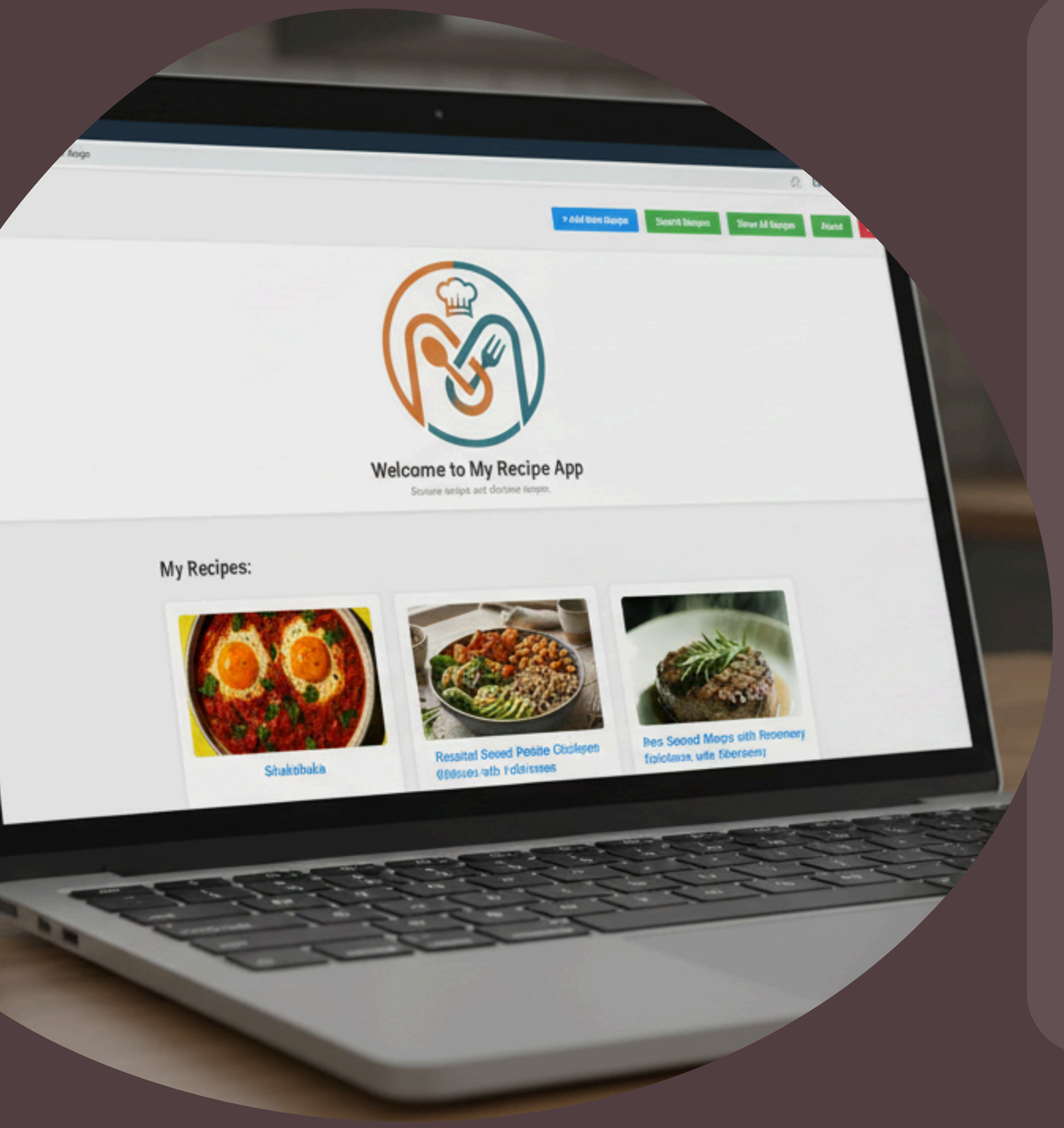
Case Study

My Recipe App

Search & Data Visualization Web Application (Django)

Mark Francis



Overview

The Recipe App is a full-stack Django web application that allows authenticated users to search for recipes, filter results by cooking time, and visualize recipe data through interactive charts.
The application implements advanced search functionality, dynamic filtering, and multiple data visualizations powered by Pandas and Matplotlib. Users can perform partial name searches, combine search criteria, view detailed recipe information, and analyze cooking time distributions through bar, pie, and line charts.
The project was developed as part of the CareerFoundry Python for Web Developers specialization (Achievement 2, Exercise 2.7) to demonstrate backend development, data analysis, testing practices, and clean architectural design.

https://my-recipe-app-6105574f7d6e.herokuapp.com/

https://github.com/CreativeMarkus/Achievement.2

Purpose

The purpose of this project was to enhance a Django application by implementing search functionality and integrating data visualization while maintaining best practices in architecture, testing, and documentation.

Objective

The primary objective of this project was to:

- Implement advanced search with partial (wildcard) matching
- Filter recipes by cooking time
- Combine multiple search criteria
- Display results in a structured table with clickable detail links
- Generate dynamic visualizations (bar, pie, line charts)
- Write comprehensive automated tests
- Follow Django best practices and separation of concerns
- Document execution flow and deliverables professionally

Duration & Credits

Role: Sole Developer
Program: CareerFoundry – Python for Web Developers
Exercise: 2.7 Search & Visualization
The project was completed over several development sessions, with additional time dedicated to testing, documentation, and refining architecture.

Methodologies & Tools

- Django 6.0.2
- Python 3.14
- SQLite3
- Pandas 3.0.0
- Matplotlib
- Django TestCase
- HTML Templates
- Git & GitHub

Server-Side Integration

The application follows Django's Model–View–Template (MVT) architecture.

Models

- Recipe model with cooking time categorization

tilities

- Chart generation functions using Pandas and Matplotlib

Forms

- Custom search form with validation

Templates

- Dynamic HTML rendering
- Embedded Base64 charts

Data Visualization

- Bar Chart: Displays number of recipes per cooking time category
- Pie Chart: Shows percentage distribution of cooking times
- Line Chart: Displays cumulative recipe count
- Users can select which chart type to display. Charts are generated using Matplotlib and processed with Pandas before being embedded directly into the HTML template as Base64 images.

Search Functionality

The application allows users to:

- Search recipes by name (partial/wildcard matching supported)
- Filter recipes by cooking time
- Combine name search and cooking time filter
- Display all recipes without filters
- Click on results to view detailed recipe pages

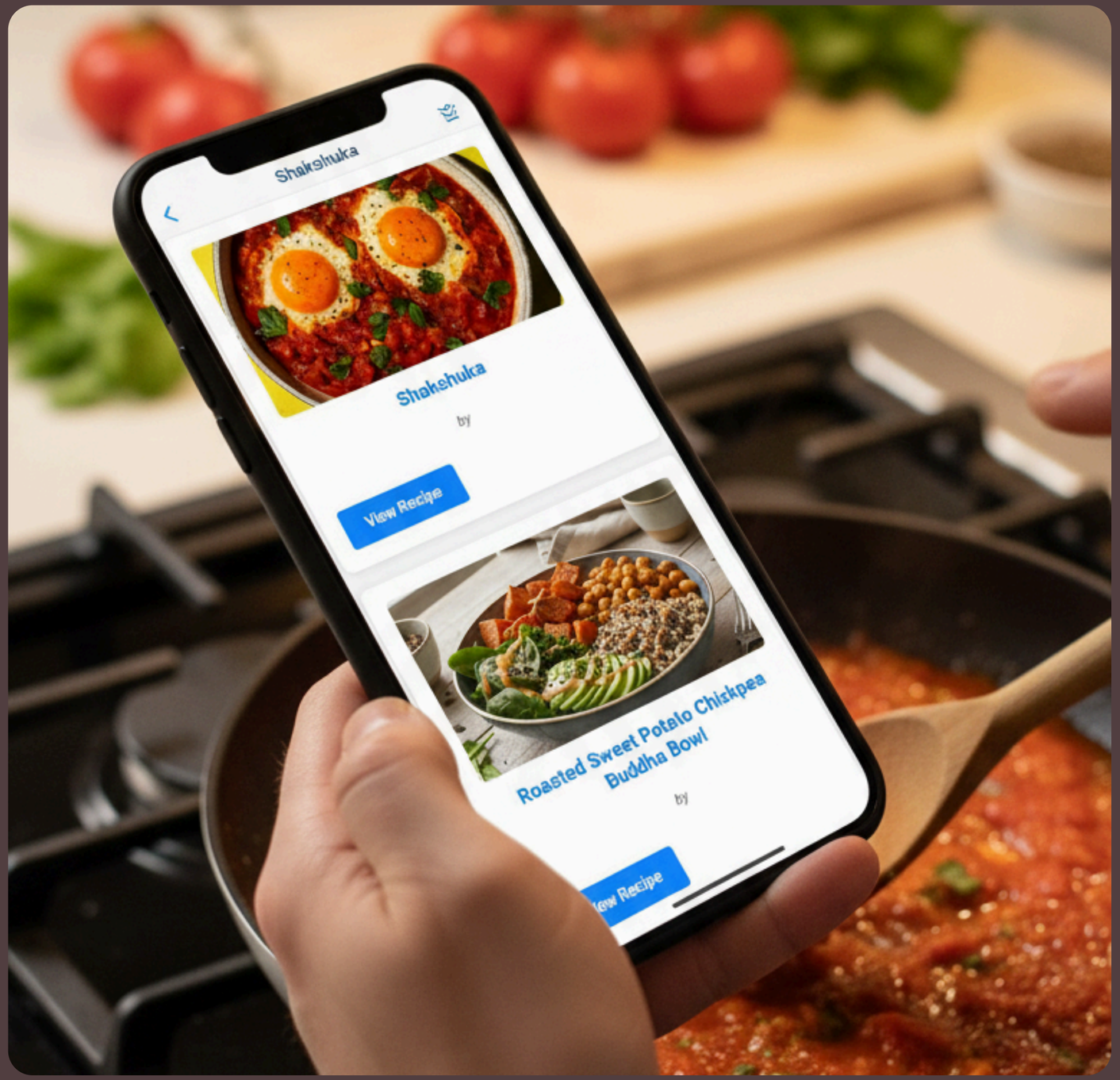
The filtering is handled using Django's QuerySet API to ensure efficient database querying.

Progressive Web App Features

- RecipeModelTest
- RecipeSearchFormTest
- RecipeSearchViewTest
- RecipeListViewTest
- RecipeDetailViewTest

This Ensured

- Correct search filtering
- Form validation accuracy
- View response integrity
- Model behavior verification



Retrospective

What Went Well

The separation between search logic and visualization utilities made the architecture clean and maintainable. Integrating Pandas for data processing significantly simplified chart data preparation. Writing comprehensive tests improved reliability and confidence in feature behavior.

Challenges

Generating Base64-encoded charts required careful handling of Matplotlib output streams. Combining multiple search filters while maintaining clean QuerySet logic required thoughtful structuring. Ensuring 100% test pass rate required iterative debugging and validation.

Future Steps

- Add pagination to search results
- Implement filtering by ingredients or difficulty level
- Export search results to PDF or CSV
- Add recipe rating system
- Introduce recommendation logic

Final Thoughts

This project strengthened my understanding of Django’s architecture, QuerySet filtering, automated testing, and backend-driven data visualization. Implementing search, filtering, and analytics within a structured framework demonstrated how backend systems can provide meaningful insights beyond simple CRUD functionality. The Recipe App reflects strong backend development practices, structured documentation, and production-ready coding standards.