**PROJECT TOKO KOMPUTER**

**(Authentication dan Authorization)**

**AUTH**

Pada modul sebelumnya, kita telah mempelajari tentang pembuatan endpoint untuk akses, menambah, mengubah, dan menghapus data pada *database (data resource)*. *Data Resource* merupakan sesuatu yang harus diamankan karena berpotensi adanya pencurian atau penyalahgunaan data oleh pihak yang tidak dikenal. Oleh karena itu, tidak sembarang orang dapat menggunakan endpoint yang kita buat. Endpoint hanya dapat digunakan oleh orang-orang yang diizinkan oleh sistem. Dalam literasi IT terdapat istilah *Authentication* dan *Authorization*. ***Authentication*** adalah proses mengenal user yang akan masuk pada sebuah sistem. ***Authorization*** adalah proses pemberian izin akses user terhadap sistem.

Pada konteks pemanfaatan REST API ini, proses *authentication* dilakukan dengan memberikan kode token kepada user yang telah terverifikasi oleh sistem melalui proses login. Jika user sudah mendapatkan kode token, user dapat memanfaatkan endpoint yang sudah disediakan. Namun setiap kali user melakukan *request* pada suatu endpoint, akan terjadi proses *authorization* terlebih dahulu dengan memvalidasi kode token yang ada. Jika kode token valid, maka user akan mendapatkan *response* sesuai dengan karakteristik endpoint yang dibuat.

## Persiapan Library

1. Install library yang dibutuhkan yaitu multer dengan command

**npm i --save jsonwebtoken**

jsonwebtoken (JWT) adalah sebuah token berbentuk string panjang yang sangat random yang gunanya sendiri untuk melakukan sistem Autentikasi dan Pertukaran Informasi. Umumnya untuk melakukan login tidak seperti pada aplikasi website biasa dimana kita menggunakan session untuk mengingat siapa yang sedang Login. Tapi didalam API sendiri kita menggunakan konsep JWT atau dibacanya sebagai "jot" . Website resminya ada di jwt.io

1. Buat file **auth.js** pada project utama lokasi index.js berada
2. Tambahkan authorization code pada file auth.js seperti berikut

| const jwt = require("jsonwebtoken")  const SECRET\_KEY = "BelajarNodeJSItuMenyengankan"  auth = (req, res, next) => {  let header = req.headers.authorization  let token = header && header.split(" ")[1]  let jwtHeader = {  algorithm: "HS256"  }  if(token == null){  res.status(401).json({ message: "Unauthorized"})  }else{  jwt.verify(token, SECRET\_KEY, jwtHeader, (error,user) => {  if (error) {  res  .status(401)  .json({  message: "Invalid token"  })  } else {  console.log(user);  next()  }  })  }  }  module.exports = auth |
| --- |

1. Pada **customer.js** dan **admin.js** tambahkan import auth seperti berikut

| //import auth  const auth = require("../auth")  const jwt = require("jsonwebtoken")  const SECRET\_KEY = "BelajarNodeJSItuMenyengankan" |
| --- |

1. Tambahkan endpoint untuk login, pada customer.js

| app.post("/auth", async (req,res) => {  let data= {  username: req.body.username,  password: md5(req.body.password)  }  let result = await customer.findOne({where: data})  if(result){  let payload = JSON.stringify(result)  // generate token  let token = jwt.sign(payload, SECRET\_KEY)  res.json({  logged: true,  data: result,  token: token  })  }else{  res.json({  logged: false,  message: "Invalid username or password"  })  }  }) |
| --- |

1. Tambahkan endpoint untuk login pada admin.js

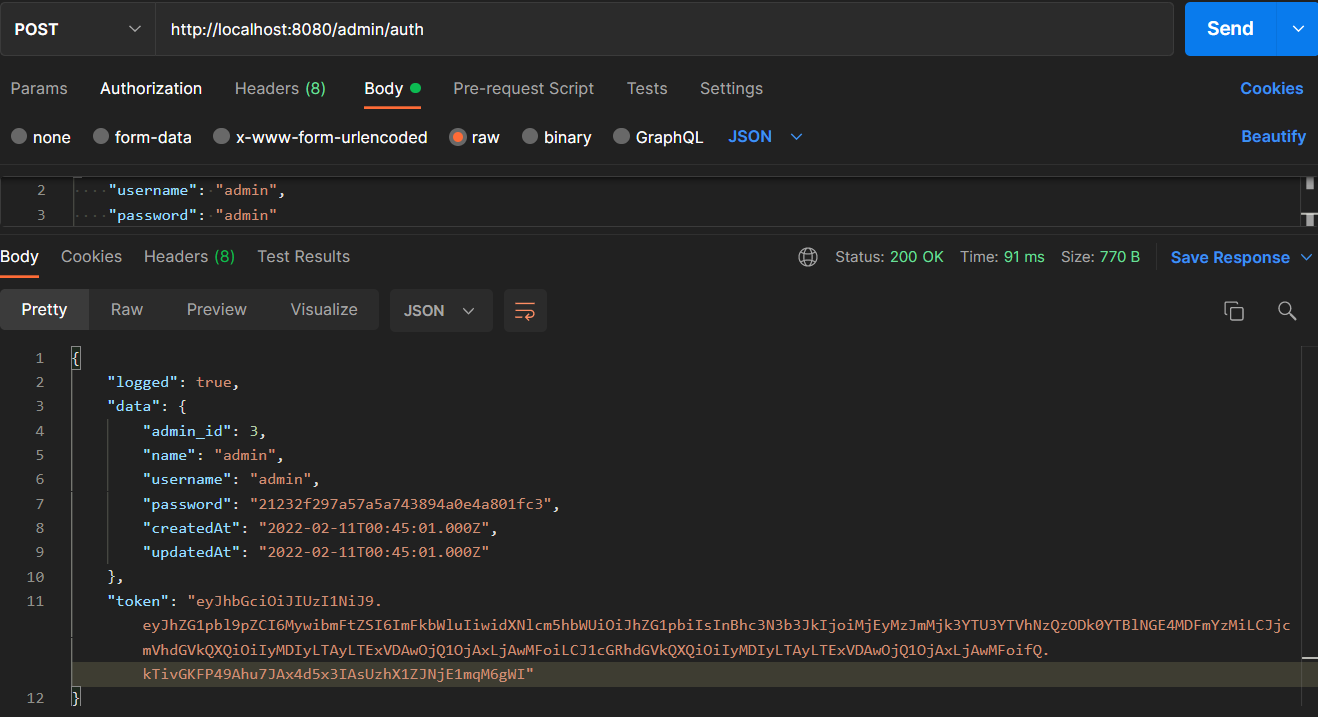
| app.post("/auth", async (req,res) => {  let params = {  username: req.body.username,  password: md5(req.body.password)  }  let result = await admin.findOne({where: params})  if(result){  let payload = JSON.stringify(result)  // generate token  let token = jwt.sign(payload, SECRET\_KEY)  res.json({  logged: true,  data: result,  token: token  })  }else{  res.json({  logged: false,  message: "Invalid username or password"  })  }  }) |
| --- |

1. Tambahkan auth pada endpoint yang membutuhkan authorization atau login, contohnya seperti berikut (NOTE: tambahkan juga pada endpoint lain yang membutuhkan login)

| //endpoint menampilkan semua data admin, method: GET, function: findAll()  app.get("/", **auth**, (req,res) => {  admin.findAll()  .then(result => {  res.json({  admin : result  })  })  .catch(error => {  res.json({  message: error.message  })  })  }) |
| --- |

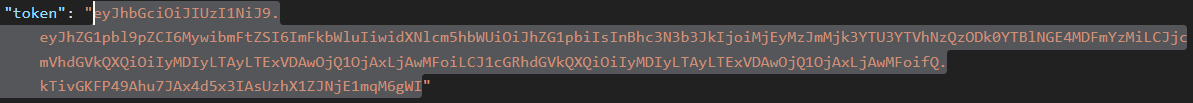
1. Jalankan dan Test hasil API pada postman

LOGIN ADMIN

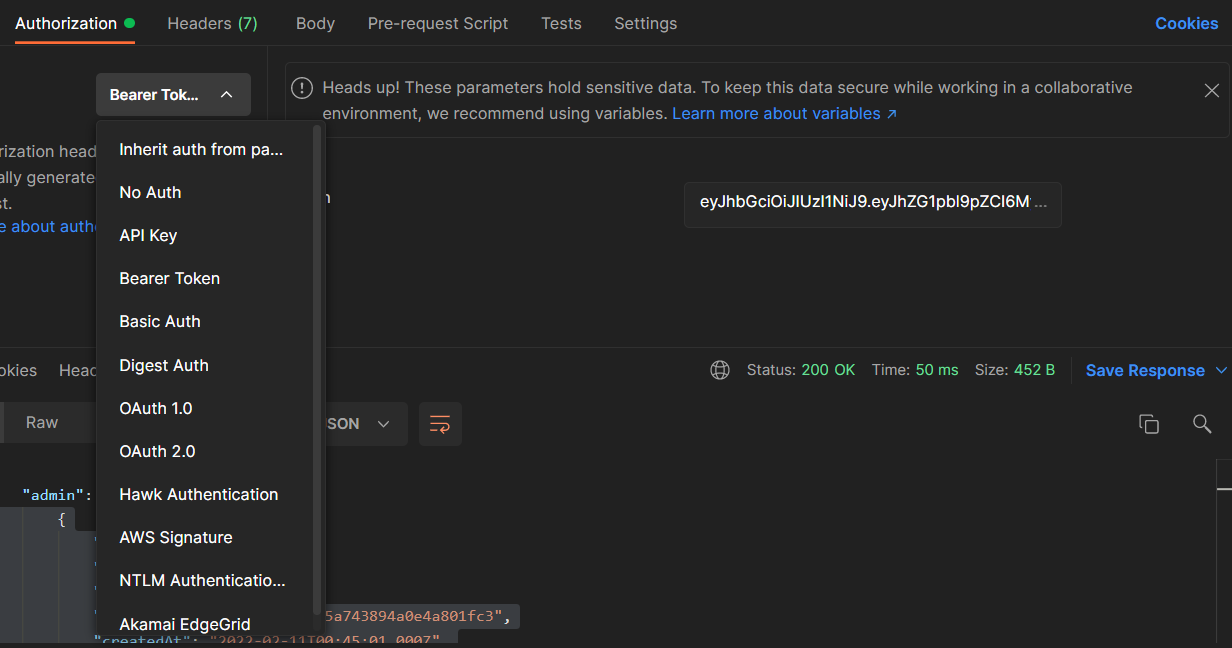


**AUTHORIZATION**

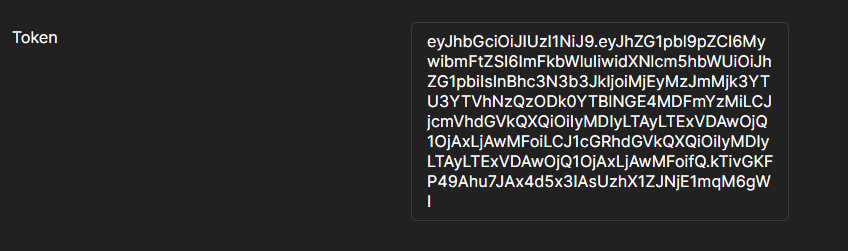
1. Copy token hasil login seperti berikut (TANPA TANDA KURUNG)



1. Pada Authorization, pilih Bearer Token

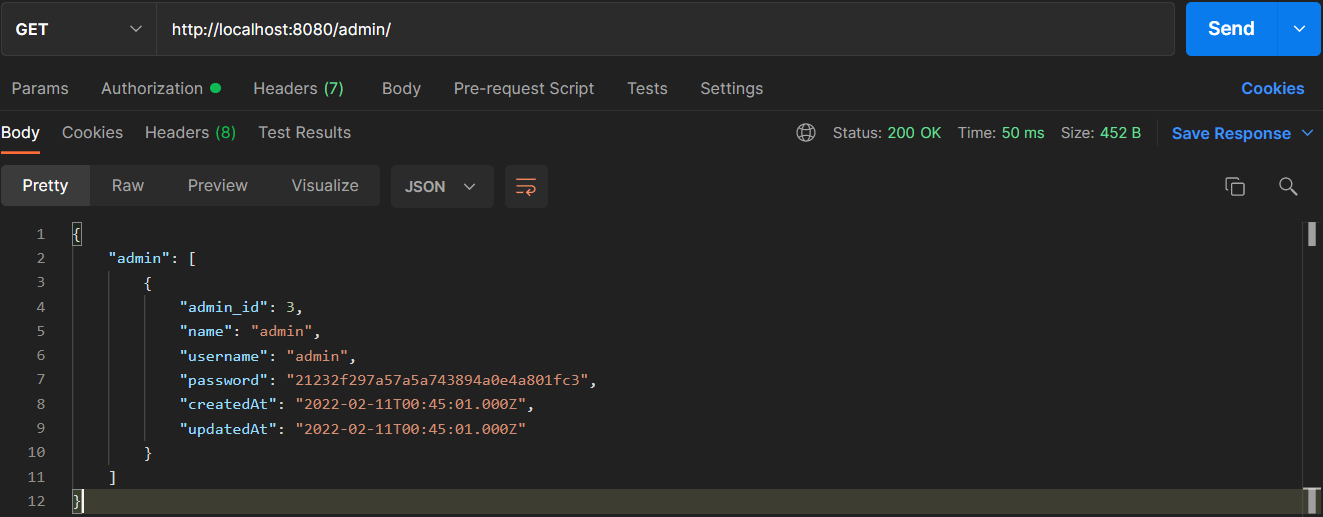


1. Kemudian, paste token pada kotak seperti berikut

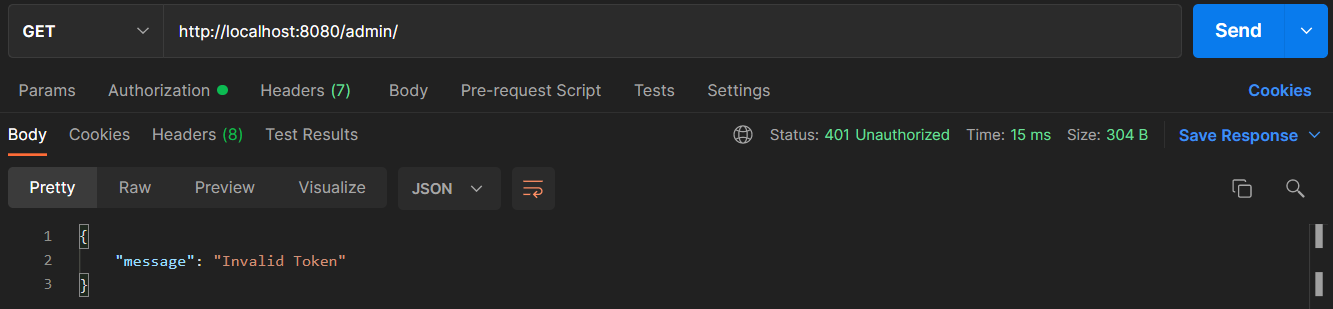


1. Jalankan API

HASIL Jika token benar



HASIL jika token salah



HASIL jika token tidak ditambahkan

