Introduction

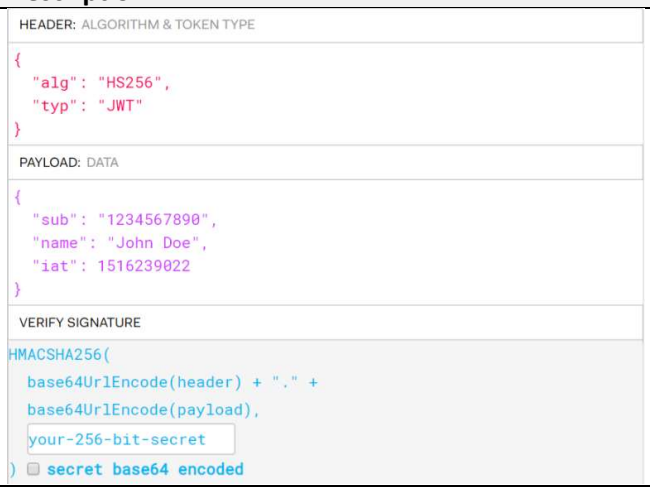
This is the server slide report for CAB230 assessment. This server slide application has implemented all requirement. The table below shows the completion of the application

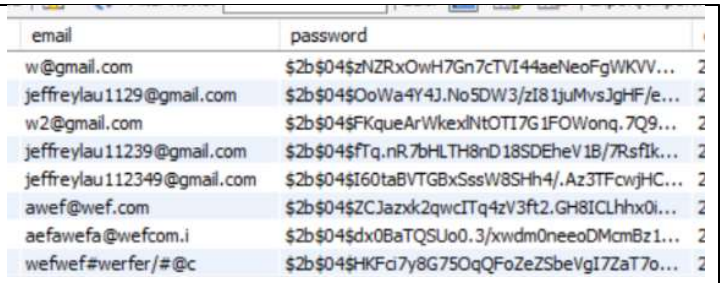
End point	**states**
/register	DONE
/login	DONE
/search	DONE
/offences	DONE
/areas	DONE
/ages	DONE
/genders	DONE
/years	DONE
Non-functional Requirement	**States**
https	DONE
Password hashed	DONE

Technical description

Technical	**Description**
NodeJs	JavaScript environment http server
Express	Nodejs web framework for building API
Sequelize	Nodejs promise-based ORM
Bcrypt	Password hashing function for protecting passowords
Helmet	Express HTTPS security header. Increase the security of the express
Json web token (JWT)	RFC 7519 standard authorization token signature signed by private key
Mysql	Open-source relational database management system

Security

Technical	**Description**
JWT Token This system used JWT token for securely transmit information between front-end and backend. It is hashed with HS256 algorithm signed with the secret code. With the JWT token, it can successfully prevent information integrity, even the data with in the JWT token is public to anyone who gets the token.	

Hashed password The photo on the right hand side is the user database table, it shows the email and password. The password in the image is in an hashed format, this format is called bcrypt password hashing with salt password, which has a strong protection against rainbow table attacks.	
Helmet header Helmets provides some various of https header protection for the express sever which prevent DNS prefetching, clickjacking, remove X-Power,Xss protection and others basic security protection	
https Https namely HyperText Transfer Protocol Secure that provide extended security protection for http. It is commonly used nowadays to keep information communication securely.	

Testing and limitation

For testing the API, the server side has implemented swaggerJS in order to try out the api result manually.The testing swagger testing link is: https://172.22.30.8:8443/

Testing Scenario: Register

Test Case	Request	Response	Result

Register with a new user email	**Request URL:**
	https://172.22.30.8:8443/register
	
	Request body:
	{
	"email":"newUsesr@gmail.com",
	"password":"demouserpassword"
	}
		Code status:
		201
		
		Response body:
		{
		"message": "yay! you've successfully registered your user account :)"
		}
			CORRECT

Register a existing user email	**Request URL:**
	https://172.22.30.8:8443/register
	
	Request body:
	{
	"email":"newUsesr@gmail.com",
	"password":"demouserpassword"
	}
		Code status:
		400
		
		Response body:
		{
		"message": "oops! It looks like that user already exists :("
			CORRECT

		}	

esting Scenario: Login

Test Case	Request	Response	Result

Login with an existing user email	**Request URL:**
	https://172.22.30.8:8443/login
	
	Request body:
	{
	"email":"newUsesr@gmail.com",
	"password":"demouserpassword"
	}
		Code status:
		200
		
		Response body:
		{
		"token":
		"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImlkIjo2LCJlbWFpbCI6Im5ld1VzZXNyQGdtYWlsLmNvbSJ9LCJpYXQiOjE1NTkyMTI2MTksImV4cCI6MTU1OTI5OTAxOX0.itV5MMWB9H0Fxtn7NQKFobBbxue7PPsrqCnZ5uF41Gs",
		"access_token":
		"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImlkIjo2LCJlbWFpbCI6Im5ld1VzZXNyQGdtYWlsLmNvbSJ9LCJpYXQiOjE1NTkyMTI2MTksImV4cCI6MTU1OTI5OTAxOX0.itV5MMWB9H0Fxtn7NQKFobBbxue7PPsrqCnZ5uF41Gs",
		"token_type": "Bearer",
		"expires_in": 86400
		}
			CORRECT

Login with Wrong user detail	**Request URL:**
	https://172.22.30.8:8443/login
	
	Request body:
	{
	"email":"newUsesr@gmail.com",
	"password":"wrongpassword"
	}
		Code status:
		401
		
		Response body:
		{
		"message": "invalid login - bad password"
		}
			CORRECT

Testing Scenario : Help

Test Case	Request	Response	Result

Get all offences	**Request URL:**
	https://172.22.30.8:8443/areas
		Code status:
		200
		Response body:
		{
		"offences": [
		"Advertising Prostitution",
		"Armed Robbery",
		"Arson",
		"Assault",
		"Attempted Murder",
		...
		]
		}
			CORRECT

Get all Areas	**Request URL:** https://172.22.30.8:8443/areas	**Code status:** 200 **Response body:** { "areas": ["Aurukun Shire Council", "Balonne Shire Council", "Banana Shire Council", "Barcaldine Regional Council",	CORRECT

		"Barcoo Shire Council", ...] }	

Get all ages	**Request URL:**
	https://172.22.30.8:8443/ages
		Code status:
		200
		
		Response body:
		{
		"ages": [
		"Adult",
		"Juvenile"
		]
		}
			CORRECT

Get all genders	**Request URL:**
	https://172.22.30.8:8443/genders
		Code status:
		200
		Response body:
		{
		"genders": [
		"Female",
		"Male",
		"Not stated",
		]
		}
			CORRECT

Get all years	**Request URL:**
	https://172.22.30.8:8443/years
		Code status:
		200
		Response body:
		{
		"years": [
		2001,
		2002,
		...
		]
		}
			CORRECT

Testing Scenario : Search

Test Case	Request	Response	Result

search with validate JWT token	**Request URL:**
	https://172.22.30.8:8443/search?offence=Homicide(Murder)&area=Banana%20Shire%20Council&age=Adult&gender=male&year=2001&month=2
		Code status:
		200
		
		Response body:
		{
		"query": {
		"offence": "Homicide (Murder)",
		"gender": "male",
		"year": "2001",
		"age": "Adult",
		"area": "Banana%20Shire%20Council",
		},
		"result": [
		{
		"LGA": "Aurukun Shire Council",
		"total": 4,
		"lat": -27.470812,
		"lng": 153.022455
		}
		...
			CORRECT

		] }	

search without offence query parm	**Request URL:**
	https://172.22.30.8:8443/search
		Code status:
		400
		
		Response body:
		{
		"message": "oops! it looks like you're missing the offence query parm"
		}
			CORRECT

search with invalidate JWT token	**Request URL:**
	https://172.22.30.8:8443/search?offence=Homicide(Murder)&area=Banana%20Shire%20Council&age=Adult&gender=male&year=2001&month=2
		Code status:
		401
		
		Response body:
		{
		"message": "oh no! it looks like your authorization token is invalid..."
		}
			CORRECT

References

Technical	**url**
NodeJs	https://nodejs.org/en/
Express	https://expressjs.com/
Sequelize	https://www.npmjs.com/package/sequelize
Bcrypt	https://www.npmjs.com/package/bcrypt
Helmet	https://helmetjs.github.io/
Json web token (JWT)	https://jwt.io/
Mysql	https://www.mysql.com/

Appendix

Instruction for running the application

1. npm install
2. npm run start