Software Design Document

Programming Blog Website

Version 1.0
Finished: May 7, 2023
Vladyslav Braslavskyi

Table of Contents

1. Introduction

1.1. Purpose

The Software Design Document describes the architecture and system design for the Programming Blog website. Programming Blog is designed to help programmers remember the most often used technologies and prepare for the interview.

1.2. Scope

This document describes the implementation details of the Programming Blog website. The content will consist of six major components: Trip Planning, Database, Optimization, Map, User, and Authentication. Each of the components will be explained in detail in this Software Design Document.

1.3. Technologies

1.3.1. TypeScript

TypeScript is a strongly typed programming language that builds on JavaScript, giving you better tooling at any scale.

JavaScript and More: TypeScript adds additional syntax to JavaScript to support a tighter integration with your editor. Catch errors early in your editor.

A Result You Can Trust: TypeScript code converts to JavaScript, which runs anywhere JavaScript runs: In a browser, on Node.js or Deno and in your apps.

Safety at Scale: TypeScript understands JavaScript and uses type inference to give you great tooling without additional code.

1.3.2. NestJS

Nest (NestJS) is a framework for building efficient, scalable Node.js server-side applications. It uses progressive JavaScript, is built with and fully supports TypeScript.

Nest vs Angular, React, and Vue: The main difference between Nest and Angular, React, and Vue is that Nest is used to develop server-side applications, while Angular, React, and Vue are used to develop front-end applications.

Nest vs Express: Both Nest and Express are used to develop server-side applications. The main difference is that Express is a minimalist framework which does not provide a predefined architecture. Instead, Nest provides an out-of-the-box application architecture which allows developers and teams to create highly testable, scalable, loosely coupled, and easily maintainable applications. The Nest architecture is heavily inspired by Angular.

1.3.3. MySQL

MySQL is a relational database management system (RDBMS) that is based on structured query language (SQL).

MySQL vs PostgreSQL: MySQL is a purely relational database, whereas PostgreSQL is an object-relational database. Today, MySQL is one of the most popular and widely used SQL databases.

1.3.4. GitHub

GitHub, Inc. is an Internet hosting service for software development and version control using Git. It provides the distributed version control of Git plus access control, bug tracking, software feature requests, task management, continuous integration, and wikis for every project.

1.3.5. Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.

With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

1.3.6. Koyeb

Koyeb is a cloud platform that enables developers to host websites and web services that deploy instantly, scale automatically, and require no supervision.

1.3.7. Swagger

Swagger is a set of open-source tools built around the OpenAPI Specification that can help you design, build, document and consume REST APIs.

2. System Overview

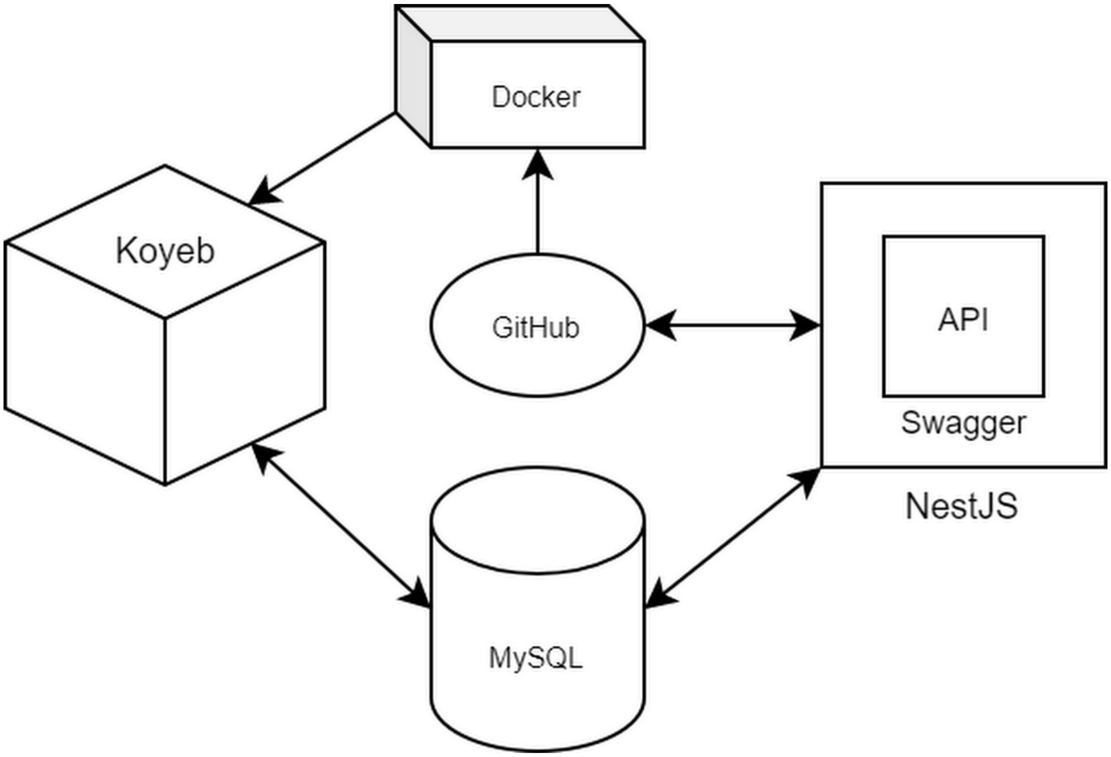
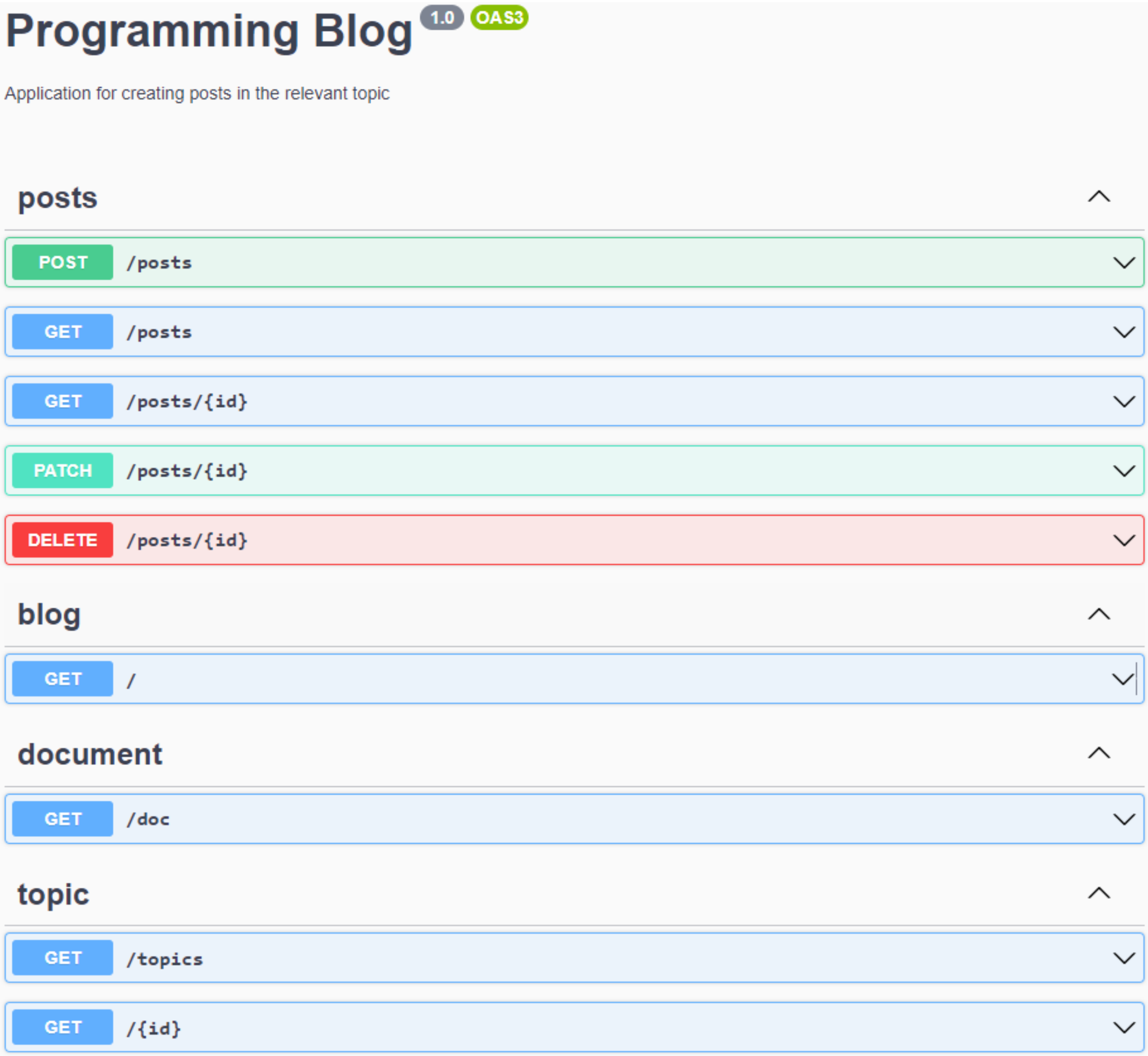


Figure 2.1: System Overview

The system is a web application built using the NestJS framework and runs on the Koyeb serverless platform. Koyeb provides a powerful and flexible environment for building, deploying, and scaling serverless applications. The application's data is stored in a MySQL database.

The website is deployed and managed using Docker containers, which provide a portable and efficient environment for running applications. Docker allows for easy scalability, as additional instances of the application can be added to handle increased traffic.

The application's code is hosted on GitHub, which provides version control and collaboration tools for developers. Changes to the code are managed through a Git workflow, with code reviews and pull requests used to ensure quality and maintainability.



Programming Blog 1.0 OAS3

Application for creating posts in the relevant topic

posts

POST /posts

GET /posts

GET /posts/{id}

PATCH /posts/{id}

DELETE /posts/{id}

blog

GET /

document

GET /doc

topic

GET /topics

GET /{id}

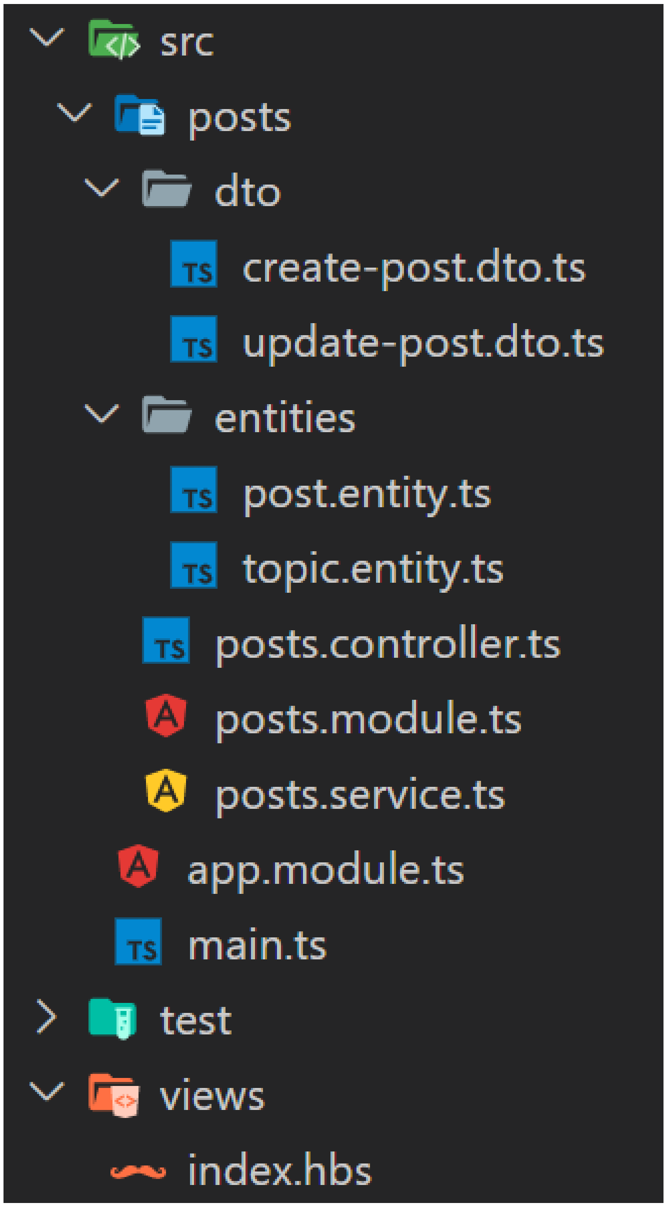
Figure 2.2: API Overview

NestJS provides an out-of-the-box application architecture, which helps to create APIs using TypeScript or JavaScript. It uses decorators, dependency injection, and other popular design patterns to make the development process easier and more organized.

Overall, this system provides a flexible, scalable, and efficient platform for building and deploying web applications using NestJS, MySQL, GitHub, Docker, and Koyeb.

3. System Components

3.1. Components structure



```
src
  posts
    dto
      create-post.dto.ts
      update-post.dto.ts
    entities
      post.entity.ts
      topic.entity.ts
    posts.controller.ts
    posts.module.ts
    posts.service.ts
  app.module.ts
  main.ts
test
views
  index.hbs
```

Figure 3.1: Components structure

3.2. Entities

An Entity is a data model that represents a business object, such as a user or a product. It typically maps to a database table and is responsible for defining the schema and relationships.

```
+---------+--------------+------+-----+---------+----------------+
| Field   | Type         | Null | Key | Default | Extra          |
+---------+--------------+------+-----+---------+----------------+
| id      | int          | NO   | PRI | NULL    | auto_increment |
| title   | varchar(255) | NO   |     | NULL    |                |
| content | text         | NO   |     | NULL    |                |
| topicId | int          | YES  | MUL | NULL    |                |
+---------+--------------+------+-----+---------+----------------+
```

Figure 3.2.1: Post table

```
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| id    | int          | NO   | PRI | NULL    | auto_increment |
| name  | varchar(255) | NO   |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
```

Figure 3.2.2: Topic table

3.3. Controllers

A Controller is responsible for handling incoming HTTP requests and returning appropriate HTTP responses. It receives the request, delegates the processing to the relevant service, and then returns the response.

3.4. Services

A Service is responsible for handling business logic, data manipulation, and data retrieval operations. It contains all the necessary logic and data access to fulfill the request from the Controller.

3.5. DTO

A DTO is a plain TypeScript class that represents the data that is exchanged between the client and server. It is used to ensure type-safety and to decouple the internal data model from the external data representation.

3.6. Modules

A Module is a logical boundary within the application and encapsulates related functionality. It typically groups related Controllers and Services together.

3.7. Pages

Handlebars (hbs) is a popular templating engine that allows you to generate dynamic HTML pages using a template and data. NestJS provides support for Handlebars through the @nestjs/plus package.

4. MVC

Model-view-controller (MVC) is a software architectural pattern commonly used to develop web applications containing user interfaces. This pattern divides the application into three interconnected elements.

- Model contains the business logic of the application. For example, the online store application product data and its functions.
- View contains the user interface of the application. For example, a view to register products or users.
- Controller acts as an interface between model and view elements. For example, a product controller collects information from a “create product” view and passes it to the product model to be stored in the database.

Nest provides support for the MVC pattern thanks to the integration of the Handlebars templating engine. The MVC pattern provides some advantages: better code separation, multiple team members can work and collaborate simultaneously, finding an error is easier, and maintainability is improved.

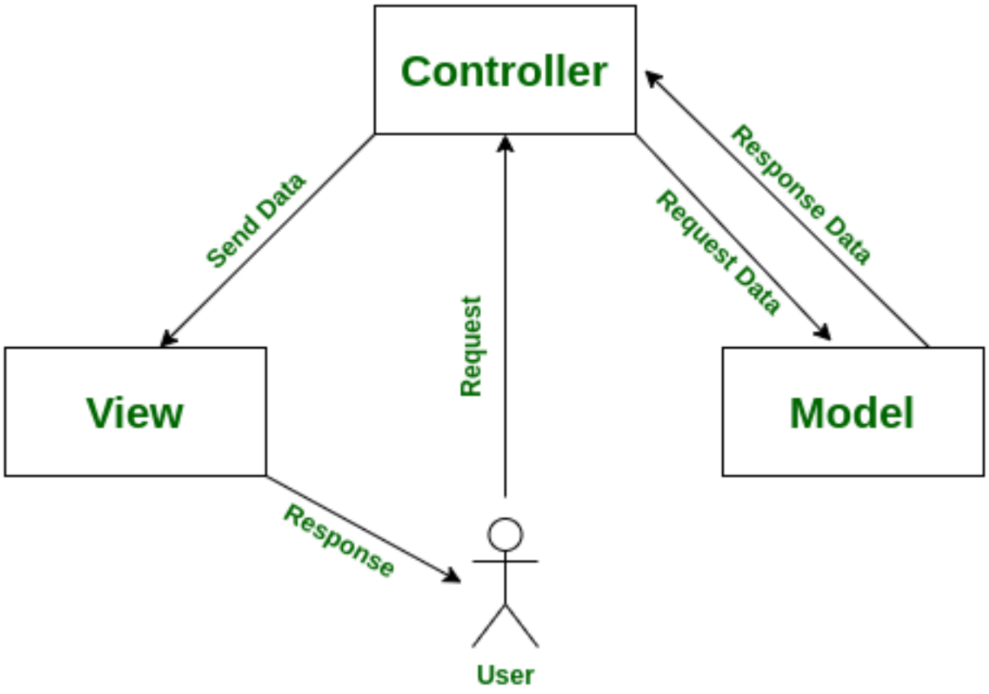


Figure 4: MVC