Aplikasi ini dikembangkan dengan standar MVC.

Tech stack yang digunakan:

1. Mongodb
2. Mongoose
3. Redis
4. Async dan race condition
5. Docker

Hanya saja, deployment belum sempat dilakukan ke cloud karena ada batasan laptop untuk melakukan docker-compose. Oleh karena itu, deployment masih dilakukan manual di docker.



Gambar 1 Struktur Folder Aplikasi

**Struktur folder yang dipakai:**

1. Folder Config berisi seluruh config untuk seluruh aplikasi dan ini sudah menyesuaikan dengan environment online seandainya inject data environment yang sensitive melalui: process.env.PORT misalnya.
2. Folder Routes berisi routing untuk satu path tertentu, misalnya: user.routes.js untuk routing /users.
3. Folder Controllers berisi setiap fungsi logic yang dipanggil langsung oleh routes. Misalnya: user.routes.js menggunakan fungsi yang ada di user.controller.js.
4. Folder Services berisi setiap proses query yang digunakan oleh controller. Misalnya: user.controller.js menanggil fungsi query di UserService.js.
5. Folder Model berisi setiap mapping DTO (Data Transfer Object) dari aplikasi ke table. Misalnya: user.js berisi mapping DTO ke table terkait di table users.
6. Folder Middleware berisi middleware yang digunakan dalam tiap routes. Misalnya: user.routes.js beberapa path menggunakan validasi token.

**Alur aplikasi:**

Index.js -> user.routes.js -> user.controller.js -> user.js

**Proses Instalasi wajib terinstall docker di laptop / PC testing:**

1. Setup mongodb:
   1. docker volume create --name=mongodata
   2. docker run --name mongodb -v mongodata:/data/db -d -p 27017:27017 mongo
   3. winpty docker exec -it mongodb bash (windows) atau docker exec -it mongodb bash
   4. setelah masuk ke bash, ketik mongo
   5. dalam mongo cli, ketik use aguskianto untuk create DB aguskianto
   6. tambahkan user di mongo cli dengan cara: db.createUser({user:"aguskianto", pwd:"4gusk14nt0", roles:[{role:"readWrite", db: "aguskianto"}]});
   7. tambahkan collection users dan isi dengan data sebagai berikut:



* 1. Keluar dari mongo cli.
  2. docker stop mongodb
  3. docker rm mongodb
  4. docker run --name mongodb -v mongodata:/data/db -d -p 27017:27017 mongo –auth

Setelah perintah ini, connection wajib menggunakan username dan password dalam DB connection: mongodb://aguskianto:4gusk14nt0@localhost:27017/aguskianto

1. Setup redis:
   1. docker run --name container-redis -d -p 6379:6379 redis
2. Setup aplikasi:
   1. Dari folder root project di terminal, jalankan: npm install
   2. Lalu, jalankan: npm start
   3. Sebenarnya, bagian ini juga bisa didockerize tetapi ada kendala di windows 10 home edition sehingga batasnya hanya sampai sini saja.
3. Daftar list functional yang bisa dilakukan:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **URL** | **Method** | **Fungsi** | **Notes** |
| 1 | <http://localhost:8080/token> | GET | Mendapatkan token | Tidak ada pengecekan token |
| 2 | <http://localhost:8080/> | GET | Mengetes aplikasi berjalan atau tidak | Tidak ada pengecekan token |
| 3 | <http://localhost:8080/users> | GET | Menampilkan data user | Menggunakan token |
| 4 | <http://localhost:8080/users> | POST | Menambah user | Menggunakan token |
| 5 | <http://localhost:8080/users/user/605d993a1d0f5b355459ca7e> | PUT | Mengubah data user | Menggunakan token |
| 6 | <http://localhost:8080/users/user/605d993a1d0f5b355459ca7e> | DELETE | Menghapus data user | Menggunakan token |
| 7 | <http://localhost:8080/users/user/accountNumber/IT17020150> | GET | Query berdasarkan account number | Menggunakan token |
| 8 | <http://localhost:8080/users/user/identityNumber/12345678> | GET | Query berdasarkan identity number | Menggunakan token |

**Bagian yang bisa dienhance:**

1. Unit testing
2. Design pattern menggunakan typescript