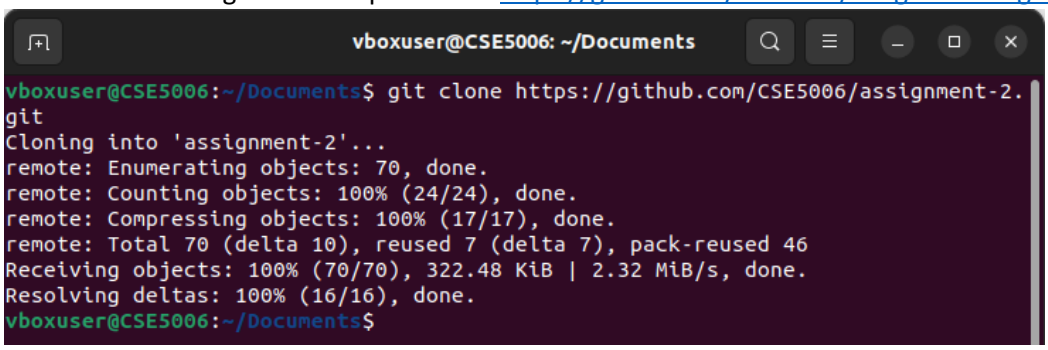
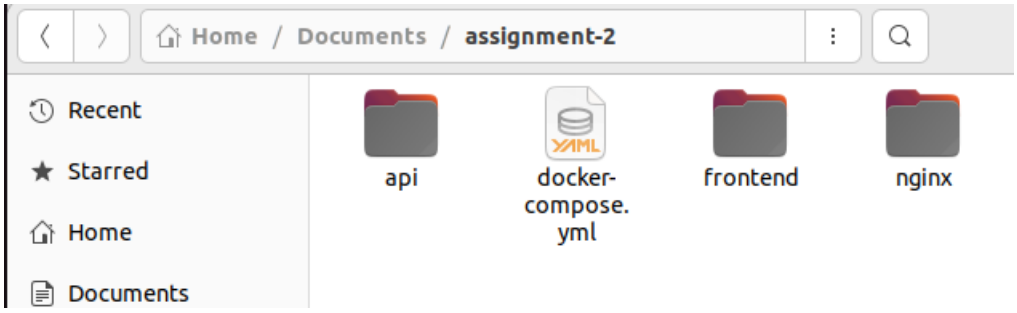
CSE3CWA/CSE5006 Assignment 2 Hint
Sem 2 - 2023

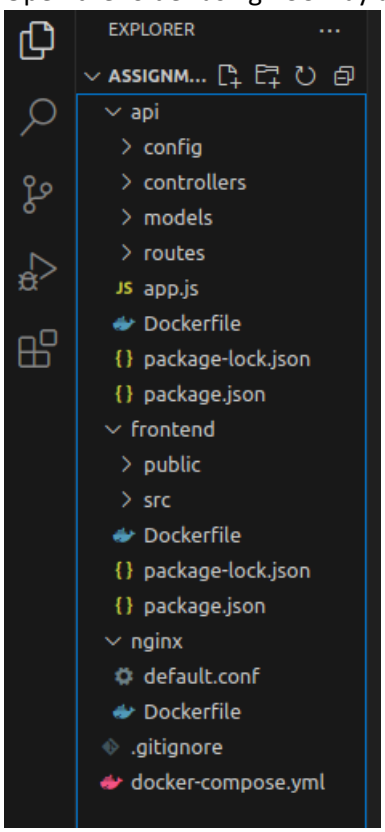
1. Download the assignment template from https://github.com/CSE5006/assignment-2.git



Verify that the repository has been downloaded successfully.

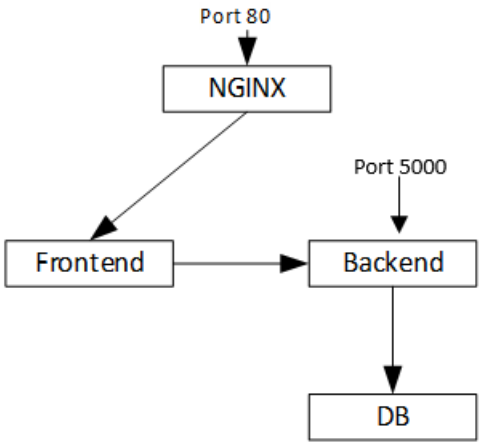


2. Open the folder using VSC. Pay attention to sub-folders



Every sub folder has its own purpose. Please check your labs/tutorials.

3. The docker hasn't been configured properly.
   Refer to the lecture notes Week 7 how to configure the docker properly



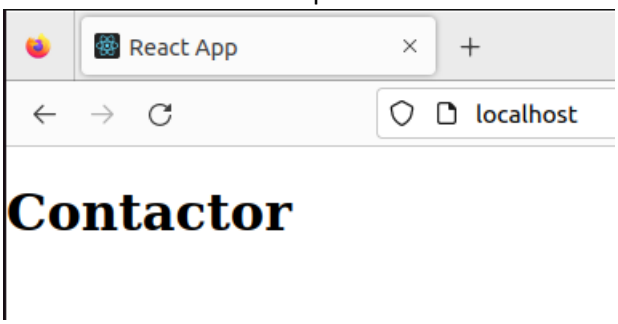
Note: During the development phase, you may open other ports or use different port number. Please make sure you use the right port as specified in the assignment document before submitting your work.

When you have configured the Docker properly, run the docker using this command
$ docker compose up –build

```
vboxuser@CSE5006:~/Documents/assignment-2$ docker compose up --build
 [+] Running 9/1
  ✓ db 8 layers [⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿]      0B/0B      Pulled
 [+] Building 81.9s (19/19) FINISHED
  => [api internal] load build definition from Dockerfile
  => => transferring dockerfile: 192B
  => [api internal] load .dockerignore
  => => transferring context: 2B
  => [frontend internal] load metadata for docker.io/library/node:18.16.1-alpine3.18
  => [frontend 1/3] FROM docker.io/library/node:18.16.1-alpine3.18@sha256:d5b2a7869a4016b1847986ea52
  => => resolve docker.io/library/node:18.16.1-alpine3.18@sha256:d5b2a7869a4016b1847986ea52098fa4044
  => => sha256:bf6c61feabc1a1bd565065016abe77fa378500ec75efa67f5b04e5e5c4d447cd 1.16kB / 1.16kB
  => => sha256:f85482183a4f18ec843fea14e64575a5550f153211c5b497fea50353cb088645 6.73kB / 6.73kB
  => => sha256:31e352740f534f9ad170f75378a84fe453d6156e40700b882d737a8f4a6988a3 3.40MB / 3.40MB
  => => sha256:560412e561fb4693d4bc6e5d197ccdfb42d1453d0eebf0d772baeacdbaab4b63 47.49MB / 47.49MB
```

Make sure you don't have any errors while running the application.

4. Check every component one by one
   a. Frontend
      Launch the browser and open localhost. You should see the following output



   b. Database
      Open the new terminal and run the following script to test if the database has been configured properly or not

$ docker exec -it assignment-2-db-1 bash

$ psql -U $POSTGRES_USER -h localhost -d $POSTGRES_DB

```
vboxuser@CSE5006:~/Documents/assignment-2$ docker exec -it assignment-2-db-1 bash
1db0f900a7b4:/# psql -U $POSTGRES_USER -h localhost -d $POSTGRES_DB
psql (15.3)
Type "help" for help.

postgres=# \dt
            List of relations
 Schema |   Name   | Type  |  Owner
--------+----------+-------+----------
 public | contacts | table | postgres
 public | phones   | table | postgres
(2 rows)

postgres=#
```

As you can see, the tables are available in the database, however these tables only contain “id”.

```
postgres=# \d contacts
                                    Table "public.contacts"
  Column   |           Type           | Collation | Nullable |                Default
-----------+--------------------------+-----------+----------+----------------------------------------
 id        | integer                  |           | not null | nextval('contacts_id_seq'::regclass)
 createdAt | timestamp with time zone |           | not null |
 updatedAt | timestamp with time zone |           | not null |
Indexes:
    "contacts_pkey" PRIMARY KEY, btree (id)

postgres=# \d phones
                                    Table "public.phones"
  Column   |           Type           | Collation | Nullable |                Default
-----------+--------------------------+-----------+----------+-----------------------------------
 id        | integer                  |           | not null | nextval('phones_id_seq'::regclass)
 createdAt | timestamp with time zone |           | not null |
 updatedAt | timestamp with time zone |           | not null |
Indexes:
    "phones_pkey" PRIMARY KEY, btree (id)

postgres=#
```

You have to add additional columns to store the contact information in the following files: api/models/contact.model.js

```
EXPLORER                    contact.model.js
ASSIGNMENT-2                api > models > contact.model.js > <unknown>
  api                       module.exports = (sequelize, Sequelize) => {
    config                      const Contact = sequelize.define("contact", {
      db.config.js                  id: {
    controllers                         type: Sequelize.INTEGER,
      contact.controller.js             autoIncrement: true,
      phone.controller.js               primaryKey: true,
      stats.controller.js           },
    models                          // DEFINE YOUR MODEL HERE
      contact.model.js          });
      index.js
      phone.model.js            return Contact;
    node_modules            };
```

Hint: You must add the contact name column here

Api/models/phone.model.js

```
EXPLORER
ASSIGNMENT-2
  api
    config
      db.config.js
    controllers
      contact.controller.js
      phone.controller.js
      stats.controller.js
    models
      contact.model.js
      index.js
      phone.model.js
    node_modules
```

phone.model.js

api > models > phone.model.js > <unknown>

```
module.exports = (sequelize, Sequelize) => {
    const Phone = sequelize.define("phone", {
        id: {
            type: Sequelize.INTEGER,
            autoIncrement: true,
            primaryKey: true,
        },
        // DEFINE YOUR MODEL HERE
    });

    return Phone;
};
```

You must add the phone type and phone number

The 1-many relationship must be completed in api/models/index.js

```
EXPLORER
ASSIGNMENT-2
  api
    config
      db.config.js
    controllers
      contact.contro... M
      phone.controller.js
      stats.controller.js
    models
      contact.model.js M
      index.js M
      phone.model.js M
    node_modules
    routes
```

index.js M

api > models > index.js > ...

```
const db = {};

db.Sequelize = Sequelize;
db.sequelize = sequelize;

/* Create database tables and models */
db.contacts = require("./contact.model.js")(sequelize, Sequelize);
db.phones = require("./phone.model.js")(sequelize, Sequelize);

//add 1-many relationship
db.phones.belongsTo(db.contacts);
db.contacts.hasMany(db.phones);

module.exports = db;
```

To help you, I have provided the missing code to make the 1-many relationship for these tables.

Note: after you have changed the table structure, make sure you stop all of the containers, remove all of the containers, and the volumes that store the data using the following script:
$ docker system prune -a
$ docker volume prune -a

Make sure you do not have any result when you run the following script
$ docker ps -a
$ docker volume ls

If you still have any results, repeat the prune process. In some cases, you must stop all of the containers manually.
When you have removed all containers and volume, you can rebuild your project again using

$ docker compose up --build

c. Backend

You have to check the router of your API. In this application, the API path is set to 3 handlers as shown in the /api/app.js

```
EXPLORER
ASSIGNM...
  api
    config
    controllers
    models
    node_modules
    routes
    app.js
    Dockerfile
    package-lock.json
    package.json
  frontend
  nginx
  .gitignore
  docker-compose.... M

app.js
api > app.js > ...
1  const express = require("express");
2  const cors = require("cors");
3
4  const app = express();
5
6  var corsOptions = {
7    origin: "http://localhost:3000"
8  };
9
10 const db = require("./models");
11 db.sequelize.sync({ force: false }).then(() => {
12     console.log("Drop and re-sync db.");
13 });
14
15 app.use(cors(corsOptions));
16
17 // parse requests of content-type - application/json
18 app.use(express.json());
19
20 // parse requests of content-type - application/x-www-form-urlencoded
21 app.use(express.urlencoded({ extended: true }));
22
23 // simple route
24 app.get("/", (req, res) => {
25   res.json({ message: "Welcome to bezkoder application." });
26 });
27
28 require("./routes/contacts.routes")(app);
29 require("./routes/phones.routes")(app);
30 require("./routes/stats.routes")(app);
```

For CSE3CWA, you only need to focus on contacts and phones. No need to do the statistics module.

```
EXPLORER
ASSIGNMENT-2
  api
    config
    controllers
    models
    node_modules
    routes
      contacts.routes.js
      phones.routes.js
      stats.routes.js
    app.js
    Dockerfile
    package-lock.json
    package.json
  frontend
  nginx
  .gitignore
  docker-compose.... M

contacts.routes.js
api > routes > contacts.routes.js > ...
4      var router = require("express").Router();
5
6      router.post("/contacts/", contacts.create);
7
8      router.get("/contacts/", contacts.findAll);
9
10     router.get("/contacts/:contactId", contacts.findOne);
11
12     router.put("/contacts/:contactId", contacts.update);
13
14     router.delete("/contacts/:contactId", contacts.delete);
15
16     app.use('/api', router);
17 };
```

The PATHs for your API have been set up in api/routes/*

You don't have to make any changes here

Check the first API handler or controller that has been provided in api/controllers/contact.controller.js

```
EXPLORER
ASSIGNMENT-2
  api
    config
      db.config.js
    controllers
      contact.contro... M
      phone.controller.js
      stats.controller.js
    models
    node_modules
    routes
      contacts.routes.js
      phones.routes.js
      stats.routes.js
    app.js
    Dockerfile
    package-lock.json
    package.json
  frontend
  nginx
  .gitignore
  docker-compose.... M
```

contact.controller.js M

api > controllers > contact.controller.js > ...

```
1  const db = require("../models");
2  const Contacts = db.contacts;
3  const Phones = db.phones;
4  const Op = db.Sequelize.Op;
5
6  // Create contact
7  exports.create = (req, res) => {
8
9  };
10
11 // Get all contacts
12 exports.findAll = (req, res) => {
13     Contacts.findAll()
14         .then(data => {
15             res.send(data);
16         })
17         .catch(err => {
18             res.status(500).send({
19                 message: err.message || "Some error occurred"
20             });
21         });
22 };
23
24 // Get one contact by id
25 exports.findOne = (req, res) => {
26
27 };
```

Please note that the URL to check the API will be in this format:
http GET localhost:5000/api/contacts/

```
vboxuser@CSE5006:~/Documents/assignment-2$ http GET localhost:5000/api/contacts/
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://localhost:3000
Connection: keep-alive
Content-Length: 2
Content-Type: application/json; charset=utf-8
Date: Sun, 01 Oct 2023 13:02:32 GMT
ETag: W/"2-l9Fw4VUO7kr8CvBlt4zaMCqXZ0w"
Keep-Alive: timeout=5
Vary: Origin
X-Powered-By: Express

[]
```

Since our table do not have any data in it, this request will return empty set.

Please note:
Make sure to use the following URL to access your API
localhost:5000/api/<your_path_handler>

5. Working with Frontend
    a. The Root component in this app is stored in frontend/src/index.js

```
EXPLORER
ASSIGNM...
  api
  frontend
    node_modules
    public
    src
      App.css
      App.js
      App.test.js
      index.css
      index.js
      reportWebVitals.js
      setupTests.js
    Dockerfile
    package-lock.json
    package.json
  nginx
  .gitignore
  docker-compose.... M
```

```
index.js
frontend > src > index.js > ...
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

You don't have to do anything here

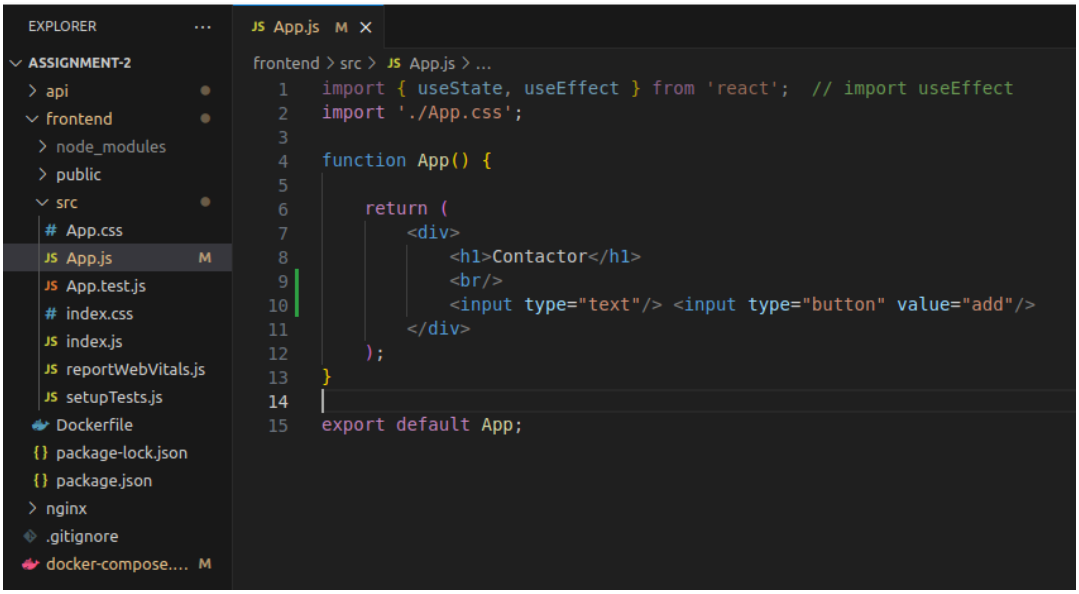
b. Your main React component is stored in frontend/src/App.js

```
EXPLORER
ASSIGNMENT-2
  api
  frontend
    node_modules
    public
    src
      App.css
      App.js
      App.test.js
      index.css
      index.js
      reportWebVitals.js
      setupTests.js
    Dockerfile
    package-lock.json
    package.json
  nginx
  .gitignore
  docker-compose.... M
```

```
App.js
frontend > src > App.js > default
import { useState, useEffect } from 'react';  // import useEffect
import './App.css';

function App() {

    return (
        <div>
            <h1>Contactor</h1>
        </div>
    );
}

export default App;
```

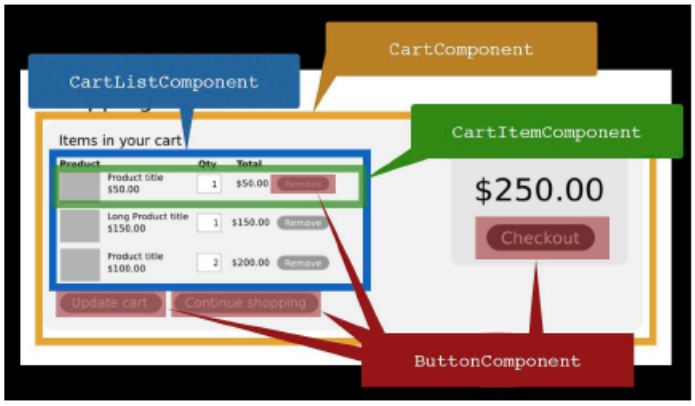
Here, you can start building your components.

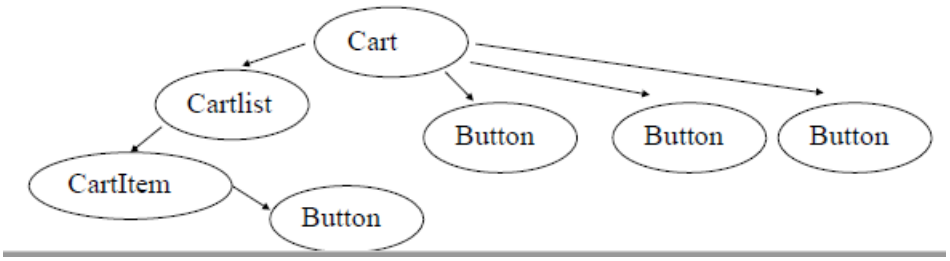
c. You can start working by adding static HTML component until you can mimic the structure of the application. When you are satisfied with your design, then you can start migrating the components to React component.
For example:



d. **You may put your components in different files or put everything in one file as demonstrated in lab. A component is considered a function that will create a group of objects.**

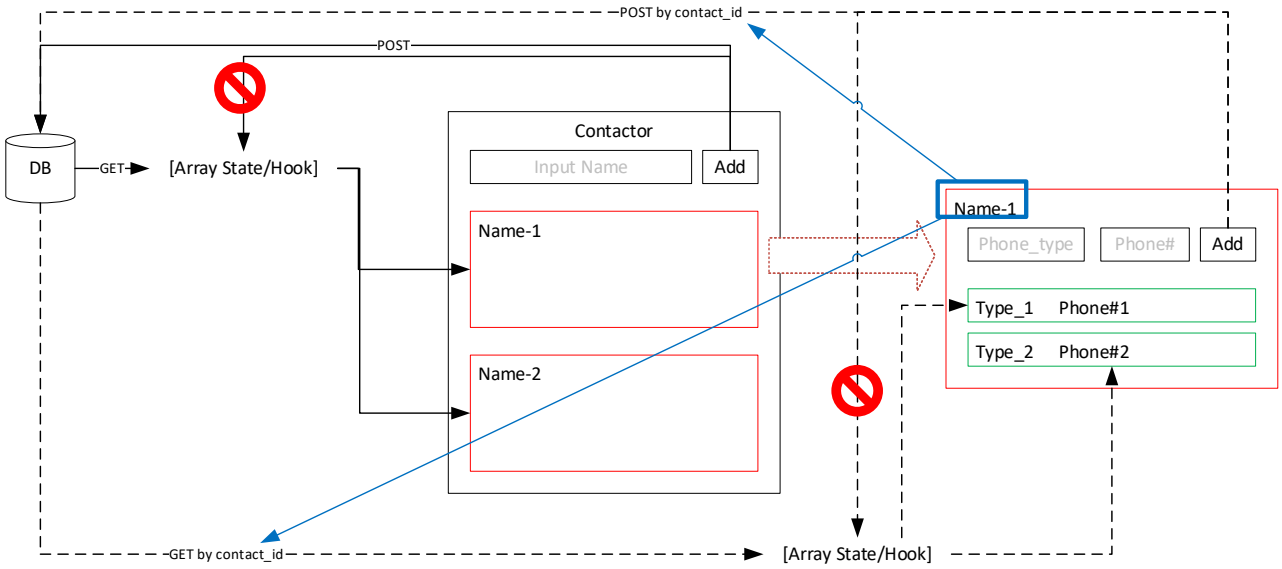
e. Hint: React components must be arranged in a hierarchy manner.
A component refers to a web component or a group of web components.





f. Hint, the add name function is similar to the add task function in Lab 4 and 5. The add phone function is similar to add name function.

g. The overall frontend design can be like this



h. To call an API from a component, you can use the fetch method as shown in Lab 8

```
36          function onClick() {
37              fetch(`http://localhost/api/tasks/${props.id}`, {
38                  method: 'DELETE',
39              })
40              .then(() => {
41                  // remove it from the state
42                  props.setTasks(tasks => tasks.filter(task => task.id !== props.id));
43              })
44              .catch((error) => {
45                  console.error('Error:', error);
46              });
47          }
48
49              return (
50                      <li><button type="button" onClick={onClick}>X</button> { props.description }
51              );
52      }
53
```

i. You may apply any CSS to the component. Do not focus on the appearance of your application as any application of CSS is more than enough.

j. This document might be updated again. Please check the LMS regularly

Notes: Please submit whatever you have. It will be better to submit your work as is on time, rather than late submission or request for the extension. Extension approval will be processed on the case-by-case basis. Without strong and clear evidence, the extension request might not be granted.

Please remember that this assignment only contributes 25% to your final grade. You need time to prepare the biggest contributions for your final grade, which is EXAM.

Any allegations of academic misconduct will be prosecuted seriously. Penalties may vary from mark reduction to suspension.

Check here https://www.latrobe.edu.au/students/admin/academic-integrity/penalties-for-academic-misconduct