(NPK & pH) which are less in soil based on standard values for specific crop.



Fig 8.12 After add nutrients case 2

The above image describes the additional crops that can be added after adding the less nutrients.

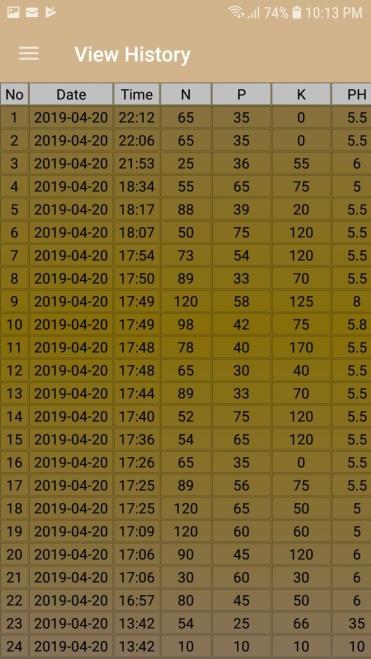


Fig 8.13 History

History tab allows user to check his past soil history i.e the lst of crops he got for various values of NPK & pH, at what time and on which day he checked that fertility.

Also by clicking on view more he can view the entire record of his previous process with Region, rainfall, crop list, less nutrients and additional crops after adding nutrients.

The above image is a Splash Screen for user to give feedback of his application is loading with a progress bar indicating the percent(%) of application loaded.

* 1. **Accuracy of SVM**

We compared two SVM models with 100 training and 33 test data and we got the accuracy as 0.84375 and 0.96875 respectively. The accuracy is shown in the fig 8.14 wherein we check if our model predicts the fertility correctly and the accuracy is 0.96875.

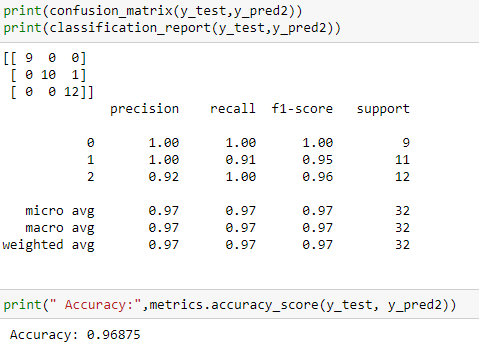
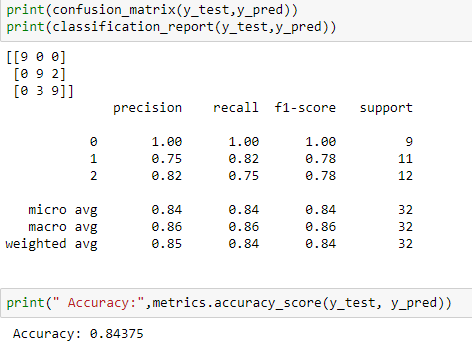


Fig 8.14 Accuracy of SVM model

# 

# Chapter 9 Conclusion and Future Work

We have created a dedicated simple solution by crafting a Soil Profile Based Agricultural System which is developed for farmers to help them decipher the soil reports.

Our application is mainly developed for the farmers. As farmers is the backbone of our nation. Thus we have successfully developed the ‘Soil Profile Based Agricultural System’, based on data mining. We have provided a list of crops a farmer can cultivate based on inputted soil attributes (NPK and pH) and rainfall of farmer’s region. In addition to this, it also suggests fertilizers that can be used to improve soil quality and thus bring more crops under successful cultivation. Additionally we also show which crops can be grown after adding the missing nutrients. This Android application is developed to solve the growing problem of crop failure. Farmers can view their previous searched result in history.

In future we also plan on adding information about pesticides that can be added to the crop to improve its growth and to prevent the plants from being affected by the pests. A graph can be generated that gives the information about the crops that are currently planted by the other farmers in a particular location. Using these information farmers can get a clear idea about the crop to be planted. Other than these features we also plan to add OCR wherein the farmer will have to just scan the reports and the NPK and pH values will be extracted from the reports making it convenient for the farmers.

# 

# Appendix I

**Terminologies**

| **N** | Sodium a primary nutrient of soil. |
| --- | --- |
| **P** | Phosphorus a primary nutrient of soil. |
| **K** | Potassium a primary nutrient of soil. |
| **SVM** | A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples |

# Literature Cited

[1]SunWei Xia; Huiyun Li; Baopu Li “A Control Strategy of Autonomous Vehicles based on Deep Reinforcement Learning”, 9th International Symposium on Computational Intelligence and Design, Year: 2016.

[2]Nerijus Kudarauskas “Analysis of emergency braking of a vehicle”, TRANSPORT – 2007, Vol XXII, No 3, 154–159.

[3]Keshav “Autonomous Cars: Past, Present and Future”, 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Year:2015.

[4]Alexey Dosovitskiy; German Ros; Felipe Codevilla; Antonio Lopez; Vladlen Koltun “CARLA: An Open Urban Driving Simulator”, 1st Conference on Robot Learning (CoRL 2017), Mountain View, United States, Year:2017.

[5]https://timesofindia.indiatimes.com/city/chandigarh/Human-negligence-overspeeding-main-causes-of-accident-Police/articleshow/ (Date: January 25, 2015)

[6] R.S. Sutton ; A.G. Barto “Reinforcement Learning: An Introduction”, Volume 9, IEEE Transactions on Neural Networks, Year:1998.

[7]<https://medium.freecodecamp.org/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc> (Date: September 4, 2018)

[8]  Takafumi Okuyama; Tad Gonsalves “Autonomous Driving System based on Deep Q Learning”, International Conference on Intelligent Autonomous Systems, Year:2018.

[9] Hado V. Hasselt, “Double Q-Learning” , Advances in Neural Information Processing Systems 23, Year:2010.

# Acknowledgements

A project is always coordinated, guided and scheduled team effort aimed at realizing a common goal. One which is executed by the team members though never attains the final goal or does not reach the far off shores unless there are light houses to guide the ship sailing astray. We would like to extend our sincere thanks to all those individuals for their kind support and help in making our endeavor a success. We are highly indebted to our institute St. Francis Institute of Technology and the Computer Engineering Department for providing us with this teaming opportunity with the required Resources to accomplish our task.

We wish to express our deep gratitude to our project guide Asst. Prof. Priya Karunakaran, of Computer Engineering Department, for all the advice, encouragement and constant supervision fee the betterment of the project. This work would not have been possible without her intellectual and valuable suggestions.

Also, a special Thanks to our respected principal Dr. Sincy George. We are also grateful to our friends and family members, who have significantly contributed to the project by providing valuable suggestions.

Sincerely,

Acvin Gonsalves

Praveen Fernandes

Ashley Mascarenhas

Sanford Mascarenhas