EmpowerU Africa

System Design Document

CSCC01-H3

Instructor: Ilir Dema

Collaborators: Dezhi Ren, Jiayu Lu,
Jiazheng Li, Siyang Chen, Yuanqian Fang

Table of Contents:

CRC Cards

signup.js	
none	
Sign up page for users	• signin.js • header.js • footer.js

index.js	
none	
Is the home page of the app	• header.js • footer.js

signin.js	
None	
Sign in page for users	• signup.js • header.js • footer.js

App.js	
None	
Route the user to different react pages	• header.js • footer.js • All other react pages

header.js	
None	
The navigation bar for all pages of the app	• index.js • community.js • learn.js • calendar.js • notifications.js • chat.js • profile.js • settings.js

footer.js	
None	
The footer for all pages of the app	None

db.js	
None	
Contains interactions with the MySQL database	• db-init.js

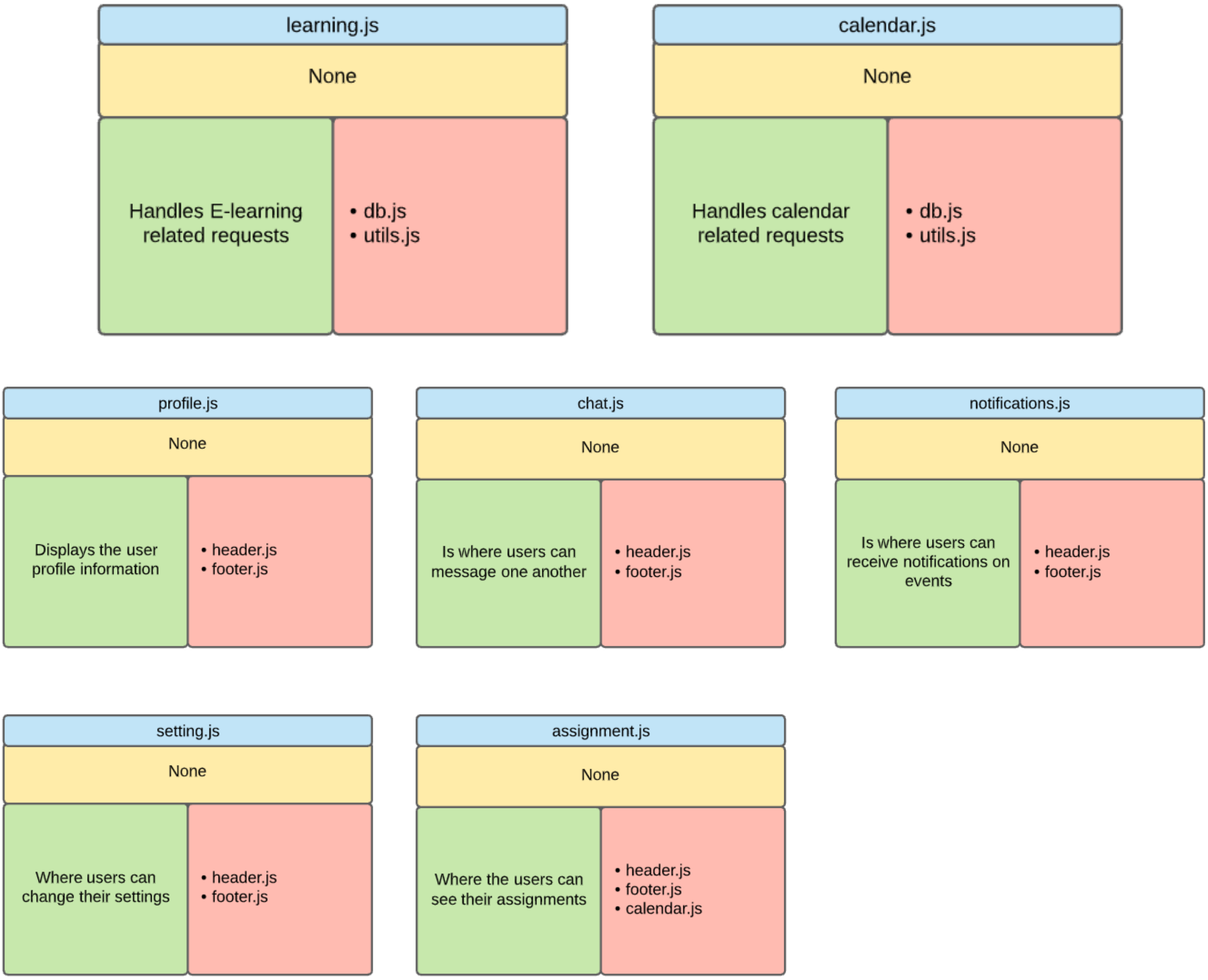
server.js	
None	
Setup the Express.js server	• accounts.js • utils.js • community.js • learning.js • calendar.js

account.js	
None	
Handles account related requests	• db.js • utils.js

utils.js	
None	
Contains utility functions that may be useful accross the whole backend.	None

db-init.js	
None	
initializes the database for db.js, exports an async function that returns a MySQL connection.	None

community.js	
None	
Handles comunity related requests	• db.js • utils.js



learning.js

None

Handles E-learning related requests

- db.js
- utils.js

calendar.js

None

Handles calendar related requests

- db.js
- utils.js

profile.js

None

Displays the user profile information

- header.js
- footer.js

chat.js

None

Is where users can message one another

- header.js
- footer.js

notifications.js

None

Is where users can receive notifications on events

- header.js
- footer.js

setting.js

None

Where users can change their settings

- header.js
- footer.js

assignment.js

None

Where the users can see their assignments

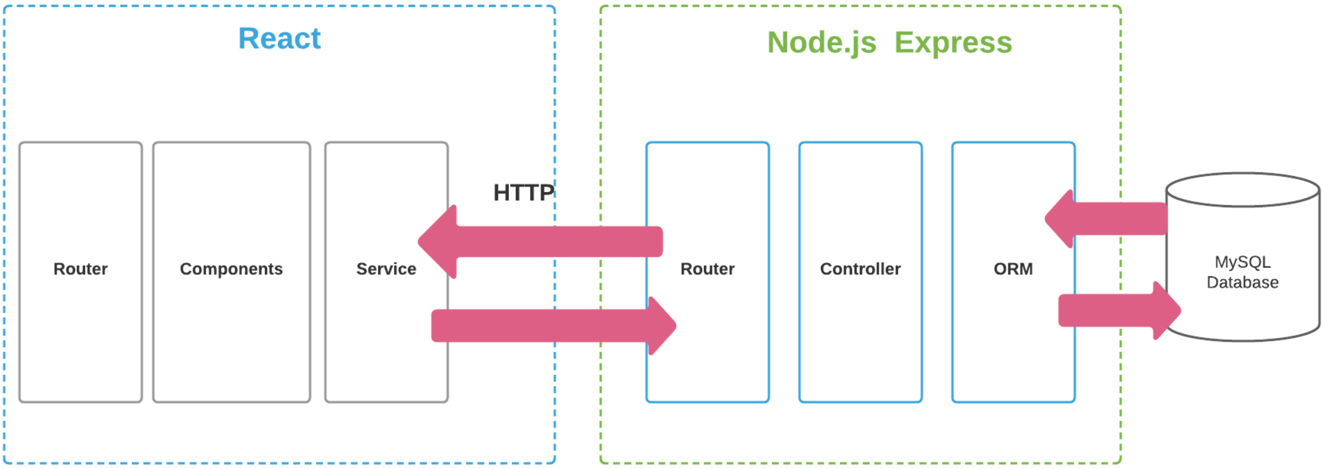
- header.js
- footer.js
- calendar.js

System Interaction Description

Our program assumes the user has node.js(^12.12.0) installed and has npm installed as well. The user should also install MySQL(^8.0.25) and have a MySQL server running. The user must be using either MacOS, Linux or Windows 7+. To view our web app from a remote device, the website must be hosted on a machine with sufficient bandwidth and device storage space, as well as have a domain name for the user to type in. Then the user can just access the website through the DNS.

System Architecture

The architecture that we are using is one of a typical web page. Our front end using React and CSS is connected to the backend using HTTP requests. Our backend is running on Node.js and Express.js. Our backend is connected to the database system using Javascript, and for the database, we are using MySQL. Below is a chart displaying our system architecture.



System Decomposition

The user will access web pages from the internet through our React.js server, which contains ajax requests that will send api requests to our Express.js server. These requests may include creating an account, signing in, requesting top 10 most recent community posts etc. The express server will then call functions to query the MySQL database, sending back the data that the user wanted. For safety precautions, the user is not allowed to directly interact with the MySQL database.

In case of an invalid user input, the web page will first try to stop the user from submitting the request. If the request with invalid user input gets submitted, the express.js server will identify it and stop it from causing damage.