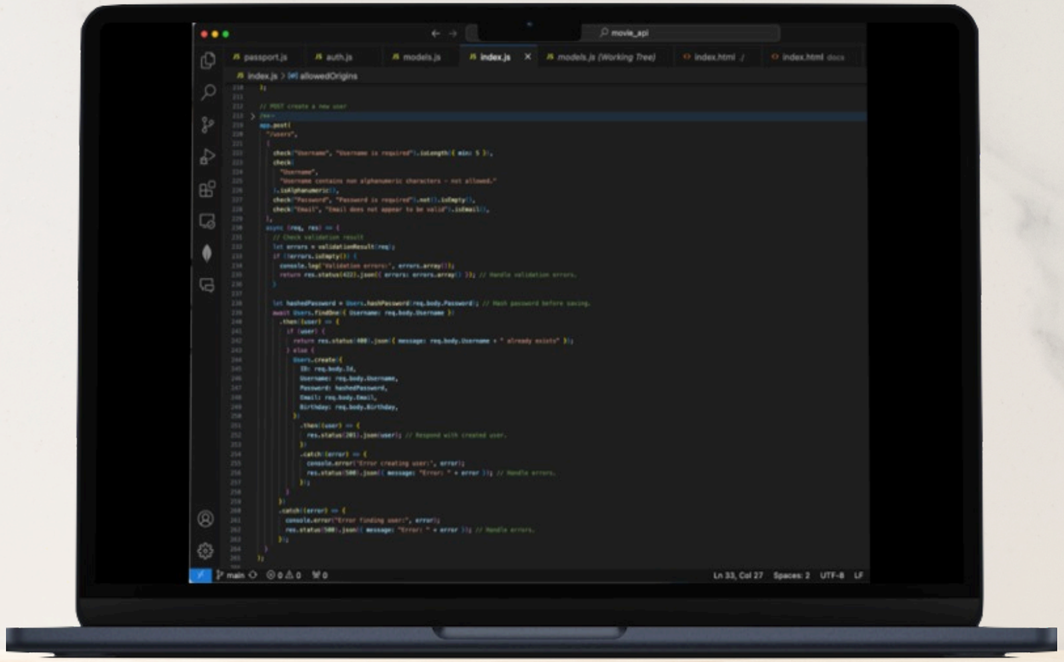

BiteBase-Case Study

A Python-based Django Recipe Management Web Application

Developer: Evan Danowitz



Try out the BiteBase application here and visit my portfolio here for more.

Overview

BiteBase is a Django-powered web app where users can **create**, **edit**, and **manage recipes**. They can **save details** like name, cooking time, ingredients, and an optional image. The app automatically calculates difficulty based on input values.

It also includes **user authentication** (signup, login, logout), a **custom search form** (filter by name or ingredient), and dynamic **data visualization** to track recipe trends. Each user has a **profile page** with basic account details and the option to **delete their account**.

Purpose & Objective

The purpose of creating BiteBase was to **showcase my Django and frontend skills** while creating a fully-functional and user-friendly application. While the goal wasn't to necessarily solve a real-world problem, it **demonstrates my ability to structure, develop, and style a full-stack web application** that follows best practices and industry standards.

Project Timeline & Development Role

📅. **Duration:** The development of BiteBase took **two months** to develop.

Before transitioning to Django, I previously developed a command-line version of the BiteBase application. The **transition to Django allowed me to expand its functionality** and implement a complete web-based interface.

I served as the **lead (and sole) full-stack developer for every aspect of BiteBase**. Throughout the project, I was responsible for planning, designing, and implementing both the backend and frontend components. To progress through each stage, I required approval from my dedicated course mentor, who provided feedback but did not directly guide the development process.

Credits

Lead Developer: Evan Danowitz | **Mentor:** Alfredo Salazar Vélez

---

Tools & Technologies

BiteBase was built using a combination of modern web development tools and technologies, including:

- 💻 **Backend:** Django web framework, Python
- 🎨 **Frontend:** HTML5, CSS3, Bootstrap, JavaScript
- 📊 **Database:** SQLite (development), PostgreSQL (production)
- 🚀 **Hosting & Deployment:** Heroku
- 🔗 **Version Control:** Git, GitHub

Core Features and Functionality

Recipe Management

- Users can **create**, **edit**, **delete and store recipes** in a database
- Recipes contain essential details: name, cooking time, ingredients, description, and an optional image
- The application automatically calculates difficulty level based on cooking time and number of ingredients

```
def recipe_list
```

```
def create_recipe_view
```

```
def edit_recipe_view
```

```
def delete_recipe_view
```

Search & Dynamic Data Visualization

- **Search recipes** by name, ingredient, or difficulty level
- Toggleable **search form** for improved user experience
- Three **interactive charts** for data analysis:
  - *Bar* Chart: Recipes by difficulty level
  - *Pie* Chart: Cooking time distribution
  - *Line* Chart: Most common ingredients
- **Charts dynamically update** based on search filters
- **Error handling** - Displays messages when no search results are found

User Authentication & Account Management

- Secure **user authentication** - signup, login, logout
- Custom **profile page** displaying username, email, and first name
- Ability to **delete account**, eliminating all user data

```
def signup_view
```

```
def login_view
```

```
def logout_view
```

Deployment & Performance Considerations

- **Hosted on Heroku** for live accessibility
- **Cloudinary integration** for image uploads in production
- **Responsive design** for various screen sizes and devices

```
% git push heroku main
```

Challenges Faced & Lessons Learned

Version Control & Repository Structure

Managing code between multiple repositories was initially confusing, especially when merging changes and pushing updates to the final recipe-app repository. Through practice, I became more comfortable with Git workflows.

Understanding Virtual Environments

Initially, I struggled with when to activate, use, and manage virtual environments Repeated practice helped solidify this.

Test-Writing Strategy

Determining what to test and how to structure test coverage was overwhelming. A lot that came from trying to ensure that I don't miss anything and cover all bases. Breaking tests into models, views, forms, and templates made it more manageable.

Handling Deleted Recipe Navigation Issues

If a user deleted a recipe and clicked back in the browser, they would see a cached version that no longer existed, causing the app to break. The solution was to redirect users to the recipes list upon successful deletion to prevent navigation errors.

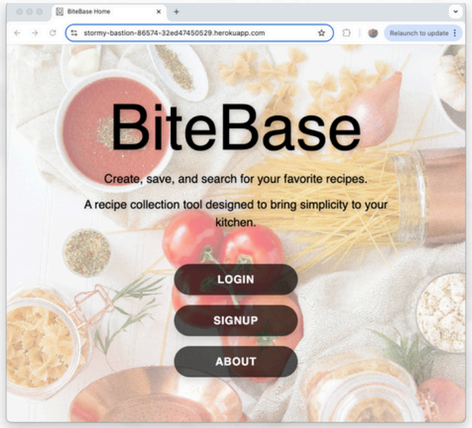
Image Uploads in Deployment

The live Heroku version of the application could not handle image uploads like the local app could. After research and testing, I implemented Cloudinary as an image hosting solution for BiteBase.

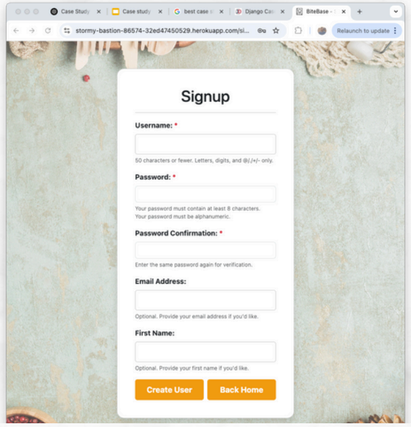
Final Deployment Troubleshooting

Some Heroku deployment guides were outdated, requiring additional research to troubleshoot missing steps.

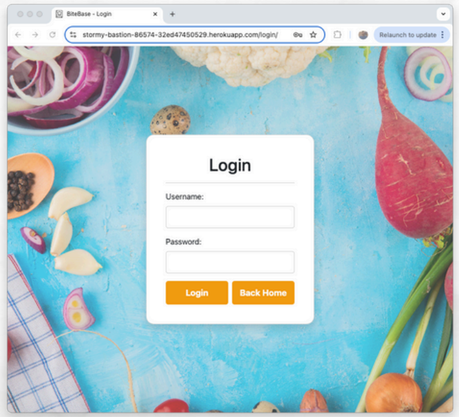
Page Views



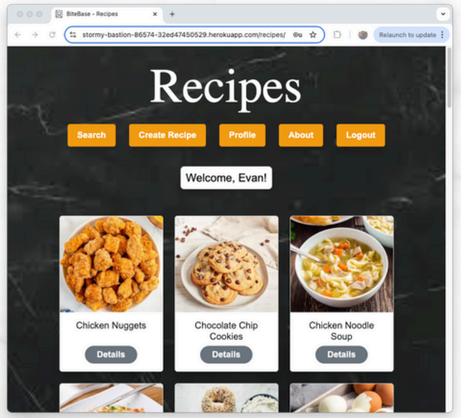
Home/Welcome



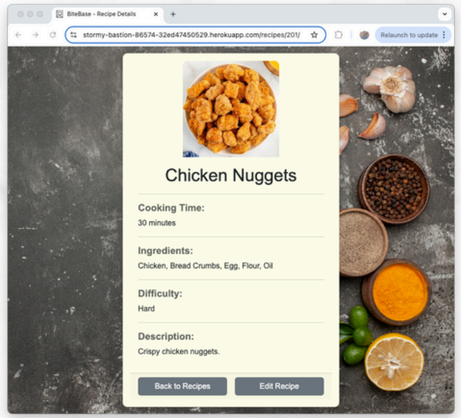
Sign Up



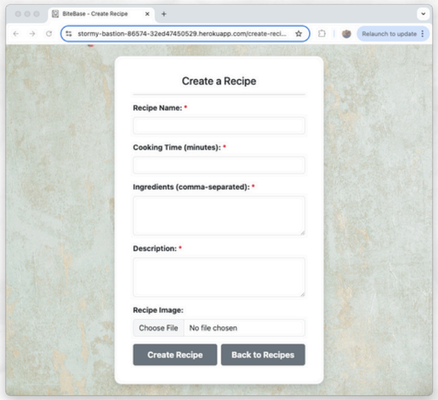
Log In



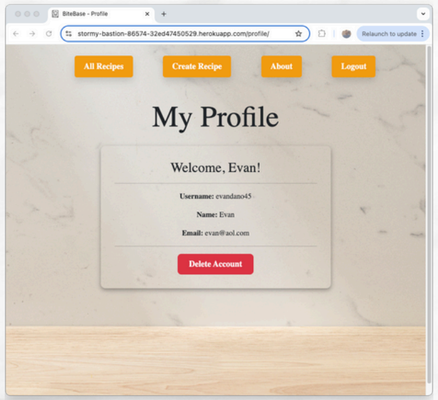
All Recipes



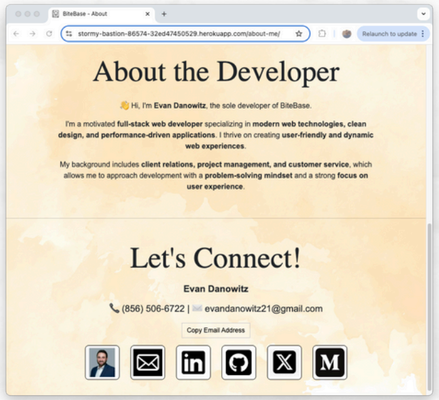
Recipe Details



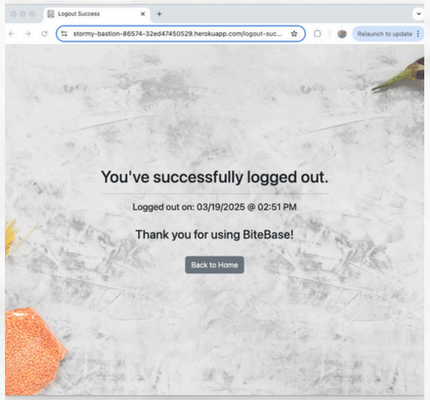
Create Recipe



Profile



About



Logout

Main Takeaway → Django's Power & Automation

One of my biggest takeaways was how **Django simplifies complex backend functionality**, truly allowing for developers to focus more on application logic and what matters most–the users.

Future Application Upgrades & Next Steps

While BiteBase is fully functional in its current state, I have several **ideas for future improvements:**

- **User Account Management:** Allow users to edit/update their username, password, and other personal details.
- **Favorite Recipes:** Enable users to add and remove recipes from a favorites list, accessible from their profile.
- **Styling Refinements:** Reduce the default Bootstrap aesthetic by incorporating more custom styling.

Final Thoughts

Building BiteBase was **a valuable learning experience**, reinforcing my knowledge of Django, database management, user authentication, and frontend development.

This project not only **showcases my technical skills** but also **highlights my problem-solving abilities and adaptability** in overcoming challenges across all aspects of the development process.