

UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS
CARRERA DE INGENIERIA EN SOFTWARE

FABRICA DE SOFTWARE

NOMBRE: Heinz Delgado, Isaac Romero, Gustavo Gualán, Mario Salazar

DOCENTE: MSC. Antonio Quiña Mera

FECHA: 12/01/2024

INDICACIONES:

- Documentación de la Aplicación Backend UTN

DESARROLLO:

Integrantes	Rol	Porcentaje
Isaac Romero	Scrum Master	100%
Heinz Delgado	Developer	100%
Gustavo Gualán	Developer	100%
Mario Salazar	Developer	100%

Repositorio Código: https://github.com/Isaacmirex/app_backend_utn

Repositorio Azure:

Contenido

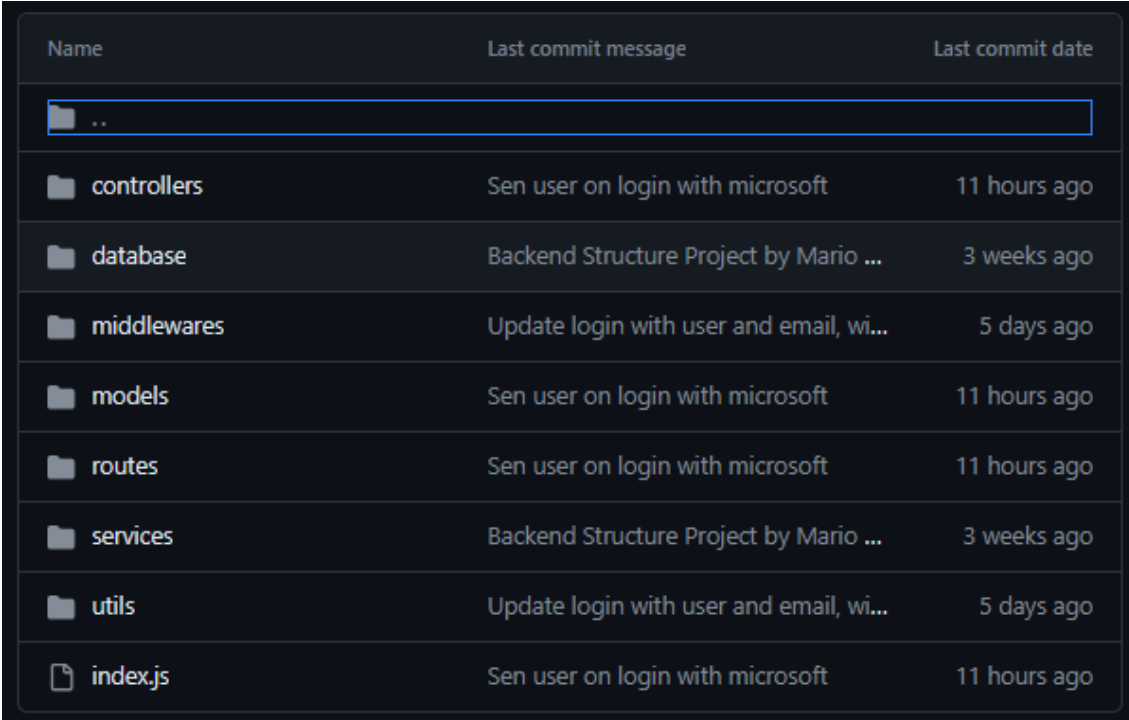
Estructura general de la APP UTN BACKEND.....	3
CONTROLLERS	3
DATABASE	5
MIDDLEWARES	6
ROUTES.....	6
INDEX.....	9

Ilustración 1 Estructura General Api	3
Ilustración 2 Controladores	4
Ilustración 3 Lógica de los controladores	5
Ilustración 4 Base de datos.....	5
Ilustración 5 Conexión a la BD.....	6
Ilustración 6 Login	6
Ilustración 7 Rutas de los controladores	7
Ilustración 8 Rutas documentadas	8
Ilustración 9 Rutas en Swagger	9
Ilustración 10 Rutas en el Index 1.....	10
Ilustración 11 Rutas en el Index 2.....	11
Ilustración 12 Rutas en el Index 3.....	11

Estructura general de la APP UTN BACKEND

La siguiente aplicación fue desarrollada para poder dar una solución a un proyecto basado en Asserts los cuales estén correctamente documentados para su futuro uso en otras aplicaciones, en esta sección se va a desarrollar las rutas y dependencias para poder levantar el Backend de la aplicación.

En la siguiente imagen se muestra la estructura general de cómo se va a desarrollar y que se va a utilizar en la aplicación que se va a desarrollar de acuerdo con los requerimientos del Product Owner, este proyecto va a ser realizado por diferentes grupos de trabajo en los cuales tendrán un líder y delegará actividades a desarrollar, basándose en Sprints para el entendimiento y coordinación del desarrollo.



Name	Last commit message	Last commit date
..		
controllers	Sen user on login with microsoft	11 hours ago
database	Backend Structure Project by Mario ...	3 weeks ago
middlewares	Update login with user and email, wi...	5 days ago
models	Sen user on login with microsoft	11 hours ago
routes	Sen user on login with microsoft	11 hours ago
services	Backend Structure Project by Mario ...	3 weeks ago
utils	Update login with user and email, wi...	5 days ago
index.js	Sen user on login with microsoft	11 hours ago

Ilustración 1 Estructura General Api

CONTROLLERS

En esta sección se organizan y distribuyen los controladores de los diferentes requerimientos establecidos por el Product Owner, estos controladores ayudan a cumplir ciertas funciones CRUD dependiendo de la solicitud del producto, en ellos se basa y se maneja la lógica de programación con la cual se podrá asignar funcionalidades al sistema en sus diferentes módulos.

En la siguiente imagen se muestra los diferentes controladores que se utilizarán en la aplicación:

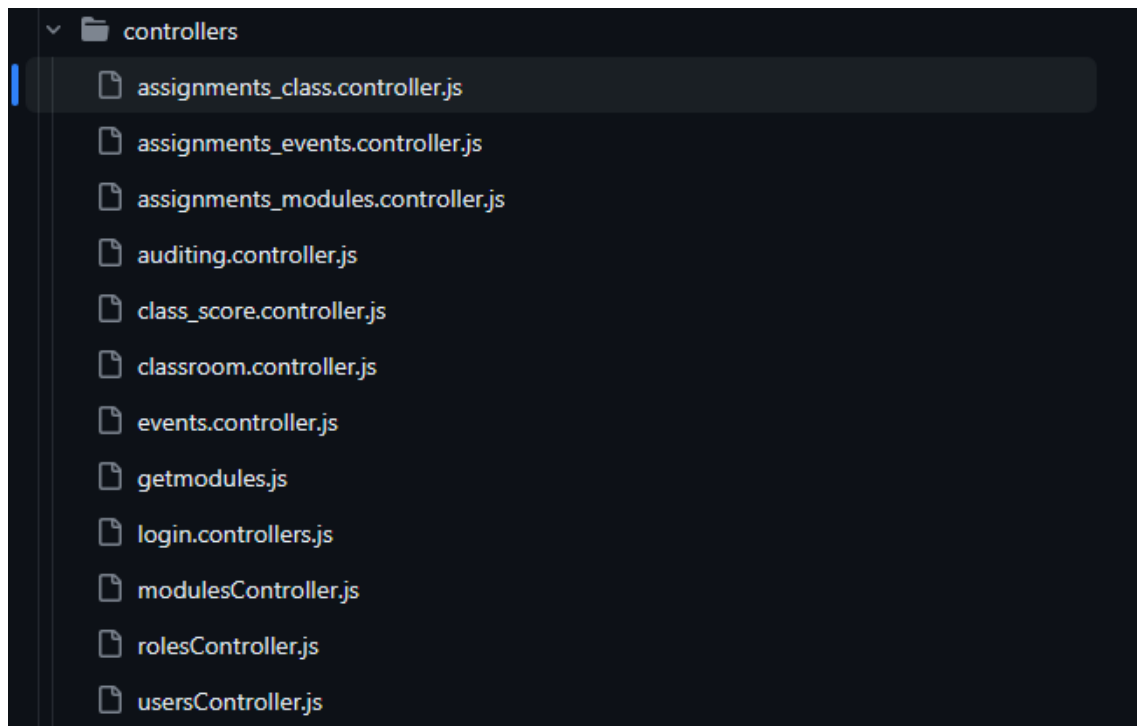


Ilustración 2 Controladores

En la siguiente imagen se muestra la lógica y funcionalidades de uno de los controladores:

```

const updateAssignmentsClass = async (req, res) => {
  try {
    const {assignment_class_id, class_id, user_id, assignment_class_state} = req.body;
    const response = await client.query(
      "UPDATE public.assignments_class SET class_id = $1, user_id = $2, assignment_cla
      [class_id, user_id, assignment_class_state, assignment_class_id]
    );
    res.json({
      message: "AssignmentsClass Updated successfully",
      body: {
        assignments_class: {
          assignment_class_id,
          class_id,
          user_id,
          assignment_class_state,
        },
      },
    });
  } catch (err) {
    console.error('Error updating assignments_class', err);
    res.status(500).json({error: 'An error occurred while updating assignments_class'});
  }
};

export {
  getAssignmentsClass,
  getAssignmentsClassById,
  createAssignmentsClass,
  updateAssignmentsClass,
}

```

Ilustración 3 Lógica de los controladores

DATABASE

En esta sección se almacena el archivo.js de la base de datos que utiliza la aplicación, la cual se maneja en base a clientes, en esta sección se realiza la conexión de la base de datos con la Api, la base de datos tiene permisos de accesos a los desarrolladores la cual se encuentra desplegada en Azure.

..		
DigiCertGlobalRootCA.crt.pem	Backend Structure Project by Mario Salazar	3 weeks ago
database.js	Backend Structure Project by Mario Salazar	3 weeks ago

Ilustración 4 Base de datos

```

1  import pkg from 'pg';
2  import fs from 'fs';
3  const {Client} = pkg;
4  import {config} from 'dotenv';
5  config()
6  const client = new Client({
7      host: process.env.POSTGRES_HOST,
8      user: process.env.POSTGRES_USER,
9      password: process.env.POSTGRES_PASSWORD,
10     database: process.env.POSTGRES_DATABASE,
11     port: process.env.POSTGRES_PORT,
12     ssl: {
13         ca: fs.readFileSync('./src/database/DigiCertGlobalRootCA.crt.pem')
14     }
15 });
16
17 client.connect()
18     .then(() => console.log('Conecting successfull'))
19     .catch(err => {
20         console.error('Error with database ', err);
21         client.end();
22     });
23
24 export {client};

```

Ilustración 5 Conexión a la BD

MIDDLEWARES

En esta sección se realiza los accesos basándose en la plataforma y Api de Microsoft, esta sección se detalla en el documento específico relacionado a la seguridad y login de la Aplicación:

Name	Last commit message	Last commit date
..		
microsoft.js	Update login with user and email, withou...	5 days ago
verifyAccess.js	Update Routes, Merge	last week

Ilustración 6 Login

ROUTES

En esta sección se realiza o se establece las rutas en las cuales los controladores van a realizar las funcionalidades designadas, es donde se van a ubicar, las mismas que sirven para el consumo de servicios superiores, las cuales le dan la estructura organizacional a la aplicación.

Name	Last commit message	Last commit date
..		
assignments_class.js	Update Routes, Merge	last week
assignments_events.js	Update Routes, Merge	last week
assignments_modules.js	Nueva ruta de usuarios con sus modulos ...	yesterday
auditing.js	Update Routes, Merge	last week
class_score.js	Update Routes, Merge	last week
classroom.js	Update Routes, Merge	last week
events.js	Update Routes, Merge	last week
getmodules.routes.js	Sen user on login with microsoft	11 hours ago
login.routes.js	Update login with user and password	5 days ago
microsoft.js	Sen user on login with microsoft	11 hours ago
modules.js	Update Routes, Merge	last week
roles.js	Update Routes, Merge	last week
swagger.js	Sen user on login with microsoft	11 hours ago
users.js	Update login with user and email, withou...	5 days ago

Ilustración 7 Rutas de los controladores

En la siguiente imagen se muestra un ejemplo de las rutas que se manejan en la Api, cada una está documentada con Swagger para un mejor entendimiento:

```

import { Router } from 'express';
import { getAssignmentsClass, getAssignmentsClassById, createAssignmentsClass,
        updateAssignmentsClass } from '../controllers/assignments_class.controller.js';

const assignments_classRouter = Router();
//Routes AssignmentsClass
/**
 * @openapi
 * /utnbackend/v1/assignments_class:
 *   get:
 *     tags:
 *       - Assignments Class
 *     responses:
 *       200:
 *         description: OK
 *         content:
 *           application/json:
 *             schema:
 *               type: object
 *               properties:
 *                 status:
 *                   type: string
 *                   example: OK
 *                 data:
 *                   type: array
 *                   items:
 *                     type: object
 */
assignments_classRouter.get('/', getAssignmentsClass);

```

Ilustración 8 Rutas documentadas

Users			^
GET	/utnbackend/v1/users		✓
POST	/utnbackend/v1/users	Create a new user	✓
GET	/utnbackend/v1/users/{id}		✓
PUT	/utnbackend/v1/users/{id}	Update an existing user	✓
Modules			^
GET	/utnbackend/v1/modules		✓
POST	/utnbackend/v1/modules	Create a new module	✓
GET	/utnbackend/v1/modules/{id}		✓
PUT	/utnbackend/v1/modules/{id}	Update an existing module	✓

Ilustración 9 Rutas en Swagger

INDEX

En esta sección se maneja todas las rutas que van a ser visibles para la aplicación ya en producción, manejando accesos y control, inicios de sesión, enrutamientos, páginas de inicio.

```

1  import express from "express";
2  import session from "express-session";
3  import cors from "cors";
4  import passport from "passport";
5  import {loginRouter} from "../routes/microsoft.js";
6  import {usersRouter} from "../routes/users.js";
7  import {rolesRouter} from "../routes/roles.js";
8  import {modulesRouter} from "../routes/modules.js";
9  import {assignments_modulesRouter} from "../routes/assignments_modules.js";
10 import {eventsRouter} from "../routes/events.js";
11 import {assignments_eventsRouter} from "../routes/assignments_events.js";
12 import {classroomRouter} from "../routes/classroom.js";
13 import {assignments_classRouter} from "../routes/assignments_class.js";
14 import {class_scoreRouter} from "../routes/class_score.js";
15 import {auditingRouter} from "../routes/auditing.js";
16 import "../middlewares/microsoft.js";
17 import {authorize} from "../middlewares/verifyAccess.js";
18 import {swaggerDocs as V1SwaggerDocs} from "../routes/swagger.js";
19 import {router} from '../routes/login.routes.js'
20 import { getmRouter } from "../routes/getmodules.routes.js";
21 const port = process.env.PORT || 3000;
22
23 const app = express();
24 app.use(express.json());
25 app.use(express.urlencoded({extended: false}))
26 app.use(cors());

```

Ilustración 10 Rutas en el Index 1

```

28   app.use(
29     session({
30       secret: '$serverbackutn',
31       resave: true,
32       saveUninitialized: true,
33       cookie: {
34         secure: false, // Cambiar a true si estás usando HTTPS
35         maxAge: 24 * 60 * 60 * 1000, // Tiempo de vida de la sesión en miliseg
36       },
37     })
38   );
39
40   app.use(passport.initialize());
41   app.use(passport.session());
42   app.use("/auth", loginRouter);
43
44   //Hola mundo en el servidor de bienvenida
45   app.get('/', (req, res) => {
46     res.send(`Hola mundo es una API de Login`);
47   });
48
49   //swagger
50   V1SwaggerDocs(app, port);
51
52   app.use('/utnbackend/v1/mario', getmRouter)
53   app.use('/utnbackend/v1/login', router)
54   app.use('/utnbackend/v1/users', usersRouter);
55   app.use('/utnbackend/v1/roles', rolesRouter);
56   app.use('/utnbackend/v1/modules', modulesRouter);
57   app.use('/utnbackend/v1/assignments_modules', assignments_modulesRouter);

```

Ilustración 11 Rutas en el Index 2

```

58   app.use('/utnbackend/v1/events', eventsRouter);
59   app.use('/utnbackend/v1/assignments_events', assignments_eventsRouter);
60   app.use('/utnbackend/v1/classroom', classroomRouter);
61   app.use('/utnbackend/v1/assignments_class', assignments_classRouter);
62   app.use('/utnbackend/v1/class_score', class_scoreRouter);
63   app.use('/utnbackend/v1/auditing', auditingRouter);
64   //app.use('/utnbackend/v1/assignments_modules', authorize(['Administrador'], ['Ass
65
66   app.listen(port, () => {
67     console.log(`Escuchando en el puerto: http://localhost:${port}`);
68
69   });

```

Ilustración 12 Rutas en el Index 3