1. files upload
2. secure mails – gmail
3. redux
4. rrd v5
5. useRef
6. connect nodejs with react
7. register
   1. with picture
   2. confirm email
8. 2 step auth
   1. email + password
   2. 8 chars
9. forget password
10. send email – contact us
11. files upload

react:

vdom - https://reactjs.org/docs/faq-internals.html

when use what

* url: /api/something?key1=value1&key2=value2
  + method: all
  + req.queryparams
* url: /api/something/:data1/:data2
  + method: all
  + the order of the data must be match by the order of the url
  + req.params
* url: /api/something
  + method: all except get
  + req.body

nodejs:

* how to start prod project: npm start
* how to start a dev project: npm run dev

react:

* how to start prod project: npm run build
* how to start a dev project: npm start

Register - nodejs:

1. create route = “/register”
2. create register file
3. import all modules
4. validate user input
   1. if failed
   2. response errors from joy with status 400
5. find user by email to check if email exists
   1. if exists
      1. response errors “the user already exists” with status 400
   2. else (not exists)
      1. encrypt the password
      2. store user data with encrypted password to db
      3. (Do not store the plain text password only the encrypted password)
      4. send confirmation email
         1. generate random number
         2. create url and send it to mail
      5. send “user created successfully” with status 201
6. when user access the url then update db

Confirm register:

1. create route /api/:email/:secretkey
2. user input validation
3. find user by email to check if email exists
   1. if not exists
      1. response error with status 400
   2. else – the email exists
      1. check secretkey
         1. if they no match
            1. response error with status 400
         2. else – they match
            1. update db
            2. response msg ok with status 200

Login – nodejs:

1. create route = “/login”
2. create login file
3. validate user input
   1. if failed
   2. response errors from joy with status 400
4. find user by email to check if email exists
   1. if not exists
      1. response error “invalid email/password” with status 400
   2. if exists (else)
      1. decode password and compare
         1. if not
            1. response error “invalid email/password” with status 400
         2. else (password correct)
            1. check if email validated

if so then

create token

response token

* + - * 1. else – email is invalid

response error “invalid email/password” with status 400

Forget password:

1. create route /api/forgetpassword
2. create forgetPassword file
3. validate user input
   1. if failed send error with status 400
4. find user by email
   1. if we did not find the email
      1. response: if your email exists you will receive an email
   2. else – we found the email
      1. generate link
         1. /api/resetpassword/:token
         2. create link timeout – 15 min
         3. (Email and secret key should be encrypted)
         4. response: if your email exists you will receive an email

how to reset password:

1. r: forget password
2. n: forget password
3. n: send link to react
4. r: new password
5. r: send the new password with token to node
6. n: reset password

Reset password:

1. create route /api/resetpassword/:token
2. create resetPassword file
3. validate user input
   1. if failed send error with status 400
4. find user by email
   1. if we did not find the email
      1. response: password changed
   2. else - we found the email
      1. compare secretkey
         1. if not match
            1. response: password changed
         2. else
            1. hash the password
            2. update the new password
            3. response: password changed

React: Redux: Auth

* create auth slice with loggedIn
* create action login and logout
  + login will set loggedIn to true
  + logout will set loggedIn to false

TK Redux:

1. create navbar
2. create page with

* name (input)
* send (btn)

when user input his name and press on btn it will update his name in navbar