NodeJS Clean Architecture.

Clean Architecture is an opinionated boilerplate for Node web APIs focused on separation of concerns and scalability with Uncle Bob Clean Architecture implementation.

Features:

> Union - Layered folder structure: code organization focused on codebase scalability.

> Instant feedback and reload: use Nodemon to automatically reload the server after a file change when on development mode, makes the development faster and easier.

> Scalable and easy to use web server.

> Use Express for requests routing and middlewares. There are some essential middlewares for web APIs already setup, like body-parser, compression, and method-override.

> Database integration: Mongoose, a MongoDB Database Connection, is already integrated; you just have to set the authentication configurations.

> Prepared for testing: The test suite uses Mocha, Chai and is prepared to run unit, integration and functional tests right from the beginning. A FactoryGirl adapter for Mongo is setup to make your tests DRY as well, and the tests generate code coverage measurement with. You should read about the Chai plugins that are setup by default too.

> Dependency injection: With Awilix, a practical dependency injection library, the code will not be coupled and it'll still be easy to resolve automatically the dependencies on the runtime and mock them during the tests. It's even possible inject dependencies on your controllers with the Awilix Express adapter.

>Logging: The Log4js logger is highly pluggable, being able to append the messages to a file during the development and send them to a logging service when on production. Even the requests (through Morgan) and queries will be logged.

> Linter: It's also setup with ESLint to make it easy to ensure a code styling and find code smells.

How to use:

Notice that the boilerplate comes with a small application for user management already; you can delete it with a npm script after you understand how the boilerplate works but please do the quick start first!

1. Clone the repository.

2. Setup the database on `config/database.js` (there's an example file there to be used with MongoDB).

3. Install the dependencies with `yarn`.

4. Create the development and test databases you have setup on `config/database.js`

5. Run the application in development mode with `npm run dev`

6. Access `http://localhost:3000/api/users` and you're ready to go!

Compiled and presented by Vakindu Philliam.