**1.INTRODUCTION**

Over the decades, increasing demands for supplying agri-food products have influenced agriculture patterns worldwide. Additionally, changing human lifestyles and increasing human population and urbanization have directly impacted the production, consumption of agri-food products. The financial value of strategic plants and the scarcity of natural resources for agriculture have spurred plant producers and agriculture researchers to discover new ways to overcome the food crisis. Thus far, various modern technologies and efficient strategies have been implemented in the agri-food sectors. However, reports indicate a need to find and/or improve the current agri-food tools to overcome the hunger problem and demand-supply gap by increasing production efficiency. Agricultural technology refers to the use of tools, machinery, and techniques aimed at boosting the efficiency of food production, increasing yields, and improving the quality of crops The evolution of agricultural technology dates back to the Neolithic Revolution, where the discovery of farming led to the domestication of plants and animals. This marked the transition from nomadic hunting and gathering to settled agriculture. However, over centuries, agriculture has been transformed by various technological advancements, shaping farming into a science-driven and productive sector that it is today . The advent of the Industrial Revolution in the late 18th and early 19th centuries marked a significant milestone in the evolution of agricultural technology. Introduction of mechanized tools like the cotton gin, steam tractor, and eventually, the combine harvester revolutionized agricultural practices by significantly reducing human labor and increasing productivity.

**2. LITERATURE SURVEY**

**2.1 Precision Agriculture:**

* **Title**: “Advancements in Precision Agriculture Technologies for Sustainable Farming”
* **Authors**: Smith,J.,Johnson,A.,& Lee, K
* **Journal/Conference**: Journal of Agriultural Science
* **Year:** 2019
* **Summary:** This paper reviewers recent advancements in precision agriculture technologies,including GPS-guided machinery,drones,and sensor-basedmonitoring systems. It discusses hoe these technologies enhance productivity optimize resource use, and mitigate environmental impact.

**2.2 IOT Applications in Agriculture:**

* **Title**: “Internet of things applications in Agriculture: A Review”
* **Authors:** Patel, K, & Shah,S.
* **Journal/Conference**: International Journal of Computer Applications
* **Year:** 2020
* **Summary** : This review paper explores various IOT applicatins in agriculture, such as smart irrigation systems, crop monitoring, and livestock management. It discusses the integration of IOT devices with data analytics platforms for real-time decision-making .

**2.3** **Block chain in supply Chain Management**

* **Title** : “Block chain Technology for Agricultural Supply Chain Management: A Literature Review”
* **Authors**: Gupta, R., & Kumar, A.
* **Journal/Conference**: Computers and Electronics in Agriculture
* **Year:** 2021
* **Summary:** This literature review focuses on the use of block chain technology in agricultural supply chain management. It discusses how block chain enables transparent ad secure transactions, traceability of products, and efficiency improvements, ultimately contributing to increased productivity and trust in the agricultural value chain.

**2.4** **Machine Learning for Crop Yield Prediction:**

* **Title**: “Machine Learning Techniques for Crop Yield Prediction: A Review”
* **Author**s: Singh, A, etal.
* **Journal/Conference**: Information processing in Agriculture
* **Year:**2018
* **Summary:** This review paper surveys the application of machine learning techniques for crop yield prediction. It discusses various approaches, such as regression models, neural networks, and ensemble methods, and evaluates their performance in predicting crop yields based on factors like weather data, soil conditions.

**3. SYSTEM ANALYSIS**

**3.1 Existing System:**

The existing work has used less algorithms and their accuracy are displayed low range. Any algorithm is been measured using accuracy to obtain a best model. Based on the model evaluation all the algorithms used by existing authors seems to have less accuracy.The predictions as well as the details are more important for a user to analyze and select a crop. The existing research restricts the crop recommendation of only one. And does not provide any additional values.

**3.2 Proposed System:**

Due to the changes taking place in the environment the proposed work helps to identify how to manage crops and harvest in a smart way. It guides an individual for smart farming. The aim of this work is to help an individual cultivate crops efficiently and hence achieve high productivity at low cost. It also helps to predict the total cost needed for cultivation. This would help an individual to pre-plan the activities before cultivation resulting in an integrated solution in farming.

**3.2.1 Architecture Diagram**

Data Pre-Processing

Prepared Data

**Fig 3.2.1 Architecture Diagram**

**4. SOFTWARE DESCRIPTION**

**4.1 Introduction to Python**

In today’s blog, we will provide an introduction to Python programming. This blog is dedicated to all those who are from any domain whether they are students, working employees, mechanical engineers who are willing to learn to program, and newbies. Python is the most widely used programming language by tech giants like Google, Netflix, Facebook. Therefore, it is essential to get started with Python programming. Looks interesting right, then let’s get started.

**What is Python Programming?**

With an Introduction to Python, we can understand that it is a high-level object-

Oriented programming language that was created by Guido van Rossum. Python is

Also known as a general-purpose programming language, as it is used in the domains given below:

 Web Development

 Software Development

 Game Development

 AI &amp; ML

**Why Python Programming?**

Every programming language serves some purpose or use-case according to a

domain, and Python is no exception. An introduction to Python programming can

help us understand the purpose of the language. For example, Python is widely used in data science, machine learning, and artificial intelligence due to its simplicity and powerful libraries. Similarly, JavaScript is the most popular language among web developers as it gives the developer the power to handle applications via different frameworks like React, Vue, Angular, which are used to build beautiful user interfaces. Similarly, they have pros and cons at the same time. so if we consider python it is general-purpose which means it is widely used in every domain the reason is it’s very simple to understand, scalable because of which the speed of development is so fast. Now you get the idea why besides learning python it doesn’t require any programming background so that’s why it’s popular amongst developers as well. Python has simpler syntax similar to the English language and also the syntax allows developers to write programs with fewer lines of code. Since it is open- source there are many libraries available that make developers’ jobs easy ultimately results in high productivity. They can easily focus on business logic and Its demanding skills in the digital era where information is available in large data sets

What we can do is follow the documentation which is a good starting point for learning Python. Once we are familiar with Python concepts or terminology, we can dive deeper into the language and explore more advanced topics..

Real-World Examples:

1) NASA (National Aeronautics and Space Agency): One of Nasa’s Shuttle

Support Contractors, United Space Alliance developed a Workflow Automation

System (WAS) which is fast. Internal Resources Within critical project stated that:

“Python allows us to tackle the complexity of programs like the WAS without getting bogged down in the language”. Nasa also published a website (https://code.nasa.gov/) where there are 400 open

source projects which use python.

2) Netflix: There are various projects in Netflix which use python as follow:

 Central Alert Gateway

 Chaos Gorilla

 Security Monkey

 Chronos

Amongst all projects, Regional failover is the project they have as the system

decreases outage time from 45 minutes to 7 minutes with no additional cost.

3) Instagram: Instagram also uses python extensively. They have built a photo-

sharing social platform using Django which is a web framework for python. Also,

they are able to successfully upgrade their framework without any technical

challenges.

**Applications of Python Programming:**

1) **Web Development:** Python offers different frameworks for web development like Django, Pyramid, Flask. This framework is known for security, flexibility,

scalability.

2) **Game Development**: PySoy and PyGame are two python libraries that are used

for game development

3) **Artificial Intelligence and Machine Learning**: There is a large number of open-

source libraries which can be used while developing AI/ML applications.

4) **Desktop GUI:** Desktop GUI offers many toolkits and frameworks using which we

can build desktop applications.PyQt, PyGtk, PyGUI are some of the GUI

frameworks.

**4.2 Introduction To Numpy**

**What is NumPy?**

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

**Why Use NumPy?**

In Python we have lists that serve the purpose of arrays, but they are slow to process . NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting. functions that make working with ndarray very easy.

**Why is NumPy Faster Than Lists?**

NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behavior is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures.

**Which Language is NumPy written in?**

NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

**4.3 Introduction To Pandas**

**What is Pandas?**

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.The name quot;Pandas&quot; has a reference to both &quot;Panel Data&quot;, and &quot;Python Data Analysis&quot; and was created by Wes McKinney in 2008.

**Why Use Pandas?**

Pandas allows us to analyze big data and make conclusions based on statistical theories. Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science.

What Can Pandas Do?

Pandas gives you answers about the data. Like:

 Is there a correlation between two or more columns?

 What is average value?

 Max value?

 Min value?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called cleaning the data.

What is Python Pandas used for?

The Pandas library is generally used for data science, but have you wondered why?

This is because the Pandas library is used in conjunction with other libraries that are used for data science. It is built on top of the **NumPy library** which means that a lot of the structures of NumPy are used or replicated in Pandas. The data produced by Pandas is often used as input for plotting functions in  **Matplotlib** ,

statistical analysis in **SciPy,** and machine learning algorithms in  **Scikit-learn** .

You must be wondering, Why should you use the Pandas Library. Python’s Pandas

library is the best tool to analyze, clean, and manipulate data. Here is a list of things that we can do using Pandas.

 Data set cleaning, merging, and joining.

 Easy handling of missing data (represented as NaN) in floating point as well as non-

floating point data.

 Columns can be inserted and deleted from DataFrame and higher-dimensional

objects.

 Powerful group by functionality for performing split-apply-combine operations on

**Getting Started with Pandas**

Let’s see how to start working with the Python Pandas library:

**Installing Pandas**

The first step in working with Pandas is to ensure whether it is installed in the system or not.  If not, then we need to install it on our system using the pip command.

Follow these steps to install Pandas:

**Step 1**: Type ‘cmd’ in the search box and open it.

**Step 2:** Locate the folder using the cd command where the python-pip file has been installed.

**Step 3:** After locating it, type the command:

**Data Structures in Pandas Library**

Pandas generally provide two data structures for manipulating data. They are:

** Series**

** DataFrame**

**Pandas Series**

A Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, Python objects, etc.). The axis labels are collectively.

The Pandas Series is nothing but a column in an Excel sheet. Labels need not be unique but must be of a hashable type.

The object supports both integer and label-based indexing and provides a host of

methods for performing operations involving the index.

**4.4 Introduction To Matplotlib**

Matplotlib is a cross-platform, data visualization and graphical plotting library (histograms, scatter plots, bar charts, etc) for Python and its numerical extension

NumPy. As such, it offers a viable open source alternative to MATLAB. Developers

can also use matplotlib’s APIs (Application Programming Interfaces) to embed plots in GUI applications.

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:

The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot

An OO (Object-Oriented) API collection of objects that can be assembled with

greater flexibility than pyplot. This API provides direct access to Matplotlib’s

backend layers.

**Matplotlib and Pyplot in Python**

The pyplot API has a convenient MATLAB-style stateful interface. In fact, the matplotlib Python library was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use.

**matplotlib.pyplot.figure**: **Figure** is the top-level container. It includes everything

visualized in a plot including one or more **Axes.**

**matplotlib.pyplot.axes**: Axes contain most of the elements in a plot:**Axis, Tick,**

**Line2D, Text,**etc., and sets the coordinates. It is the area in which data is plotted.

Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well. For more information about the pyplot API and interface, refer to

**What Is Pyplot In Matplotlib** **Installing Matplotlib**

Matplotlib and its dependencies can be downloaded as a binary (pre-compiled) package from the Python Package Index (PyPI), and installed with the following command:

Matplotlib is also available as uncompiled source files from GitHub. Compiling from source will require your local system to have the appropriate compiler for your OS, all dependencies, setup scripts, configuration files, and patches

**5. SYSTEM REQUIREMENTS**

**5.1 HARDWARE REQUIREMENTS:**

• System : I – 5 Processor

• Hard Disk : 250 GB SSD

• Monitor : 15 VGA Color.

• Mouse : Optical Mouse

• Ram : 8 GB.

**5.2 SOFTWARE REQUIREMENTS:**

• Operating system : Windows – 10 - 64 bit.

• Coding Language : Python 3.12.3

• Development Tool : Tkinter

• Applications : Pandas / Numpy / Scipy

• Backend : MySQL 5.6.2

• Server : XAMPP – Apache Tomcat

**6. INPUT AND OUTPUT DESIGN**

The input design is the link between the information system and the user. It

comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design

considered the following things:

 What data should be given as input?

 How the data should be arranged or coded?

 The dialog to guide the operating personnel in providing input.

 Methods for preparing input validations and steps to follow when error occur.

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system. The output form of an information system should accomplish one or more of the

following objectives.

 Convey information about past activities, current status or projections of the

**7. SYSTEM STUDY**

**7.1 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**7.2 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**7.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user.This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

**8. SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to

discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**8.1 TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields

**Functional test**

Functional tests provide systematic demonstrations that functions tested are

available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets

requirements. It tests a configuration to ensure known and predictable results. An

example of system testing is the configuration-oriented system integration test.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has

knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner

workings, structure or language of the module being tested. Black box tests, as most otherkinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases. Test strategy and approach Field testing will be performed manually and functional tests will be written indetail.

**9**. **IMPLEMENTATION**

**9.1 EVOLUTION OF DIGITAL TECHNOLOGIES IN AGRICULTURE**

1. The Onset of Digital Technologies in Agriculture

The onset of digital technologies in agriculture is a relatively recent

phenomenon. Initial digitization in agriculture emerged in the 1980s with the advent of Geographical Information Systems (GIS) and the global positioning system (GPS). These technologies facilitated spatial data collection, mapping, and analysis of farming systems. The term &#39;precision agriculture&#39; was coined around this period, representing a farming management concept based on observing, measuring, and responding to inter- and intra field variability in crops. Precision agriculture leverages technology to ensure that the crops and soil receive exactly what they need for optimum health and productivity.

2.Gradual Technological Developments and their Impact

The gradual developments in digital technologies have considerably impacted agriculture. The introduction of decision support systems (DSS) in the2000s, combining multiple sources of data, helped farmers to make informed decisions about planting, fertilizing, and harvesting crops. Wireless technology have enabled real-time data collection and processing. such as sensors, have become fundamental in monitoring soil and crop conditions and controlling automated irrigation systems .

**10. MODULE DESCRIPTION**

• Crop recommendation

• Importing Libraries and Dataset

• Descriptive Analysis

• Data Visualization

• Model Building

**11. SAMPLE SCREEN SHOTS**

**11.1 Home Page**

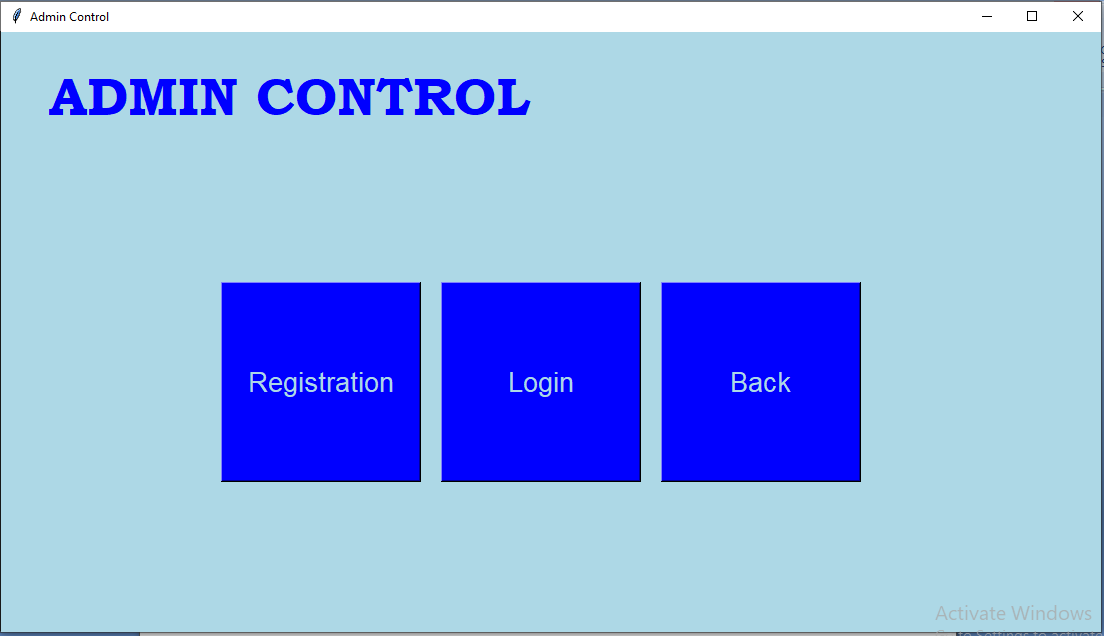
**Home Page**



**Fig 11.1 Home Page**

**11.2 ADMIN CONTROL**

**ADMIN CONTROL**



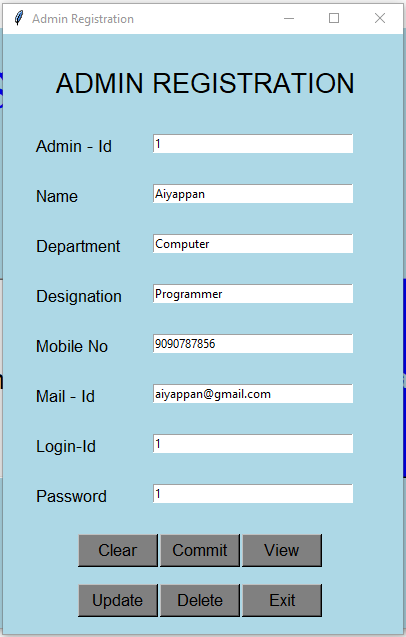
**Fig 11.2 Admin Control**

**11.3 REGISTRATION DASH-BOARD**

****

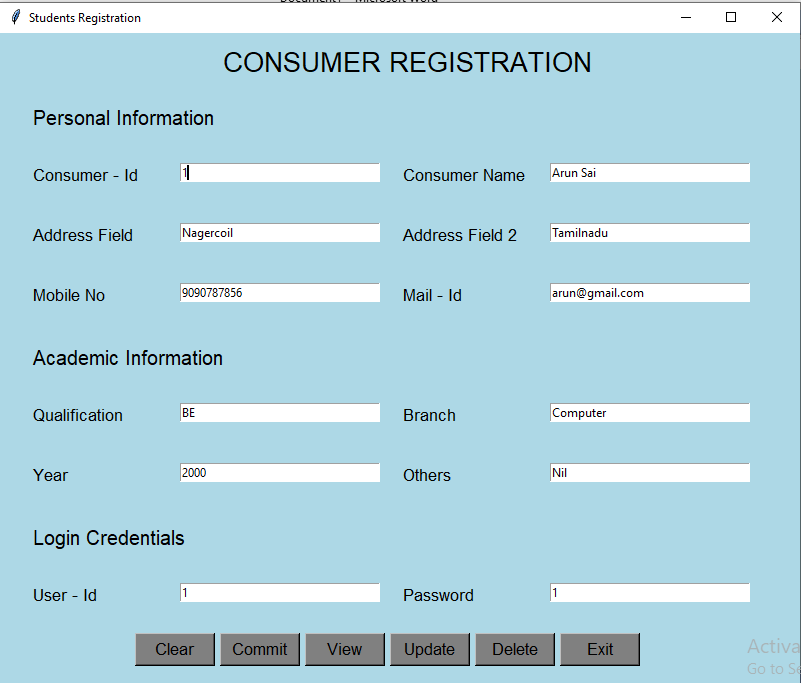
**Fig 11.3 Registration Dash-Board**

**11.4 ADMIN REGISTRATION**

****

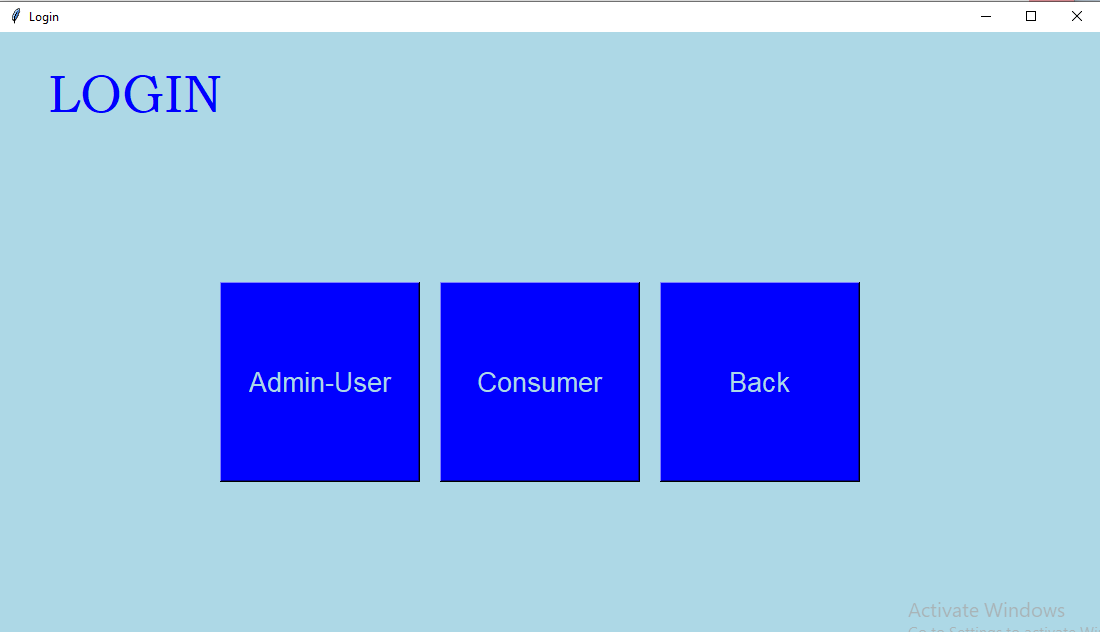
**Fig 11.4 Admin Registration**

**11.5 CONSUMER REGISTRATION**

****

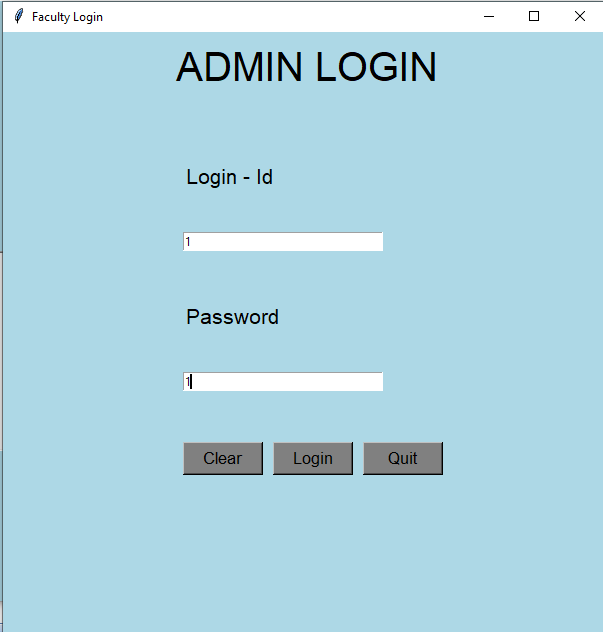
**Fig 11.5 Consumer registration**

**11.6 LOGIN CONTROL PANEL**

****

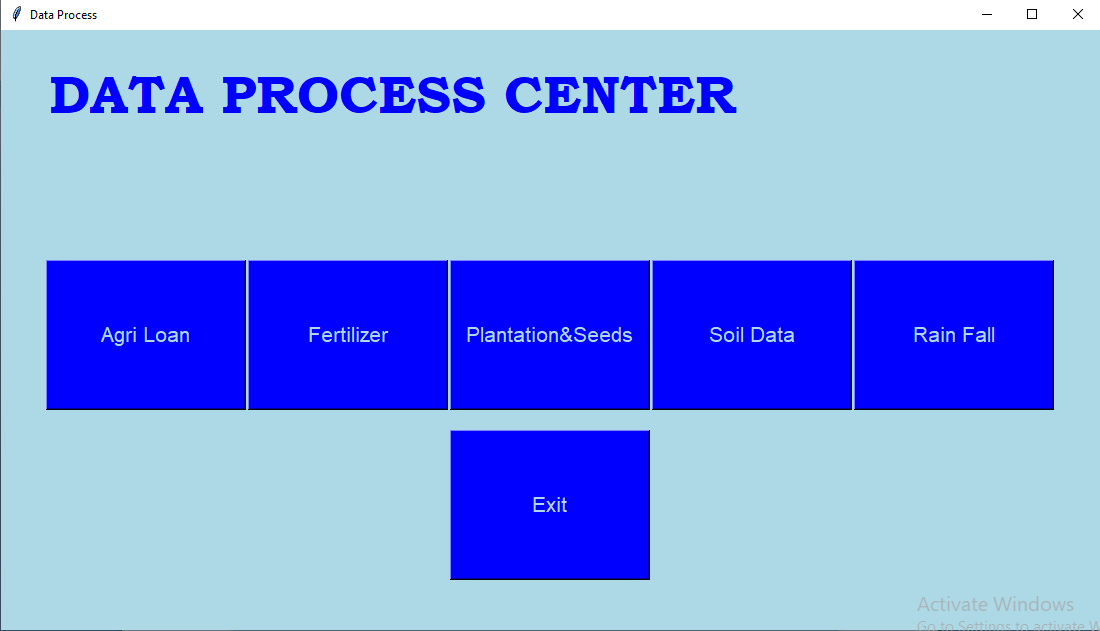
**Fig 11.6 Login Control Panel**

**11.7 ADMIN LOGIN**

****

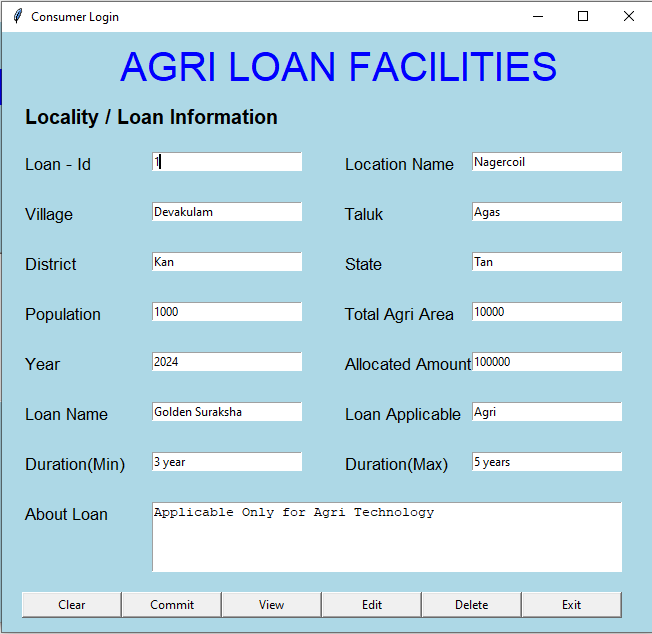
**Fig 11.7 Admin Login**

**11.8 ADMIN DATA PROCESS CENTER**

****

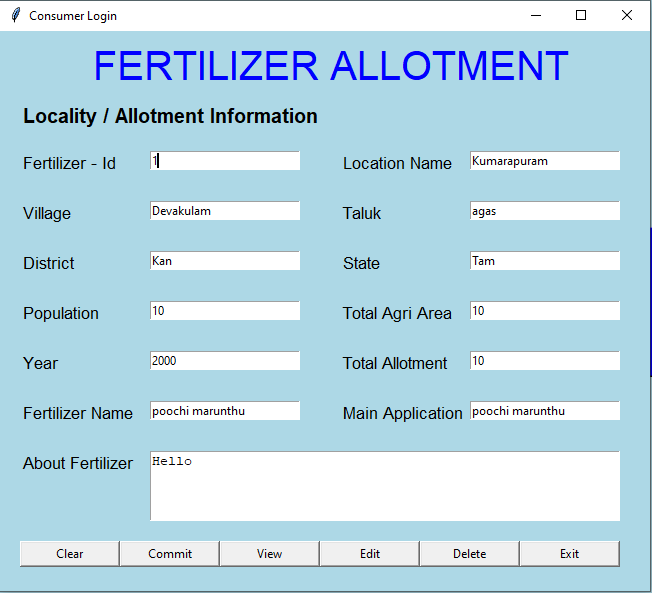
**Fig 11.8 Admin Data Process Center**

**11.9 SAMPLE AGRI LOAN DATA**

****

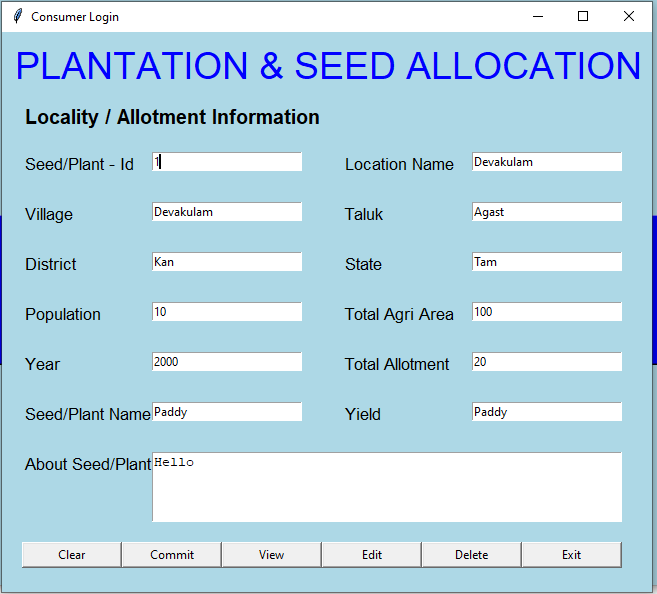
**Fig 11.9 Sample Agri Loan Data**

**11.10 SAMPLE FERTILIZER DATA**

****

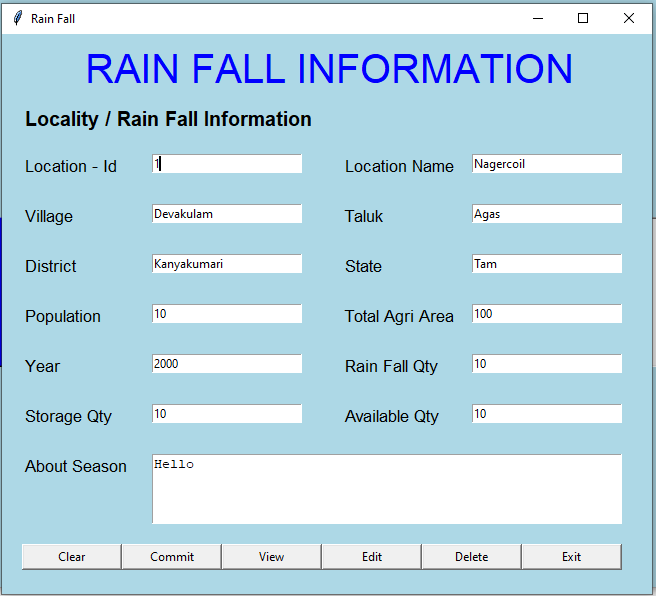
**Fig 11.10 Sample Fertilizer Data**

**11.11 PLANTATION SAMPLE DATA**

****

**Fig 11.11 Plantation Sample Data**

**11.12 RAIN FALL SAMPLE DATA**

****

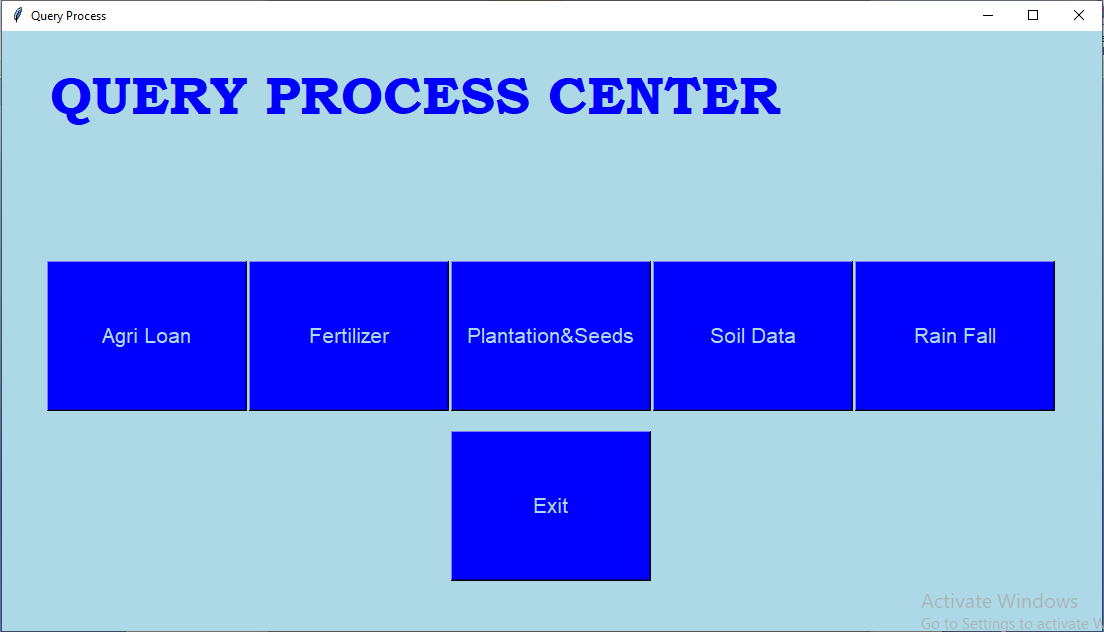
**Fig 11.12 Rain Fall Sample Data**

**11.13 CONSUMER LOGIN**

****

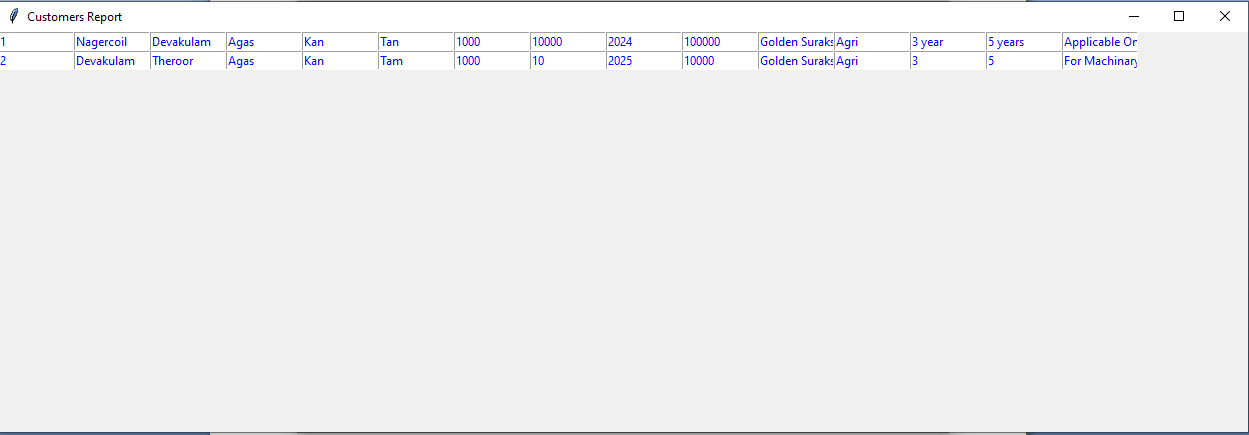
**Fig 11.13 Consumer Login**

**11.14 CONSUMER QUERY PROCESS CENTER**

****

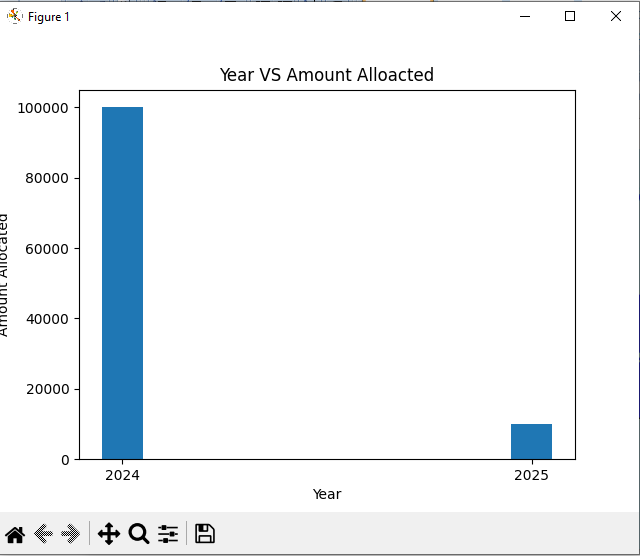
**Fig 11.14 Consumer Query Process Center**

**11.15 OVERALL ENTITY**

****

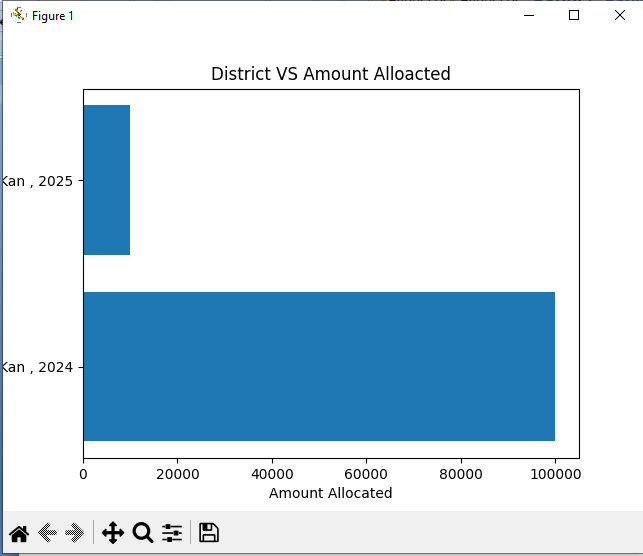
**Fig 11.15 Overall Entity**

**11.16 AMOUNT VS YEAR**

****

**Fig 11.16 Amount vs Year**

**11.17 DISTRICT VS AMOUNT**

****

**Fig 11.17 District Vs Amount**

**12. CONCLUSION**

The advent of digital technologies is revolutionizing agriculture in India, offering promising solutions to enhance efficiency, productivity, and sustainability. Notable advancements include precision agriculture, robotics, AI, IoT, blockchain, and VR/AR. Despite the compelling benefits, challenges such as technical and economic constraints, social and ethical issues, and policy inadequacies must be addressed. Future trends indicate the continued development of these technologies, which will profoundly shape global agricultural practices. Crucially, policy and regulatory adaptations will be required to foster this digital transformation. The evolution of India's agriculture, driven by these emerging technologies, holds immense potential, but realizing this future hinges on overcoming existing barriers and equipping farmers with the necessary skills and tools.

**13. FUTURE ENHANCEMENT**

Digital technology's dynamic nature means that it continually evolves, offering new opportunities for enhancement in various sectors, including agriculture.

1. **Potential Impact of Future Developments on Global Agricultural Practices**

As digital technologies become more prevalent, their impact on global agricultural practices will increase. Precision farming techniques will likely become more advanced, with a greater emphasis on sustainability and efficiency. The integration of different digital technologies will likely result in smart farming systems, where all aspects of farming are interconnected and optimized

1. **The Role of Policy and Regulatory Adaptations for Future Progress**

Adapting policies and regulations to support the growth and integration of digital technologies in agriculture will be crucial. Policies should facilitate access to digital technologies, particularly for small and marginalized farmers. Regulations should ensure data privacy and security while promoting transparency and interoperability between different digital platforms. Also Governments can play a significant role in promoting research and development in agricultural technologies and providing necessary infrastructure for their adoption.

**14. APPENDIX**

import sys

import os

import tkinter as tk

from tkinter import \*

def Registration():

os.system('python Registration.py')

def Login():

os.system('python Login.py')

root = tk.Tk()

w=1100

h=600

screen\_width = root.winfo\_screenwidth()

screen\_height = root.winfo\_screenheight()

x = (screen\_width/2) - (w/2)

y = (screen\_height/2) - (h/2)

root.geometry('%dx%d+%d+%d' % (w, h, x, y))

root.title("Admin Control")

root.config(bg='lightblue')

hlab1 = tk.Label(root, text ='ADMIN CONTROL' )

hlab1.place(x = 45, y = 30)

hlab1.config(bg='lightblue',fg='blue',font=('Bookman Old Style', 40,'bold'))

b1 = tk.Button(root, text='Registration', bg='blue', fg='lightblue',command=Registration)

b1.place(x=220, y=250, width=200, height=200)

b1.config(font=('Helvetica bold', 20))

b2 = tk.Button(root, text='Login', bg='blue', fg='lightblue',command=Login)

b2.place(x=440, y=250, width=200, height=200)

b2.config(font=('Helvetica bold', 20))

b3 = tk.Button(root, text='Back', bg='blue', fg='lightblue',command='exit')

b3.place(x=660, y=250, width=200, height=200)

b3.config(font=('Helvetica bold', 20))

root.mainloop()

import tkinter as tk

from tkinter import \*

import mysql.connector

import sys

import os

mydb=mysql.connector.connect(host="localhost",username="root",password="",database="agritech")

mycursor = mydb.cursor()

def Clear():

t1.delete("0","end")

t2.delete("0","end")

def Check():

d1=t1.get()

d2=t2.get()

val=(d1,d2)

sql="select \*from adminreg where lid=%s and psw=%s"

mycursor.execute(sql,val)

resultset=mycursor.fetchall()

for x in resultset:

s1=x[6]

s2=x[7]

if d1==s1 and d2==s2:

os.system("python DataProcess.py")

else:

print("invalid")

root = tk.Tk()

w=600

h=600

screen\_width = root.winfo\_screenwidth()

screen\_height = root.winfo\_screenheight()

x = (screen\_width/2) - (w/2)

y = (screen\_height/2) - (h/2)

root.geometry('%dx%d+%d+%d' % (w, h, x, y))

root.title("Faculty Login")

root.config(bg='lightblue')

hlab = tk.Label(root, text ="ADMIN LOGIN", )

hlab.place(x = 170, y = 10)

hlab.config(bg='lightblue',font=('Helvetica bold', 30))

l1 = tk.Label(root, text ="Login - Id" )

l1.place(x = 180, y = 130)

l1.config(bg='lightblue',font=('Arial', 16))

t1 = tk.Entry(root, width = 35)

t1.place(x = 180, y = 200, width = 200)

l2 = tk.Label(root, text ="Password")

l2.place(x = 180, y = 270)

l2.config(bg='lightblue',font=('Arial', 16))

t2 = tk.Entry(root, width = 35)

t2.place(x = 180, y = 340, width = 200)

b1 = tk.Button(root, text ="Clear", bg ='grey',command=Clear)

b1.place(x = 180, y = 410, width = 80)

b1.config(font=('Helvetica bold', 13))

b2 = tk.Button(root, text ="Login", bg ='grey',command=Check)

b2.place(x = 270, y = 410, width = 80)

b2.config(font=('Helvetica bold', 13))

b3 = tk.Button(root, text ="Quit", bg ='grey',command='exit')

b3.place(x = 360, y = 410, width = 80)

b3.config(font=('Helvetica bold', 13))

root.mainloop()

import tkinter as tk

import mysql.connector

mydb=mysql.connector.connect(host="localhost",username="root",password="",database="agritech")

mycursor=mydb.cursor()

def Clear():

t1.delete(0,'end')

t2.delete(0,'end')

t3.delete(0,'end')

t4.delete(0,'end')

t5.delete(0,'end')

t6.delete(0,'end')

t7.delete(0,'end')

t8.delete(0,'end')

t9.delete(0,'end')

t10.delete(0,'end')

t11.delete(0,'end')

t12.delete(0,'end')

t13.delete(1.0,'end')

def Commit():

d1=t1.get()

d2=t2.get()

d3=t3.get()

d4=t4.get()

d5=t5.get()

d6=t6.get()

d7=t7.get()

d8=t8.get()

d9=t9.get()

d10=t10.get()

d11=t11.get()

d12=t12.get()

d13=t13.get(1.0,'end')

val=(d1,d2,d3,d4,d5,d6,d7,d8,d9,d10,d11,d12,d13)

sql="insert into fertilizer values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"

mycursor.execute(sql,val)

mydb.commit()

def View():

lid=t1.get()

val=(lid,)

sql="select \*from fertilizer where lid=%s"

mycursor.execute(sql,val)

resultset=mycursor.fetchall()

for x in resultset:

t2.insert(0,x[1])

t3.insert(0,x[2])

t4.insert(0,x[3])

t5.insert(0,x[4])

t6.insert(0,x[5])

t7.insert(0,x[6])

t8.insert(0,x[7])

t9.insert(0,x[8])

t10.insert(0,x[9])

t11.insert(0,x[10])

t12.insert(0,x[11])

t13.insert(1.0,x[12])

def Delete():

lid=t1.get()

val=(lid,)

sql="delete from fertilizer where lid=%s"

mycursor.execute(sql,val)

mydb.commit()

root=tk.Tk()

w=800

h=600

screen\_width = root.winfo\_screenwidth()

screen\_height = root.winfo\_screenheight()

x = (screen\_width/2) - (w/2)

y = (screen\_height/2) - (h/2)

root.geometry('%dx%d+%d+%d' % (w, h, x, y))

root.title("Fertilizer - Information")

root.config(bg='lightblue')

hlab = tk.Label(root, text ="FERTILIZER ALLOTMENT")

hlab.place(x = 90, y = 10)

hlab.config(fg='blue',bg='lightblue',font=('Helvetica bold', 30))

w=650

h=560

screen\_width = root.winfo\_screenwidth()

screen\_height = root.winfo\_screenheight()

x = (screen\_width/2) - (w/2)

y = (screen\_height/2) - (h/2)

root.geometry('%dx%d+%d+%d' % (w, h, x, y))

root.title("Consumer Login")

root.config(bg='lightblue')

lab = tk.Label(root, text ="Locality / Allotment Information")

lab.place(x=20, y=70)

lab.config(fg='black',bg='lightblue',font=('Helvetica bold', 15,'bold'))

l1 = tk.Label(root, text ="Fertilizer - Id" )

l1.place(x=20, y =120)

l1.config(fg='black', bg='lightblue',font=('Arial', 13))

t1 = tk.Entry(root)

t1.place(x=150, y =120,width=150)

l2 = tk.Label(root, text ="Location Name" )

l2.place(x=340, y =120)

l2.config(fg='black', bg='lightblue',font=('Arial', 13))

t2 = tk.Entry(root)

t2.place(x=470, y =120,width=150)

l3 = tk.Label(root, text ="Village" )

l3.place(x=20, y =170)

l3.config(fg='black', bg='lightblue',font=('Arial', 13))

t3 = tk.Entry(root)

t3.place(x=150, y =170,width=150)

l4 = tk.Label(root, text ="Taluk" )

l4.place(x=340, y =170)

l4.config(fg='black', bg='lightblue',font=('Arial', 13))

t4 = tk.Entry(root)

t4.place(x=470, y =170,width=150)

l5 = tk.Label(root, text ="District" )

l5.place(x=20, y =220)

l5.config(fg='black', bg='lightblue',font=('Arial', 13))

t5 = tk.Entry(root)

t5.place(x=150, y =220,width=150)

l6 = tk.Label(root, text ="State" )

l6.place(x=340, y =220)

l6.config(fg='black', bg='lightblue',font=('Arial', 13))

t6 = tk.Entry(root)

t6.place(x=470, y =220,width=150)

l7 = tk.Label(root, text ="Population" )

l7.place(x=20, y =270)

l7.config(fg='black', bg='lightblue',font=('Arial', 13))

t7 = tk.Entry(root)

t7.place(x=150, y =270,width=150)

l8 = tk.Label(root, text ="Total Agri Area" )

l8.place(x=340, y =270)

l8.config(fg='black', bg='lightblue',font=('Arial', 13))

t8 = tk.Entry(root)

t8.place(x=470, y =270,width=150)

l9 = tk.Label(root, text ="Year" )

l9.place(x=20, y =320)

l9.config(fg='black', bg='lightblue',font=('Arial', 13))

48

t9 = tk.Entry(root)

t9.place(x=150, y =320,width=150)

l10 = tk.Label(root, text ="Total Allotment" )

l10.place(x=340, y =320)

l10.config(fg='black', bg='lightblue',font=('Arial', 13))

t10 = tk.Entry(root)

t10.place(x=470, y =320,width=150)

l11 = tk.Label(root, text ="Fertilizer Name" )

l11.place(x=20, y =370)

l11.config(fg='black', bg='lightblue',font=('Arial', 13))

t11 = tk.Entry(root)

t11.place(x=150, y =370,width=150)

l12=tk.Label(root, text ="Main Application" )

l12.place(x=340, y =370)

l12.config(fg='black', bg='lightblue',font=('Arial', 13))

t12=tk.Entry(root)

t12.place(x=470, y =370,width=150)

l13 = tk.Label(root, text ="About Fertilizer" )

l13.place(x=20, y =420)

l13.config(fg='black', bg='lightblue',font=('Arial', 13))

t13 = tk.Text(root)

t13.place(x=150, y =420,width=470,height=70)

b1=tk.Button(root,text="Clear",command=Clear)

b1.place(x=20,y=510,width=100)

b2=tk.Button(root,text="Commit",command=Commit)

b2.place(x=120,y=510,width=100)

b3=tk.Button(root,text="View",command=View)

b3.place(x=220,y=510,width=100)

b4=tk.Button(root,text="Edit",command=Edit)

49

b4.place(x=320,y=510,width=100)

b5=tk.Button(root,text="Delete",command=Delete)

b5.place(x=420,y=510,width=100)

b6=tk.Button(root,text="Exit",command=exit)

b6.place(x=520,y=510,width=100)

**15. REFERENCES**

1. Ampatzidis Y, Vougioukas S. Robotic systems and solutions for crop production and crop protection. Precis Agric. 2019;’19:1-4.
2. Aryal J, Farnworth CR, Khurana R, Ray D, Sapkota TB. Agricultural policymaking in India: The role of policy processes and institutions. In: State of Indian agriculture. Springer. 2019;95-108.
3. Bakker T, Asselt K, Bontsema J, Müller J, Straten G. Systematic design of an autonomous platform for robotic weeding. J Terramech. 2010;47(2):63-73.
4. Biswas S, Muthukumar M, Yoganand B. Adoption constraints and challenges of digital agriculture: A study among marginal and small farmers in India. Curr Sci. 2020;119(9):1458-64.
5. Blackmore S, Stout B, Wang M, Runov B. Robotic agriculture– The future of agricultural mechanisation? European Conference on Precision Agriculture; 2020.
6. Bronson K, Knezevic I. Big Data in food and agriculture. Big Data Soc. 2016;3(1).
7. Caro MP, Ali MS, Vecchio M, Giaffreda R. Blockchain-based traceability in Agri-Food supply chain management: A practical implementation. In: IoT Vertical and Topical Summit on Agriculture – Tuscany (IOT Tuscany). IEEE Publications. 2018; 2018:1-4.
8. Chakraborty M, Ghosh A, Joshi PK. Understanding the adoption of precision agriculture in India. Precis Agric. 2020;21:927-45.