Backend 1.4_CW Exercises

Introducing the User Model

In this lesson, we'll dive into the concept of User Models and explore how to implement various user-related functionalities using Mongoose. Let's begin by defining the User Model with additional fields such as profile picture URL, username, and more.

Step 1: Define a User Model

Let's start by defining the User Model with fields like email, password, profile picture URL, username, and other relevant details.

https://replit.com/@tanaypratap/BE14CW-ex01

```
const mongoose = require('mongoose')

const userSchema = new mongoose.Schema(
  {
    email: {
      type: String,
      required: true,
      unique: true,
    },
    password: {
      type: String,
      required: true,
    },
    profilePictureUrl: String,
    username: { type: String, required: true, unique: true },
    nickname: String,
    // Other fields can be added here
  },
  {
    timestamps: true,
  },
)

const User = mongoose.model('User', userSchema)

module.exports = User
```

COPY

Step 2: Create a Function for User Signup

Now, let's implement the signup functionality. Create a function `signup` that accepts an object containing user details and creates a new user in the database.

- Solution https://replit.com/@tanaypratap/BE14CW-ex02

```
const User = require('./models/user')

async function signup(userDetails) {
  try {
    const user = new User(userDetails)
    const newUser = await user.save()
    console.log('New user created:', newUser)
  } catch (error) {
    throw error
  }
}

// Example usage of the signup function
signup({
  email: 'example@example.com',
  password: 'password123',
  profilePictureUrl: 'https://example.com/profile.jpg',
  username: 'exampleuser',
  nickname: 'Example Nick',
})
```

COPY

Step 3: Create a Function for User Login

Create a function `login` that accepts email and password, verifies the credentials, and returns user details upon successful login.

- Get email and password from user - function paramaters
- Need to get email and password from database - findOne
- Make sure both matches. if yes, then login, if no, then bye bye. - if else logic
- Solution https://replit.com/@tanaypratap/BE14CW-ex03

```
async function login(email, password) {
  try {
    const user = await User.findOne({ email })
    if (user && user.password === password) {
      console.log('Logged in user:', user)
    } else {
      throw new Error('Invalid credentials')
    }
  } catch (error) {
    throw error
  }
}

// Example usage of the login function
try {
  login('example@example.com', 'password123')
} catch (error) {
  console.error('Login failed:', error.message)
```

```
}
```

COPY

Step 4: Create a Function to Change Password

Create a function `changePassword` that accepts a user's email, current password, and new password. Update the password if the current password matches.

- Solution https://replit.com/@tanaypratap/BE14CW-ex04

```
async function changePassword(email, currentPassword, newPassword) {
  try {
    const user = await User.findOne({ email })
    if (user && user.password === currentPassword) {
      user.password = newPassword
      const updatedUser = await user.save()
      console.log('Password changed for user:', updatedUser)
    } else {
      throw new Error('Invalid credentials')
    }
  } catch (error) {
    throw error
  }
}

// Example usage of the changePassword function
try {
  changePassword('example@example.com', 'password123', 'newpassword456')
} catch (error) {
  console.error('Password change failed:', error.message)
}
```

COPY

Step 5: Create a Function to Update Profile Picture

Create a function `updateProfilePicture` that accepts a user's email and the new profile picture URL. Update the profile picture URL for the user.

- Solution https://replit.com/@tanaypratap/BE14CW-ex05

```
async function updateProfilePicture(email, newProfilePictureUrl) {
  try {
    const user = await User.findOne({ email })
    if (user) {
      user.profilePictureUrl = newProfilePictureUrl
      const updatedUser = await user.save()
      console.log('Profile picture updated for user:', updatedUser)
    } else {
      throw new Error('User not found')
    }
  } catch (error) {
    throw error
  }
}
```

```
// Example usage of the updateProfilePicture function
try {
  updateProfilePicture(
    'example@example.com',
    'https://example.com/new-profile.jpg',
  )
} catch (error) {
  console.error('Profile picture update failed:', error.message)
}
```

COPY

Step 6: Create a Function to Update Contact Details

Create a function `updateContactDetails` that accepts a user's email and an object containing updated contact details. Update the contact details for the user. Make sure to add the phoneNumber (Number) key in the User model first.

- Solution https://replit.com/@tanaypratap/BE14CW-ex06

```
async function updateContactDetails(email, updatedContactDetails) {
  try {
    const user = await User.findOne({ email })
    if (user) {
      Object.assign(user, updatedContactDetails)
      const updatedUser = await user.save()
      console.log('Contact details updated for user:', updatedUser)
    } else {
      throw new Error('User not found')
    }
  } catch (error) {
    throw error
  }
}

// Example usage of the updateContactDetails function
try {
  updateContactDetails('example@example.com', {
    email: 'new@example.com',
    phoneNumber: 9876543210,
  })
} catch (error) {
  console.error('Contact details update failed:', error.message)
}
```

COPY

Step 7: Create a Function to Find User by Phone Number

Create a function `findUserByPhoneNumber` that accepts a phone number and retrieves the user associated with that phone number.

- Solution https://replit.com/@tanaypratap/BE14CW-ex07

```
async function findUserByPhoneNumber(phoneNumber) {
```

```
  try {
    const userByPhoneNumber = await User.findOne({ phoneNumber })
    if (userByPhoneNumber) {
      console.log('User found by phone number:', userByPhoneNumber)
    } else {
      console.log('User not found.')
    }
  } catch (error) {
    throw error
  }
}

// Example usage of the findUserByPhoneNumber function
findUserByPhoneNumber('9876543210')
```