Planning

DreamsOnWheels

FULL STACK FRAMEWORKS WITH DJANGO MILESTONE PROJECT

Jussi Nousiainen
Student @

Code Institute

Full Stack Web Developer Program

Full Stack Frameworks with Django
Milestone Project 4.

Table of Contents

Project assignment

Project purpose:

In this project, you'll build a full-stack site based around business logic used to control a centrally-owned dataset. You will set up an authentication mechanism and provide paid access to the site's data and/or other activities based on the dataset, such as the purchase of a product/service.

Value provided:

1. By authenticating on the site and paying for some of its services, users can advance their own goals. Before authenticating, the site makes it clear how those goals would be furthered by the site.
2. The site owner is able to make money by providing this set of services to the users. There is no way for a regular user to bypass the site's mechanisms and derive all of the value available to paid users without paying.

Project Requirements

Main Technologies

HTML, CSS, JavaScript, Python + Django
Relational database (recommending MySQL or Postgres)
Stripe payments
Additional libraries and APIs

Mandatory Requirements

A project violating any of these requirements will FAIL

1. **Django Full Stack Project:** Build a Django project backend by a relational database to create a website that allows users to store and manipulate data records about a particular domain.

2. **Multiple Apps:** The project must be a brand new Django project, composed of multiple apps (an app for each potentially reusable component in your project).

3. **Data Modeling:** Put some effort into designing a relational database schema well-suited for your domain. Make sure to put some thought into the relationships between entities. Create at least 2 custom django models beyond the examples shown on the course

4. **User Authentication:** The project should include an authentication mechanism, allowing a user to register and log in, and there should be a good reason as to why the users would need to do so. e.g., a user would have to register to persist their shopping cart between sessions (otherwise it would be lost).

5. **User Interaction:** Include at least one form with validation that will allow users to create and edit models in the backend (in addition to the authentication mechanism).

6. **Use of Stripe:** At least one of your Django apps should contain some e-commerce functionality using Stripe. This may be a shopping cart checkout, subscription-based payments or single payments, donations, etc. After paying successfully, the user would then gain access to additional functionality/content on the site. Note that for this project you should use Stripe's test functionality, rather than actual live payments.

7. **Structure and Navigation:** Incorporate a main navigation menu and structured layout (you might want to use Bootstrap to accomplish this).

8. **Use of JavaScript:** The frontend should contain some JavaScript logic you have written to enhance the user experience.

9. **Documentation:** Write a README.md file for your project that explains what the project does and the value that it provides to its users.

10. **Version Control:** Use Git & GitHub for version control.

11. **Attribution:** Maintain clear separation between code written by you and code from external sources (e.g. libraries or tutorials). Attribute any code from external sources to its source via comments above the code and (for larger dependencies) in the README.

12. **Deployment:** Deploy the final version of your code to a hosting platform such as Heroku.

13. **Security:** Make sure to not include any passwords or secret keys in the project repository. Make sure to turn off the Django DEBUG mode, which could expose secrets.

Project Ideas

For the project I will combine two suggestions provided for the project:

Project Idea 0

Bring your own idea(s) to life, based on providing value to users to address a specific real or imagined need.

And a modified.

Project Idea 2

Build an e-commerce place to sell ~~historical artifacts~~ collectable items

External user’s goal:
Find, learn about and acquire collectable items they are interested in

Site owner's goal:
Earn money on selling collectables (the site owner is the seller) and share the passion with other enthusiasts.

Potential features to include:

- Create an online store focused on selling exclusive collectable items. In this case model cars from real dream cars.

- Allow users to search for artifacts based on various fields.
- Allow users to see the price, image and other basic details about a collectable.
- Users would be able to learn about the original dream car and its history.
- Allow users to vote for a model, they would like to see promoted. Users have to be registered for this.

Advanced potential feature (nice-to-have)

- Allow registered users to write comments to the blogs about dream cars and models.
- Expand the search functionality to allow users to sort results based on price and other parameters in both ascending and descending order.
- Include pagination and/or other dynamic display actions using javascript.
- Use javascript polling to update the page whenever there's a new vote.

Assessment Criteria

Your Full-stack Django project will be assessed based on the following criteria:

Usability and Visual Impact:

- Project Purpose
- UX design
- Suitability for purpose
- Navigation
- Ease of use
- Information Architecture
- Defensive Design

Layout and Visual Impact:

- Responsive Design
- Image Presentation

- Colour scheme and typography

Code Quality:

- Appropriate use of HTML
- Appropriate use of CSS
- Appropriate use of JavaScript
- Appropriate use of Python
- Appropriate use of the template language
- Appropriate use of Django

Application Features:

- App logic
- Cross-app logic
- E-commerce
- Authentication and Security

Software Development practices:

- Directory Structure and File Naming
- Version control
- Testing implementation
- Testing write-up
- Readme file
- Comments
- Data store integration
- Deployment implementation
- Deployment write-up

Task List

	Task Description	Aim	
	Create application	Create an ecommerce site	
	Design app structure	Think what you want the user to be able to achieve (Public users and registered users have different access to the application features and information. Registered users	

		are able to add to the database, edit and delete information. Registered users are able to make purchases.)	
	Write planning	Create a planning document to clarify what the project wants to achieve and how. Plan the tasks and steps. Scope the planning realistically to your level of skills and time available. Set a hierarchy of priorities.	
	Schedule the appointments with the mentor.	Plan schedule, set goalposts, and time the appointments appropriately.	
	Design database schema	Think of the relationships between the features and functions. Use foreign keys between databases to connect them where appropriate.	
	Think of testing strategy and write up	Create a document with clear steps to follow in different phases of the project	
	Allow users to Register, Login, Create Profiles, Add Votes and Comments, Add Sales Items to the Shopping Cart, Edit Shopping Cart and Make Purchases	Create back end logic – Models Create front end logic – Forms and Views	
	Set up the workspace and create the project		
	Create project	Create project app	
		Base.html	
		Create navigation and add boilerplates	
	User Login and Registration/ Profile	Create accounts app and back-end User Model for registration.	
		Login.html	
		Create Front end Registration form and view.	
		Modify Django Registration template.	
		Register.html	
		Expand the model and the form for complete user Profile.	
		Create back-end Authentication and Authorization logic for user login/ Front end Login form and view.	
		Modify Django Registration template	
	Create password reset feature	Create a password_reset view and modify Django Reset Password template.	
		Create backend email response.	
		Modify Dajango registration templates	
		Password_reset_complete.html	
		Password_reset_confirm.html	
		Password_reset_done.html	
		Password_reste_email.html	

		Password_reset_form.html	
	Create a landing/ Home page	Create a base to preview the components the users with different Authorizations may access	
		Home.html	
	Retrieve List of Products	Create products app and back end Product Model	
		Create front end admin Form for Product items (add highlight logic)	
		Create Search Form and View for user to filter Products based on a criterion (eg. a-z/ z-a – Title, Brand) Create a pagination feature	
		Sales.html	
	Retrieve Highlight of the month	Create a Highlight view	
		Create poll app and back end Poll Model	
		Create front end Poll Form and View	
		Highlight.html	
	Retrieve blogs	Create posts app and back end Post Model.	
		Create front end admin Form and view for posts.	
		Create blogcomments app and blogcomment Model.	
		Blogs.html	
		Blog.html	
		Edit.html (edit comment)	
		Delete.html (delete comment)	
	Allow user to add Product into the shopping cart, edit quantity and remove products from cart	Shopping Cart is a global context feature accessible at all times for Logged in users	
		Create cart app and back end Context	
		Create front end view	
		Cart.html	
	Allow user to purchase the products in the shopping cart.	Create a checkout app and back end Checkout Model	
		Create front end Form and view	
		Checkout.html	
	Allow user to create user profile, which data will be used to prefill order form and for comments owner information	Create Profile app	
		Create front end Form and view	
		Profile.html Updated_profile.html	
	Allow user to contact the company by sending a message	Create contact app	
		Create front end Form and view	
		Contact.html	
	AWS S3	Migrate static and media files	

	Travis	Testing with Travis before deploying to Heroku	
	Deploy to Heroku		
	Go over the check list and finish the documentation		
	Submit		

UI

Bootstrap for responsive design

Colors:
Bootstrap dark and warning as main color schema

Warning yellow to use for highlighting links, text emphasis and to mark active page in navbar

Bootstrap light as background for articles