Explanation:

* **Design**: The design itself is kept simple. I have modularized app.py to multiple files that handle each task. I have used the standard MVC architecture to build out the application.
  + Models: The application has 2 models, one of which handles user data (authentication) and the other handles the posts data (defined in models.py).
  + Views/ Controller: I have merged these functionalities for now due to the simplicity of the application. My routes.py handles the routing and handling the requests using REST APIs.
  + Authentication: For Authentication, I have used HTTPBasicAuth library. It has functionality to both register and login users (in Auth.py). As an additional step to security, I have also implemented methods in my user model that hash my password while storing it in the DB and unhash it when authentication is needed.
  + Configuring the Database: To try maintaining my data connection a secret, I have implemented a technique for database connectivity where it fetches the credentials from the system’s environment variable instead of hardcoding it (you can hardcode it for your testing purpose). (refer to config.py)
  + Testing: I have implemented a folder that contain test files for the app. Each one handles different scenarios like testing authentication and testing all my post requests.
* **Trade-offs:**
  + I have implemented simple authentication for this project given more time, I would have implemented JWT which would handle time outs better.
  + Views and Controllers could be split for a cleaner code but given its limited functionality as of now, it would be much more understandable as is. If the complexity were to increase in the future, I would modularize it even further.
  + I have currently used a deployment server to serve my application but if we were to productionize it, I would use WSGI to better serve the application.
* **Additional Improvements:**
  + Use better authentication using SSO/OAuth.
  + Try implementing a UI for better understanding of project.
  + Implement additional features into the posts like uploading images, adding comments, likes. Also add chat functionality.
  + Deploy it on cloud and possibly build a recommendation system based on the types of posts a user views.