Backend 4.4_CW Exercises

ex01: which of the following APIs *will require authentication?*

challenge

1. user signup API
2. user login API
3. changing password API
4. Forgot password API
5. updating Profile Picture API
6. updating Contact Details API
7. finding User by Phone Number API
8. adding Rating and Review API
9. get movie Reviews with User Details API

solution

- user signup API
- user login API
- changing password API
- updating Profile Picture API
- updating Contact Details API
- adding Rating and Review API

ex02: refactor - authentication routes

You can take your final solution BE2.7_CW or use the following repl to refactor the code.

https://replit.com/@tanaypratap/BE27CW-ex08

challenge

Create a routes folder and move the login and signup route in the `auth.router.js` file using express router and then include this router in your main application.

1. Create a Routes Folder: Begin by creating a `routes` folder in your project directory if it doesn't already exist. This folder will contain your route files.
2. Create an `auth.router.js` File: Inside the `routes` folder, create a file named `auth.router.js`. This file will handle authentication-related routes.
3. Move Routes to `auth.router.js`:
    - Copy the login and signup routes from your main application file (`index.js`) and paste them into `auth.router.js`.
    - Make sure you import `**User**` model in the file.
    - In `auth.router.js`, use Express Router to define these routes.
4. Include `auth.router.js` in your main application.

solution

```
const auth = require('./routes/auth.router')

app.use('/auth', auth)
```

COPY

ex03: refactor - user details routes

challenge

Create a `**users.router.js**` file. You can move the change password, update profile picture, update contact details, find user by phone number routes to the user router. Include this router in your main application.

solution

```
const users = require('./routes/users.router')

app.use('/user-details', users)
```

COPY

ex04: refactor - movies related routes

challenge

Create a `**movies.router.js**` file. You can move

the `**add rating and review**` and `**get movie reviews with user details**` route to the movies

router. Include this router in your main application.

solution

```
const movies = require('./routes/movies.router')

app.use('/movies', movies)
```

COPY

ex05: include authVerify middleware

challenge

Create a middlewares folder. Include the file **auth-verify.middleware.js** and add the code you wrote for authVerify middleware.

solution

```
const jwt = require('jsonwebtoken')
const JWT_SECRET = process.env['jwt-secret']

function verifyToken(token) {
  try {
    const decoded = jwt.verify(token, secret)
    return decoded
  } catch (error) {
    throw new Error('Invalid token')
  }
}

function extractUserIDFromToken(decodedToken) {
  if (decodedToken && decodedToken.userID) {
    return decodedToken.userID
  } else {
    throw new Error('Invalid or missing user ID in token')
  }
}

function authVerify(req, res, next) {
  const token = req.headers.authorization

  try {
    const decoded = verifyToken(token)
    const userID = extractUserIDFromToken(decoded)
    req.user = { userID }
    return next()
  } catch (error) {
    return res
      .status(401)
      .json({ message: 'Unauthorised access, please add the token' })
  }
}
```

```
module.exports = { authVerify }
```

COPY

ex06: include routeNotFound & errorHandler middleware

challenge

Create **error-handler.middleware.js** and **route-not-found.middleware.js** similarly to above include the routeNotFound & errorHandler middlewares in your application.

solution

```
const { errorHandler } = require('./middlewares/error-handler.middleware')
const { routeNotFound } = require('./middlewares/route-not-found.middleware')

app.use(routeNotFound)
app.use(errorHandler)
```

COPY

ex07: practice - generate jwt token in signup route

challenge

You have an existing signup route that registers users in your application. Your challenge is to enhance the signup route by generating a JSON Web Token (JWT) and including it in the response when a user successfully signs up.

solution

```
router.post('/signup', async (req, res) => {
  try {
    const savedUser = await signup(req.body)
    const token = jwt.sign({ userId: savedUser._id }, JWT_SECRET, {
      expiresIn: '24h',
    })
    res.json({
      user: savedUser,
      token,
      success: true,
      message: 'Sign Up Successful',
    })
  } catch (error) {
    res.status(500).json({ error: 'Failed to create user account' })
  }
})
```

COPY

ex08: practice - generate jwt token in login route

challenge

You have an existing login route in your application that authenticates users based on their credentials. Your challenge is to enhance the login route by generating a JSON Web Token (JWT) and including it in the response when a user successfully logs in.

solution

```
router.post('/login', async (req, res) => {
  try {
    const { email, password } = req.body
    const userDetails = await login(email, password)
    const token = jwt.sign({ userId: userDetails._id }, JWT_SECRET, {
      expiresIn: '24h',
    })
    res.json({
      data: { user: userDetails, token },
      success: true,
      message: 'Login Successful',
    })
  } catch (error) {
    res.status(401).json({ error: 'Invalid credentials' })
  }
})
```

COPY

ex09: include authentication to user details routes

challenge

You have user details routes in your application that provide information about specific users. To enhance security and ensure that only authenticated users can access these routes, your challenge is to include authentication middleware for these routes.

solution

```
const { authVerify } = require('./middlewares/auth-verify.middleware')

app.use('/user-details', authVerify, users)
```

COPY

ex10: include authentication to `**adding Rating and Review**` API route

challenge

You have an API route in your application that allows users to add ratings and reviews to movies. To enhance security and ensure that only authenticated users can submit ratings and reviews, your challenge is to include authentication middleware for this API route.

solution

```
const { authVerify } = require('../middlewares/auth-verify.middleware')

router.post('/movies/:movieId/rating', authVerify, async (req, res) => {
  try {
    const movieId = req.params.movieId
    const { userId, rating, review } = req.body

    const updatedMovie = await addRatingAndReview(
      movieId,
      userId,
      rating,
      review,
    )
    res.json(updatedMovie)
  } catch (error) {
    res.status(404).json({ error: 'Movie not found' })
  }
})
```

COPY

final solution

https://replit.com/@tanaypratap/BE44CW