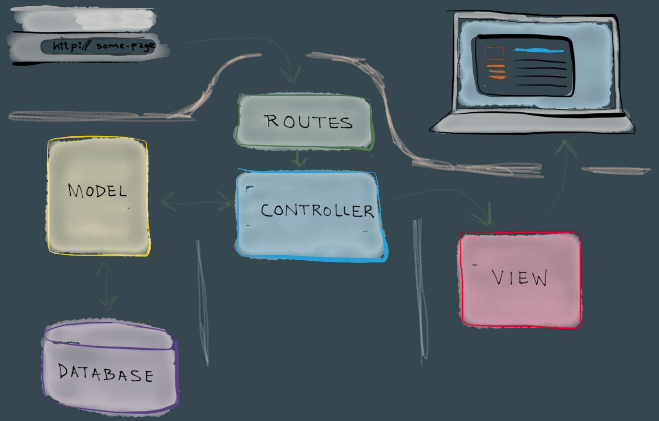
MVC Architecture

By Tarun Raja

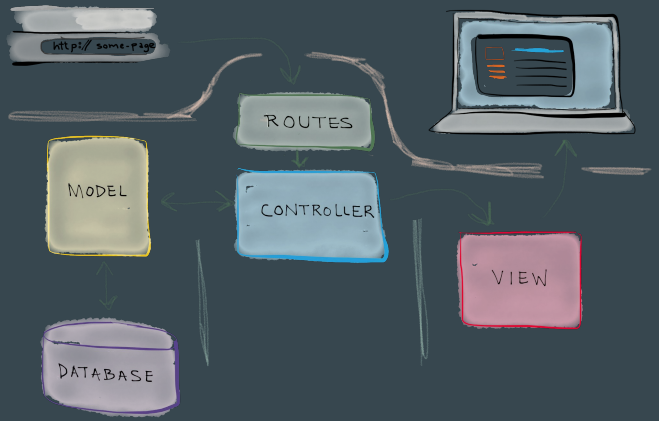
Routes

Once the user types in a website route in its url, the project starts in the server.js file which points to the inputted router.



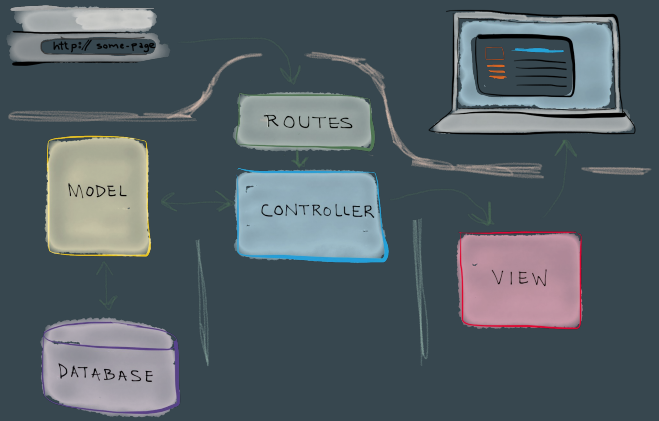
Controller

The router points to a controller depending on the type of CRUD operation that the controller can listen to. The controller then executes the operation by interacting with the model to use the db, or the views to send the user html depending on the operation.



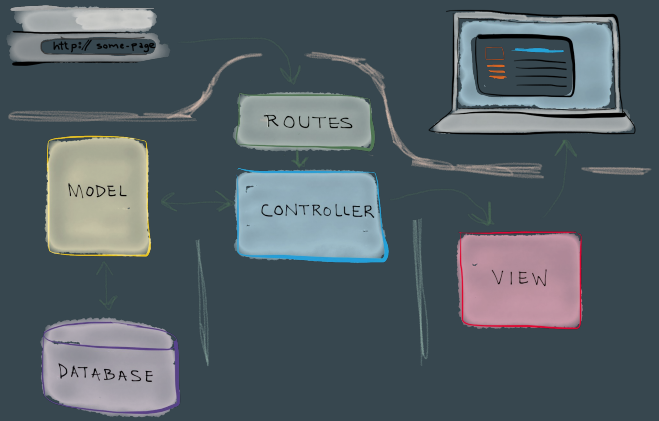
Views

The views contain .ejs or any other type of html templating or front-end frameworks which are displayed to the user depending on the specific get request that is being executed.



Model

The model folder contains js files that contain schemas for the objects of our database and interacts with the mongodb database using mongoose



Pros

- Allows for better abstraction
- Separation of concerns
- Popular architecture which is commonly understood by other developers (i.e. useful in a team environment)
- Allows for a greater degree of flexibility and adaptability by the developer to make changes to the codebase quickly and effectively
- Neat and organised structure

Cons

- May reduce speed at which the project code is executed as there are various callbacks and files, however this has a very minimal impact

Mongoose

Mongoose, as opposed to MongoClient, allows for the use of mongodb schemas, which are templates/blueprints for the objects/documents for our databases/collections! This is another layer of abstraction provided by MVC Architecture