Deeper Systems Interview Exercise Questions

Hong Chen

1. What will the database schema look like for this site? What will the tables be? What will the columns be? What indices will you use, if any, and why?

Since I am using Django default database as SQLite3, I have two tables in my database: post and comment.

The post table includes **author**, **title**, **post\_content**, **timestamp**, and **slug** with it and it also has the automatically generated **id** by Django for each object in the table.

The comment table includes **generated id**, **author**, **comment\_content**, **timestamp**, and a ForeignKey **parent** associated with objects in posts.

And while I transfer data to JSON format by using Django REST framework, I use serializer to generate url for every object in my database in order to view elements in each object with API web interface (<http://127.0.0.1:8000/api/>).

1. What API end points will you need? List them all out, with a description of what they do and what they return.

In this exercise, I cannot display required information as in JSON format in front-end website due to the knowledge I have based on JQuery and Javascript, I can only display data through Django views and templates.

For views.py in posts application, the first function will be “post\_list” displaying list of blog posts. I use $queryset = Post.objects.all().order\_by(“-timestamp") to get all post data from database and sorting by most recent post.

To create a new post on front-end,  “post\_create” is to send a POST request to back-end with designed form format by $form = PostForm(request.POST or None) and further save the data only if the required fields has been provided.

Viewing each post content and it’s comments, the “post\_detail” acquire comments by $comments = Comment.objects.filter(parent=[instance.id](http://instance.id/)) and all comments are filtered and display only with specific id to the post. At the post\_detail, a new comment can also be generated with similar function to the “post\_create” but in comment\_form and its regulations.

In order to transfer SQLite data to JSON, I use Django REST framework with serializers to each model I created. The serializer gather data with tables I created to each model and further presenting these JSON data on the API web interface.

1. How would you handle large data sets? For example what will your database queries and API calls look like if you have 100,000 posts in the database?

I would say replacing SQLite by using MYSQL would help in this case. The SQLite is an embedded relational database management system, which means the database will be file-based and working fine with single database file. However, SQLite can only handle single write operation at one time that would not be able to satisfy large scale application operations. Since I don’t have any experience with MYSQL, I would say MYSQL would be able to handle distributed operations and it also has been considered as a popular solution to current market.