* Setting up Python and Django, and using virtual.
* A Django project has various parts, like settings, URLs, views, and templates, that work together.
* Models are like blueprints for your database.
* Admin panels can be quickly created to handle data.
* Views handle web pages and URLs are connected to these views using patterns.
* Django's special template language lets you create dynamic HTML pages by adding data from views.
* Managing static files (styles, scripts) and media files (images) is done easy.
* Making forms and validating user input is made easy with Django's Form classes.
* User stuff like registration, logins, and logouts is well-handled by Django's built-in system.
* Querying databases is simplified using Django's tools for filtering, sorting, and aggregating data.
* Smart URLs allow for flexible handling of different types of content.
* Django's middleware system manages requests and responses across your app.
* Deploying your Django project on platforms like Heroku or AWS is the final step to going live.