Project

**Overview**

Our intended approach to this project is to build an application that helps for a local chicken breeder to move from paper based chicken based record system to Database system. I implemented both **Command-Line Interface (CLI)** and a **Graphical User Interface (GUI)** using Python’s **Tkinter** library. The application supports **CRUD operations** (Create, Read, Update, Delete) for managing chicken records efficiently.

|  |  |
| --- | --- |
| Proposed by: | Darshana D |
| Timeframe: | Completion by 23/07/2025  Presentation on 23/07/2025 |

Table of Contents

Contents

[Report Introduction 2](#_Toc204162412)

[Objectives and steps 2](#_Toc204162413)

[Tools and Technologies Used 2](#_Toc204162414)

[General Methodology 2](#_Toc204162415)

[Architecture Diagram 4](#_Toc204162416)

[Project Code (Command Line Interface) 4](#_Toc204162417)

[Project output (Command Line Interface) 7](#_Toc204162418)

[Project Code (Graphical User Interface) 8](#_Toc204162419)

[Project output (Graphical User Interface) 11](#_Toc204162420)

[Git & Github 13](#_Toc204162421)

[Overall conclusion 14](#_Toc204162422)

# Report Introduction

Paper based record keeping can be time consuming, error prone, inefficient and difficult to manage especially for management operation. This project aims to build an application to provides a user-friendly digital solution to replace traditional paper-based record keeping system.

I intend to build a Python based application using both Command line interface and a Graphical user Interface. This application will support for efficient CRUD operations (Create, Read, Update, Delete) on chicken name records, and will use a CSV file to store date.

This project not only simplifies day-to-day management but also introduces breeders to digital record keeping, improving accuracy and accessibility.

# Objectives and steps

# Tools and Technologies Used

* Python -Core Programming Language
  + Csv - Reading/writing/appending chicken records to .csv file
  + Tkinter-for GUI application
* Vs Code -Code Editor
* Git & Github – Version Control

### General Methodology

Planning -

* Understand the problem – Replace a paper based chicken record system into digital based record keeping system
* Client requirement - Must support Create, Read, Update, Delete (CRUD) using CLI and GUI
* Tech Stack Decision -Python, CSV, Git for version control.

Design –

* User Interface – CLI (Menu system) and GUI (tkinter)
* Data Storage - .csv(storage)
* **Architecture Diagram**: Modular functions for each action (Load, Append, Update, Delete).
* **Version Control Setup**: Initialized Git repo, wrote README, planned commits.

Development –

* Load and display chicken records.
* Add new record (with duplication check).
* Update/delete logic.
* File handling with csv
* GUI layer using tkinter.

Testing

Manual testing for:

* Edge cases (e.g. duplicate names).
* File not found errors.
* Input validation.
* Used print/debugging to validate changes.

Deployment

* CLI version is fully functional.
* GUI version launched via tkinter.
* Delivered through GitHub repository

# Architecture Diagram

User Input

Menu System (Create / Read / Update / Delete) (CRUD functions)

In-Memory Data Structure    |

| chickens = [name1, name2...]

CSV Storage   chicken.csv

Read (load)  ,Write / Append

# Project Code (Command Line Interface)

import csv

path="chicken.csv"

def load():

    try:

        chicken=[]

        with open(path,'r',newline='') as file:

            reader=csv.reader(file)

            column\_name=next(reader)

            for i in reader:

                chicken.append(i[0])

        return chicken

    except FileNotFoundError:

        print("File not found")

        return []

    except Exception as e:

        print("An error occurred:", e)

        return []

def append(name):

    try:

        with open(path,'a',newline='') as file:

            writer=csv.writer(file)

            writer.writerow([name])

            print("Chicken write to the file successfully!")

    except PermissionError:

        print("Still permission denied with simple write.")

    except Exception as e:

        print("An error occurred:", e)

def list(chickens):

    for i in chickens:

        print(i)

def create(chickens):

    print(chickens)

    name=input("Enter the name of the chicken: ")

    if name in chickens:

        print("Chicken already exists!")

        return

    else:

        chickens.append(name)

        append(name)

        print("Chicken created successfully!")

def update(chickens):

    print(chickens)

    name=input("Enter the name of the chicken to updated")

    if name not in chickens:

        print("Chicken not found")

    else:

        index=chickens.index(name)

        name=input("Enter the new name of the chicken: ")

        chickens[index]=name

        try:

            with open(path, 'w', newline='') as file:

                writer = csv.writer(file)

                writer.writerow(["name"])  # re-write header

                for name in chickens:

                    writer.writerow([name])

        except Exception as e:

            print("An error occurred while updating:", e)

        print("Chicken updated successfully!")

def delete(chickens):

    print(chickens)

    name=input("Enter the name of the chicken to deleted")

    if name not in chickens:

        print("Chicken not found")

    else:

        chickens.remove(name)

        try:

            with open(path, 'w', newline='') as file:

                writer = csv.writer(file)

                writer.writerow(["name"])

                for i in chickens:

                    writer.writerow([i])

        except Exception as e:

            print("An error occurred while updating:", e)

        print("Chicken  successfully!")

def main():

    chickens=load()

    print(chickens)

    print("Menu -")

    print("0 - Exit App")

    print("1 - Print List of Chicken Records")

    print("2 - Create New Chicken Record")

    print("3 - Update Existing Chicken Record")

    print("4 - Delete a Chicken Record")

    while True:

        choice = input("Choose an option: ")

        if choice == '0':

            print("Exiting...")

            break

        elif choice == '1':

            list(chickens)

        elif choice == '2':

            create(chickens)

        elif choice == '3':

            update(chickens)

        elif choice == '4':

            delete(chickens)

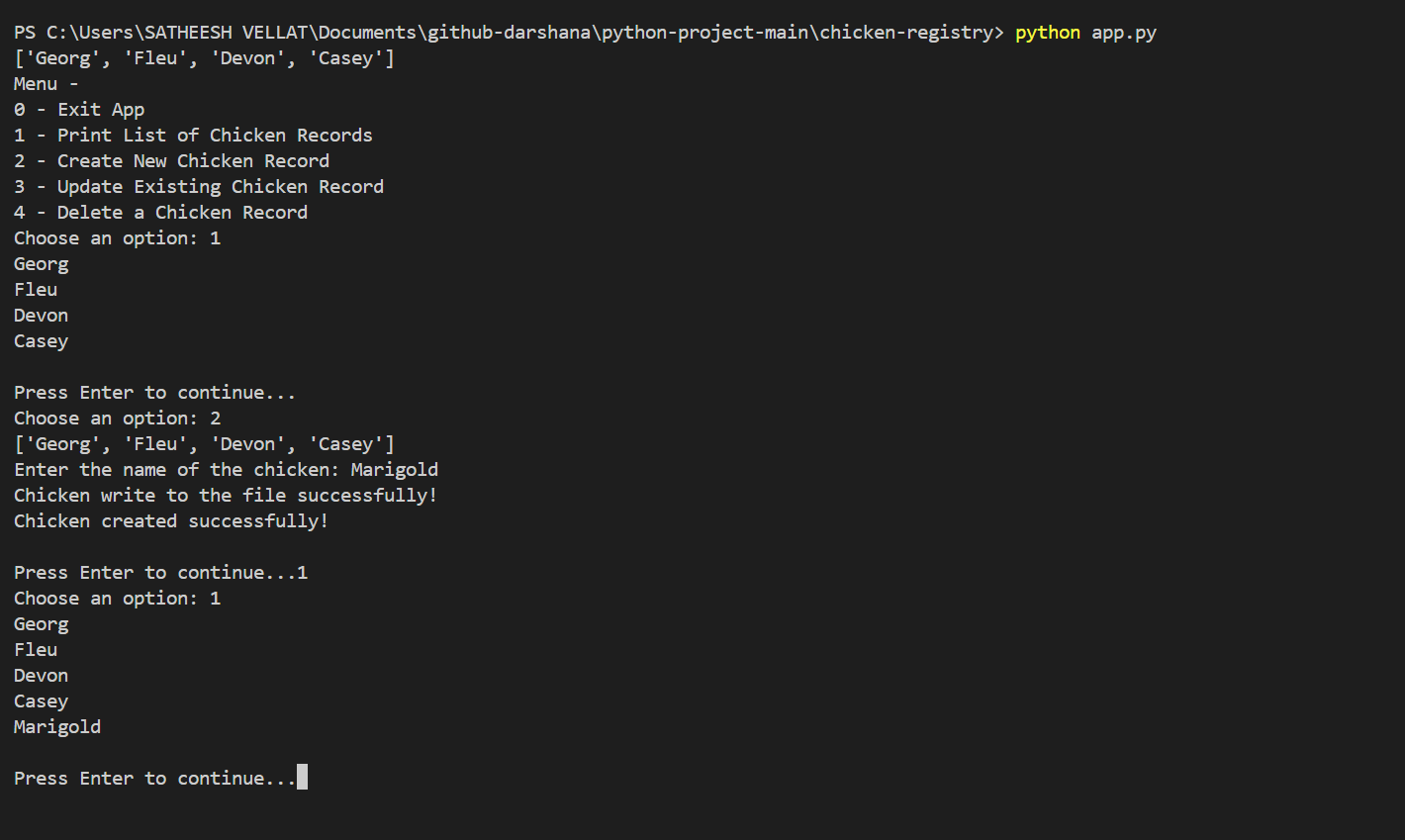
        else:

            print("Invalid choice.")

        input("\nPress Enter to continue...")

main()

# Project output (Command Line Interface)





# Project Code (Graphical User Interface)

import csv

import tkinter as tk

from tkinter import messagebox, simpledialog

from PIL import Image, ImageTk

path = "chicken.csv"

def load():

    try:

        chicken = []

        with open(path, 'r', newline='') as file:

            reader = csv.reader(file)

            column\_name = next(reader)

            for i in reader:

                chicken.append(i[0])

        return chicken

    except FileNotFoundError:

        print("File not found")

        return []

    except Exception as e:

        print("An error occurred:", e)

        return []

def append(name):

    try:

        with open(path, 'a', newline='') as file:

            writer = csv.writer(file)

            writer.writerow([name])

            print("Chicken written to the file successfully!")

    except PermissionError:

        print("Permission denied when writing to the file.")

    except Exception as e:

        print("An error occurred:", e)

def view\_chickens():

    chickens = load()

    if not chickens:

        messagebox.showinfo("Chickens", "No chickens found.")

        return

    info = "\n".join(chickens)

    messagebox.showinfo("All Chickens", info)

def add\_chicken():

    name = simpledialog.askstring("Input", "Enter chicken name:")

    if name:

        append(name)

        messagebox.showinfo("Success", f"{name} added successfully!")

def update\_chicken():

    chicken=load()

    name = simpledialog.askstring("Input", "Enter chicken name to be updated:")

    if name in chicken:

        index=chicken.index(name)

        up\_name=simpledialog.askstring("Input", "Enter new chicken name to be updated:")

        chicken[index]=up\_name

        try:

            with open(path, 'w', newline='') as file:

                writer = csv.writer(file)

                writer.writerow(["name"])

                for i in chicken:

                    writer.writerow([i])

        except Exception as e:

            print("An error occurred while updating:", e)

        messagebox.showinfo("Success", f"{up\_name} updated successfully!")

    else:

        messagebox.showinfo("Not Found", f"'{name}' was not found in the list.")

def delete\_chicken():

    chicken=load()

    name = simpledialog.askstring("Input", "Enter chicken name to be deleted:")

    if name in chicken:

        chicken.remove(name)

        try:

            with open(path, 'w', newline='') as file:

                writer = csv.writer(file)

                writer.writerow(["name"])

                for i in chicken:

                    writer.writerow([i])

        except Exception as e:

            print("An error occurred while updating:", e)

        messagebox.showinfo("Success", f"{name} deleted successfully!")

    else:

        messagebox.showinfo("Not Found", f"'{name}' was not found in the list.")

def main():

    win = tk.Tk()

    win.title("Chicken Registry App")

    win.geometry('400x200')

    win.resizable(False,False)

     # Load the image

    image = Image.open("chicken.jpeg")

    image = image.resize((150, 150))  # Resize the image to fit the label

    photo = ImageTk.PhotoImage(image)

    label = tk.Label(win, image=photo)

    label.pack(side=tk.LEFT)

    btn\_view = tk.Button(win, text="View Chicken Names", fg="blue", command=view\_chickens)

    btn\_view.pack(pady=10)

    btn\_add = tk.Button(win, text="Add Chicken Name", fg="green", command=add\_chicken)

    btn\_add.pack(pady=10)

    btn\_update = tk.Button(win, text="Update Chicken Name", fg="green", command=update\_chicken)

    btn\_update.pack(pady=10)

    btn\_delete = tk.Button(win, text="Delete Chicken Name", fg="green", command=delete\_chicken)

    btn\_delete.pack(pady=10)

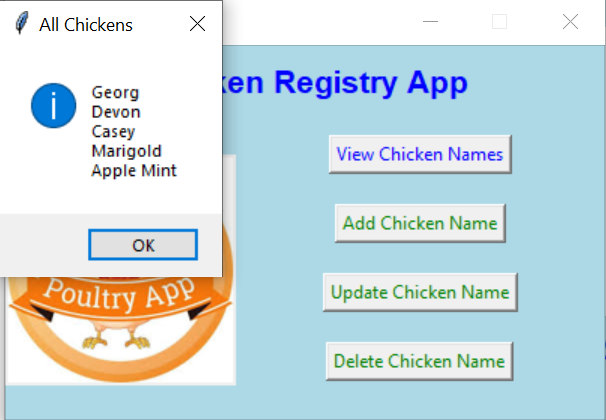
    win.mainloop()

main()

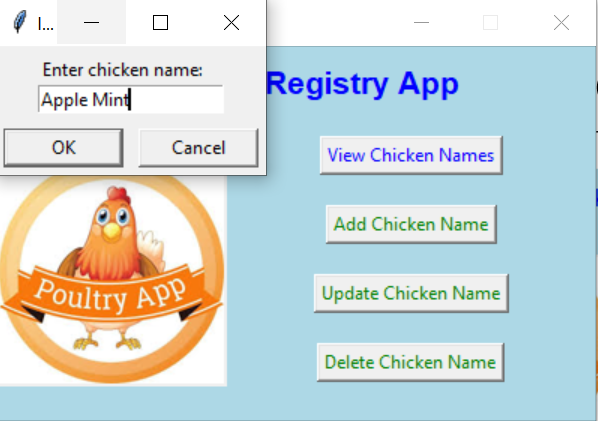
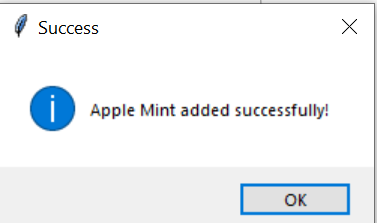
# Project output (Graphical User Interface)



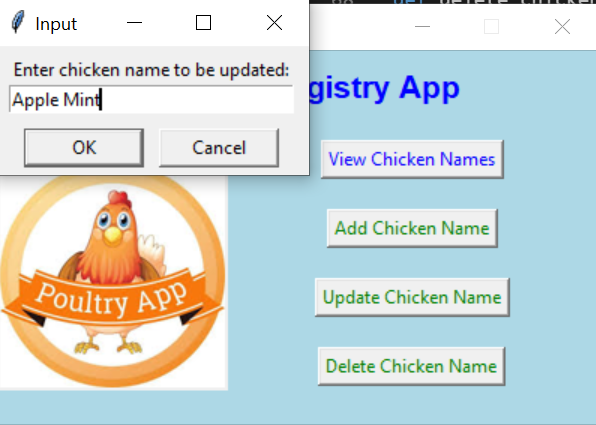
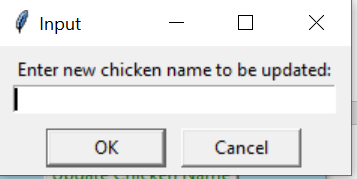
Printing the names:

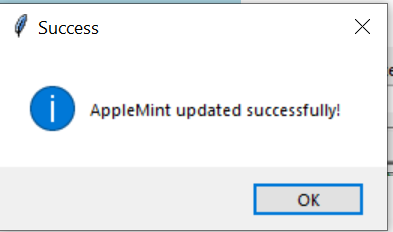


Adding Records

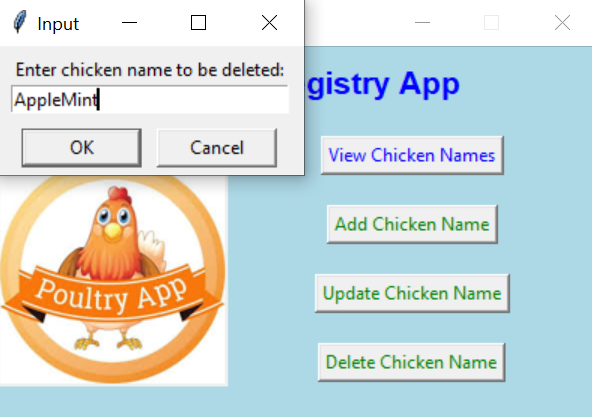
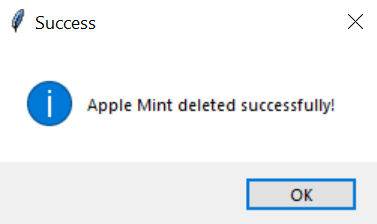
 

Updating the record



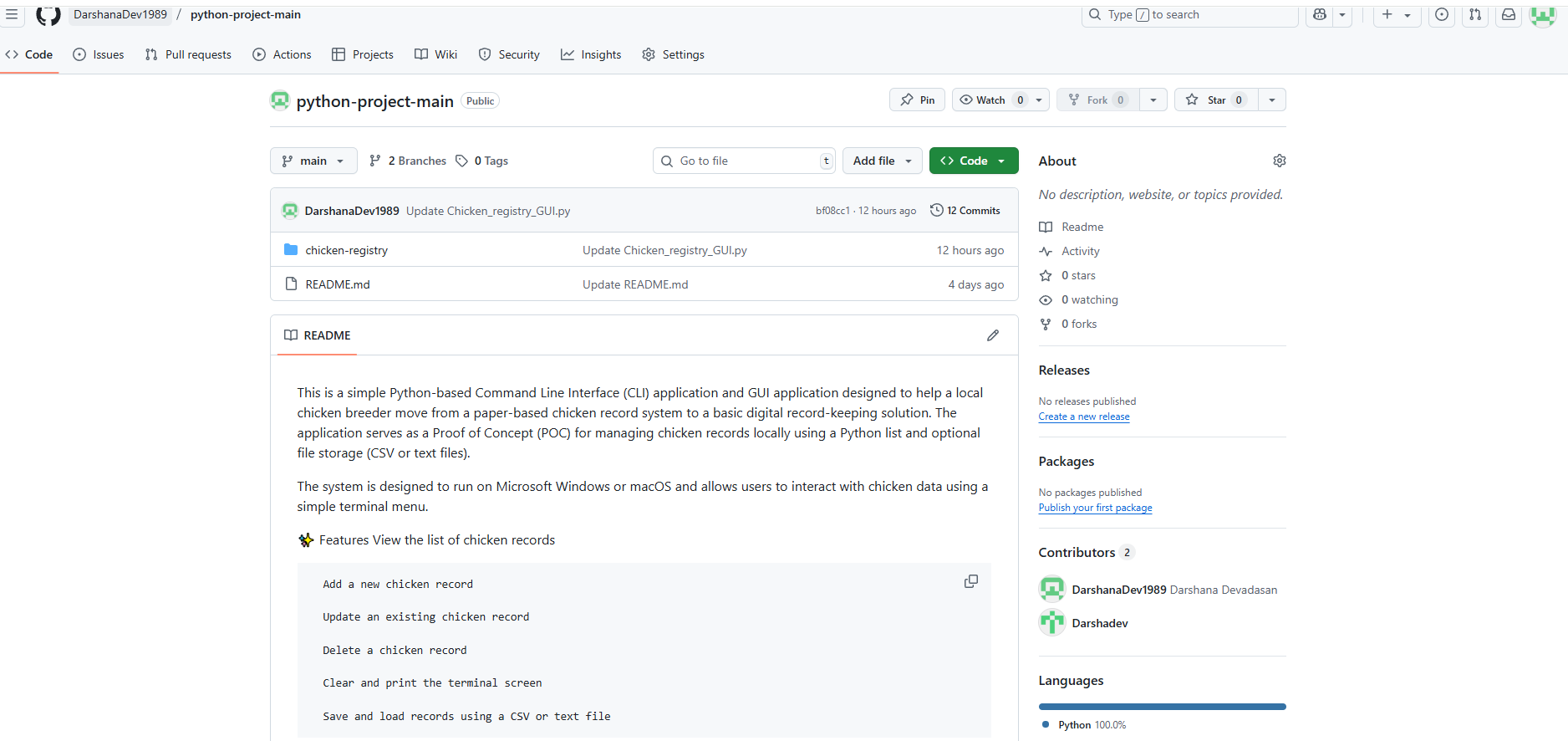
Deleting the record

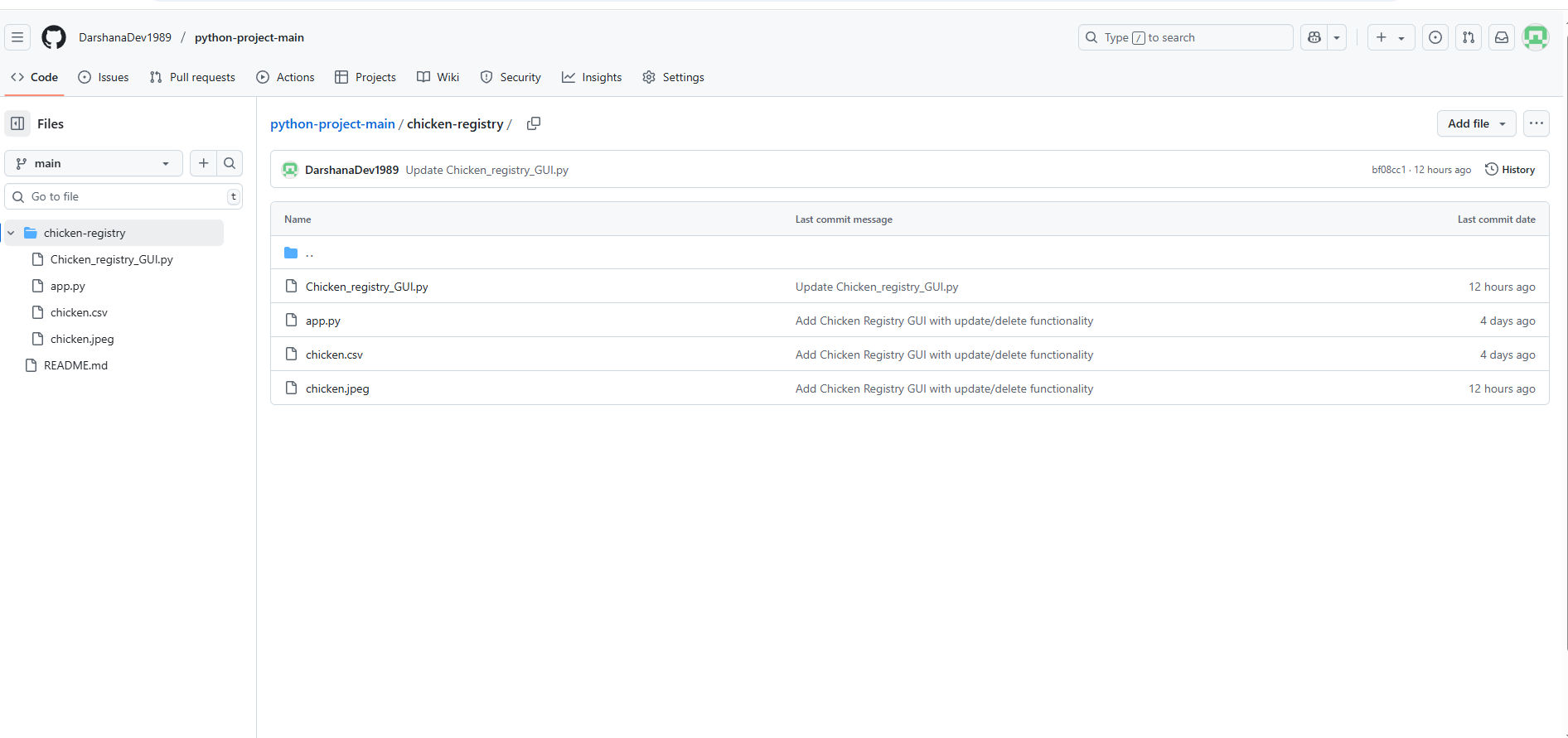
 

# Git & Github

Git and **Github** were used to manage and track the development of the Chicken Record CLI application. Git served as the **local version control system**, allowing for systematic tracking of changes, easy rollback of mistakes, and clear documentation of progress through meaningful commit messages. **GitHub**, a cloud-based platform for hosting Git repositories, was used to store the project remotely, providing a centralized and accessible version of the codebase. This allowed for better organization, a secure backup of the code, and a clear development history that could be reviewed by the client. The README.md file on GitHub was also used as live documentation, outlining how the application works, the tools used, and user instructions.

https://github.com/DarshanaDev1989/python-project-main.git





# Overall conclusion

The Chicken Record CLI and GUI application can transition from paper based to user friendly digital solution. The project met all the objectives including functional command line menu and Graphical user menu, persistent memory data handling using CSV and version control for documentation using Github. Using tools like **Python**, **VS Code**, **Git**, and **GitHub** enabled a smooth and structured development process, encouraging best practices in version control, code management, and client communication. Overall, the project proves that even basic technology solutions can significantly improve operational efficiency, data reliability and efficient for owners.

To expand the functionality of the Chicken Record CLI application and better support the real-world needs of a chicken breeder, several enhancements can be made by adding **more data columns** to each chicken record and also extends the functionality from CSV storage to relational database for enhancing security.