Cy Dixon — Project Report — CS 396

Intro

This project is the creation of a website using Django and Python. The goal of this project was to create a friendly, useful, and responsive learning based website used for discussion along with teacher and student interaction. There were many functions implemented in this project. The order in which I will list them will be the app name, and then the sub files of the app along with any other extra features I have chosen to include.

Account

Models.py This app is what creates and manages users. Inside the models.py account app we link a model called Account to database values. Account extends MyAccountManager, which makes the creation and management of these accounts much easier. Inside this model i have included a isadmin tag that will determine if the user is a teacher or not. It is a boolean field so this makes it very easy to identify.

Views.py Inside here we have registration view which manages the rendering of the registration page. Also included is logout and login views that do their respective tasks. One other view here is the accountview method that shows the user's account details. Finally we have the mustauthenticateview that authenticates the user's using Django's authenticate classes.

Forms.py In forms we manage the 'forms' that will be displayed on the html pages, and how they relate to the database and models. For this particular app we have RegistationForm, AccountAuthenticateForm, and AccountUpdateForm. All of these forms do exactly what they sound like they do. They will pass the data from the html pages into the database.

Blog

Models.py Here we have Post model and the Reply model. These handle the posts and replies. Also included here is an uploadlocation method that will set the upload location for img and other file types to the local machine.

Views.py Createblogview, this displays the creation of a post form and html page. Deatilblogview that shows a already created posts details. edit blogview, allowing the user or teacher to edit the post. Also included is deleteblog and addreply.

Forms.py I wont go over every form here as it the same as the views, we need a form for the creation of a post and that is all.

Urls.py This urls.py file has the urlm patterns needed to direct the user to wherever need based on what is clicked in the html. Include are links to creation,detail, edit,delete, and add reply pages.

Quiz

Modles.py The models included with the quiz app are Quiz, Question, Answer, QuizAttepmt and QuestionAttempt. All of these models work together to form a cohesive and well implemented quiz. Within the quiz model we just have a name and slug for url function. The question model has a foreign key relationship with the quiz model so that we can delete the questions if the quiz gets deleted.

Views.py Views here is about what you would expect, a view for the quiz to display with the html, and to set up the form with context.

Forms.py The form used for this app is the QuizForm. We have this is display the question on the html and to receive the answers to link with the database and models.

Urls.py Urls here includes the quiz url and that is the only needed one for this app.

HitCount

This is not my own created app, however is was super userful and intuitive for counting ans updating page views so I included it. https://django-hitcount.readthedocs.io/en/latest/overview.html

Database

Django comes preinstalled with SQLite database tools so I just kept that the same for this project. Django also includes, inside the admin page of the local hosted website, admin tools for the database so not extra software was required to manage the database.

Software

The software for the text editing and coding scripting is Sublime Text. It is a very simple and clean text editor with file management tools. I found it very useful for my project. The macOS terminal is the shell that I have been using for debugging and diagnosing. I have learned it is not greate so I will most likey move over to VSCode for the next phase of this project. Virtual Env was used within this project for the virtual environment an version control was used as well. The main language used here was Python. Django works mostly in python so it was very easy to use and simple to implement. Some scripts were needed for html pages and bootstrap manipulation so a very small amount of javascript was implemented so help with page views. Bootstrap was also used in the creatin of html and the look of the html files. Bootstrap helped make the website look incredible and made it user friendly as well.

Discussion

I found that for this project in particular I should have been using VSCode or some other IDE. I have used Sublime as my text editor but it has its limitations. There also is the problem that there is no debugger in sublime. As for techniques for this project lots of bootstrap was used for menus and buttons and I found that using it made the website not only look better but function better, and I will definitely continue using it. One technique that I used was that inside of urls.py you can give the app a reference name. So for example in the blog posts app inside the urls.py I gave it a reference of 'blog'. This allowed me, inside the html to very easily reference the particular url for buttons and objects. The typical way that is done is by directly referencing the file path, however that is very long and you must find the file path/url. For the reference we can just call 'blog' or 'url' and concatinate the url name with ":" to get the url. Example is 'blog':delete. This made the process of creating html so much easier and more fluid, as well as easier to understand.

In conclusion I am very happy with how my website turned out, however I do with I could have added more to it.