Report on Blog API Design and Development

Abstract

The blog API was designed and developed using Laravel (PHP) framework. The Api gets his data from open Api (Api Communication). Fetched his data first from cache and if no data returned, local database will be called and finally to open Api repository incase if the cache and local database returned nothing. The system was designed using singleton design in which the repository classes are initialized once with the help of dependency injection. This makes the system scalable and maintainable. Mysql was used for the local database with three tables (users, posts, comments).

Introduction

Solve the problems first, then code. Lots of thoughts were put in place, having known the framework and language to use as specified by the questions, then come down to the tool, the plugins etc. to enhance the successfulness of the project, the following pattern were taken to considerations.

1. **Requirement and Planning**

   The question was cleared with its needs. And to carried out the operation, a well-defined planning was put in place, as regards to plugin/extension, devops tool, etc that will enhance the work, for the sake of this project, git was only devops tool used for pushing code to gitup, guzzle extension(Laravel HTTP ext.) was used for consuming open Api and postman helps in testing the APIs.

2. **Design**

   In any project, well thorough design is the key, in fact a particular design has been norms for individual and companies. If design was done write and followed to the letter, future maintenance will be at no cost, in this regards I used singleton software design for this project. It is modern, fast, and scalable though it could be tedious at the beginning. It is simple, class will serve as repository, and interface will hold the contracts for the class, the controller can use repository class by injecting in its constructor (dependency injections).

3. **Implement, Coding and Database**

   Requirement is well understood and the agreed design has been concluded, now implementation is the next. According the requirement, three tables (users, posts, and comments) and created via migration, with posts have user_id as foreign key and comment has post_id as foreign key (RDBMS). Mysql was used to implement this. Post will be searched by title (contains), therefore fulltext constraint are added to the title attribute. There are three core modules (user, post and comment modules), a repository (Model, Controller, Interface etc) was created for each for easy maintainability.

4. **Testing**

   Testing is a necessity evil, in this project I embarked in using TDD(write test first, then code to make it pass), since the requirement mainly is “get verbs”, couples of test were written to assert if http status code are 200 or 404 or 500.

5. **Conclusion**

   The task were done and followed in line with the instructions. The task folder were structured for easy navigation. Each task with its own folder. 1. BasicCS has four folder (four questions) and 2. Advance contain the API project.