# El Backo

## Assignment 1-1 (2/13/2019)

**1 - Create a new react app with**[**Create React App**](https://github.com/facebook/create-react-app)**(also automatically creates the first git commit):**

yarn create react-app el-backo-frontend

**2 - Run the app inside the el-backo-frontend folder (can then access the app at**[**http://localhost:3000**](http://localhost:3000/)**):**

yarn start

**3 - Add**[**React Router**](https://reacttraining.com/react-router/web/guides/quick-start)**package with:**

yarn add react-router-dom

## Assignment 1-2 (2/18/2019)

**Make sure you can run your program.**

yarn start

It should show the React rotating atom.

Make sure to run it from the correct directory   
(should be C:\dev\el-backo-class\step-1\el-backo-frontend\>)

**4 - Configure React Router in App.js with one Index route and add Index component.**

First of all, add this to your import statements:

import { BrowserRouter as Router, Switch, Route } from 'react-router-dom';

Then replace this line in your App.js (this is in the src folder)

export default App;

with this:

export default () => (

<Router>

<Switch>

<Route path='/' exact component={Index} />

</Switch>

</Router>

)

Try running your application (yarn start). It will give you an error. That’s because it needs to know what this “Index” component is.

Start by importing it. Add this to your imports.

import Index from './components/Index';

Now see that the error is “Module not found” and that’s because it can’t find a file in the components directory called “Index.js”

So we need to add that.

Make a directory in src called components

Then make a new file called Index.js and put this content:

import React from 'react'

import { Link } from 'react-router-dom'

export default () => (

<div />

)

## Assignment 1-3 (2/20/2019)

Replace your components/Index.js file with this one.



5 - Add /singleplayer/new route and SingleplayerNew component

In App.js, add this import:

import SingleplayerNew from './components/SingleplayerNew';

In App.js, in Router/Switch area, add this line after the first route:

<Route path='/singleplayer/new' exact component={SingleplayerNew} />

And finally add a new file in the components directory:



Now try yarn start (if it’s not already running) and open to <http://localhost:3000/>

## Assignment 1-4 (2/25/2019)

6 - Add `Link` in `Index` component to `/singleplayer/new` route

Open src/components/Index.js.

Make sure it looks like this:

import React from 'react'

import { Link } from 'react-router-dom'

export default () => (

<div>

<div>

In Index route here

</div>

<Link to='/singleplayer/new'>

Singleplayer

</Link>

</div>

)

7 - Add `SingleplayerPlayerNames` that asks for player names for a new game; also accepts and renders `playersCount` variable from the url

Create src/components/SingleplayerPlayerNames.js and put this as its content:

import React from 'react'

export default ({

match: {

params: {

playersCount,

},

},

}) => (

<div>

This will eventually ask for {playersCount} player names to create a game

</div>

)

Edit src/App.js and make sure it looks like this:

import React from 'react';

import { BrowserRouter as Router, Switch, Route } from 'react-router-dom';

import Index from './components/Index'

import SingleplayerNew from './components/SingleplayerNew'

import SingleplayerPlayerNames from './components/SingleplayerPlayerNames'

export default () => (

<Router>

<Switch>

<Route path='/' exact component={Index} />

<Route path='/singleplayer/new' exact component={SingleplayerNew} />

<Route path='/singleplayer/new/:playersCount' exact component={SingleplayerPlayerNames} />

</Switch>

</Router>

)

See how there’s a new Route to /singleplayer/new/:playersCount

Try yarn start, see if it works!

## Assignment 1-5 (2/27/2019)

8 - Add [lodash](https://lodash.com) that will be needed for utility operations:

yarn add lodash

9 - Add router links for player counts to the `SingleplayerNew` component; map it using lodash

Open src/components/SingleplayerNew.js into Sublime Text.

Make sure that you have this line at the top:

import \_ from 'lodash'

## Assignment 1-4 (3/20/2019)

We currently have just raw menu functionality - selecting "Singleplayer", players count…it’s ugly, right? Let’s introduce some styling next.

Our app is based upon [Create React App](https://github.com/facebook/create-react-app), which encourages the use of [CSS Modules](https://facebook.github.io/create-react-app/docs/adding-a-css-modules-stylesheet) to manage CSS files. Global styles are in src/index.css.

1.1 - To use a brown background color, update the global styles it in src/index.css (or you can choose a different color)

1.2 - Create a shared button style for the game menus

Create src/components/Button/index.js

This is your code for Button/index.js:

import React from 'react'  
import { Link } from 'react-router-dom'  
import styles from './index.module.css'  
  
export default (props) => (  
 <Link className={styles.container} {...props} />  
)

and we need to create src/components/Button/index.module.css to style the button:

.container {

display: block;

width: 100%;

padding: 15px 40px;

border: 2px solid #fff;

color: #fff;

background-color: #25453D;

text-transform: uppercase;

font-weight: 600;

cursor: pointer;

box-sizing: border-box;

font-size: 12px;

text-align: center;

text-decoration: none;

margin: 10px 0;

}

.container:disabled {

background-color: grey;

}

And let’s go back to src/components/Index.js – this file exists, but we need to edit it.

Add this Button import in the “import” area at the top of the file:

import Button from './Button'

then get rid of the <Link> tag and replace it with <Button> :

delete this:

<Link to='/singleplayer/new'>

Singleplayer

</Link>

add this in its place:

<Button to='/singleplayer/new'>

Singleplayer

</Button>

Finally, edit this src/components/SingleplayerNew.js:

Delete this line:

import { Link } from 'react-router-dom'

Import the Button instead:

import Button from './Button'

Then change your tags from Link to Button…from this:

<Link key={n} to={`/singleplayer/new/${n}`}>

{n}

</Link>

to this:

<Button key={n} to={`/singleplayer/new/${n}`}>

{n}

</Button>

1.3 - Make a menu container

Start by editing src/components/Index.js

Add this import line:

import MenuContainer from './MenuContainer'

Remove this import line:

import { Link } from 'react-router-dom'

Then change <div> and </div> to <MenuContainer> and </MenuContainer>

Then create [src/components/MenuContainer/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/089f785#diff-02e7458b3b1464c86d291ba1f2c8221f) as a new file

import React from 'react'

import styles from './index.module.css'

export default ({ children }) => (

<div className={styles.container}>

{children}

</div>

)

Then create [src/components/MenuContainer/index.module.css](https://github.com/Nedomas/el-backo-frontend-steps/commit/089f785#diff-df8721107345e8c837b8dcd24a4df68c) as a new file

.container {

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

text-align: center;

height: 100vh;

max-width: 400px;

margin: 0 auto;

}

Then let’s go into [src/components/SingleplayerNew.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/089f785#diff-0f1be20033d76a23353fb84a4f8de20a) and make some edits:

Add this line to the imports

import MenuContainer from './MenuContainer'

Then change <div> to <MenuContainer>

And finally! Let’s edit [src/components/SingleplayerPlayerNames.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/089f785#diff-ab4b8cb05a16a5bee1d005e509501efc):

Add this line to the imports

import MenuContainer from './MenuContainer'

Then change <div> to <MenuContainer>

## Assignment 1-5 (3/22/2019)

1.4 - Add logo to menu container

Create this file: [src/components/Logo/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/e490c5e#diff-fb6cd2c31257c92b8c3adeac99996489)

import React from 'react'

import styles from './index.module.css'

export default () => (

<div className={styles.container}>

El Backo

</div>

)

Create this file: [src/components/Logo/index.module.css](https://github.com/Nedomas/el-backo-frontend-steps/commit/e490c5e#diff-6e2ba69081c97efe0a5e3bfdfe7cf9e9)

.container {

font-size: 80px;

padding-bottom: 30px;

letter-spacing: 4px;

font-family: 'Luckiest Guy', cursive;

color: #ea3c53;

text-shadow: 2px 0 0 #fff, -2px 0 0 #fff, 0 2px 0 #fff, 0 -2px 0 #fff, 1px 1px #fff, -1px -1px 0 #fff, 1px -1px 0 #fff, -1px 1px 0 #fff;

}

Let’s go back to [src/components/MenuContainer/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/e490c5e#diff-02e7458b3b1464c86d291ba1f2c8221f) and make these changes

Add this to the imports:

import Logo from '../Logo'

Add this inside the <div> tag

<Logo />

Finally, edit [src/components/MenuContainer/index.module.css](https://github.com/Nedomas/el-backo-frontend-steps/commit/e490c5e#diff-df8721107345e8c837b8dcd24a4df68c)

Add this line to the top:

@import url('https://fonts.googleapis.com/css?family=Luckiest+Guy');

1.5 - Style players count select

Delete this file: [src/components/SingleplayerNew.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/2da73c2#diff-0f1be20033d76a23353fb84a4f8de20a)

Create this file: [src/components/SingleplayerNew/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/2da73c2#diff-a1f7dd25008064dcae225f695348e5d9)

import React from 'react'

import \_ from 'lodash'

import Button from '../Button'

import MenuContainer from '../MenuContainer'

import styles from './index.module.css'

export default () => (

<MenuContainer>

<div>

In new singleplayer route here

</div>

<div className={styles.buttons}>

{\_.map(\_.range(2, 7), (n) => (

<Button key={n} to={`/singleplayer/new/${n}`}>

{n}

</Button>

))}

</div>

</MenuContainer>

)

Create this file: [src/components/SingleplayerNew/index.module.css](https://github.com/Nedomas/el-backo-frontend-steps/commit/2da73c2#diff-4f763b93b8604b53d7dbca6230597ef2)

.buttons {

display: flex;

}

2.2.0 – Add formic

We will use [Formik](https://jaredpalmer.com/formik/) for form state management here. Install it by running

yarn add formik

## Assignment 1-6 (3/25/2019)

**2.2.1** - Add player names form

Delete [src/components/SingleplayerPlayerNames.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/ae510e9#diff-ab4b8cb05a16a5bee1d005e509501efc)

Create [src/components/SingleplayerPlayerNames/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/ae510e9#diff-f5aed14fa3b960a574b11829b5bd6e36)

import React from 'react'

import \_ from 'lodash'

import { Formik, Field } from 'formik'

import MenuContainer from '../MenuContainer'

import initialValues from './lib/initialValues'

export default ({

match: {

params: {

playersCount,

},

},

}) => (

<MenuContainer>

<Formik

initialValues={initialValues(playersCount)}

>

{({ values: { players }, handleSubmit, errors }) => (

<form onSubmit={handleSubmit}>

{\_.map(players, ({ number }, index) => (

<div key={index}>

<Field

name={`players.${index}.name`}

placeholder={`Player ${index + 1} name`}

/>

</div>

))}

</form>

)}

</Formik>

</MenuContainer>

)

Create [src/components/SingleplayerPlayerNames/lib/initialValues.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/ae510e9#diff-6d48d3ca642da63cbc3c702369240e2e)

import \_ from 'lodash'

export default (playersCount) => ({

// create an array of players with initial names

players: \_.map(\_.times(playersCount), (n) => ({

name: `Player ${n + 1}`,

})),

})

**2.2.2** - Style player form fields

Add this to the imports in [src/components/SingleplayerPlayerNames/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/197df60#diff-f5aed14fa3b960a574b11829b5bd6e36)

import styles from './index.module.css'

Then add this to the <Field> tag after name and before placeholder

className={styles.field}

Then create [src/components/SingleplayerPlayerNames/index.module.css](https://github.com/Nedomas/el-backo-frontend-steps/commit/197df60#diff-848b2233a45a85672c0d9f881979c4b2)

.field {

display: block;

max-width: 300px;

width: 100%;

padding: 15px;

min-width: 200px;

font-size: 24px;

}

**2.2.3** - Submit the form and console log the inputs

In [src/components/Button/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/cfbe780#diff-bbc6749162635fba28ca5450c9a99830)

Delete these lines

export default (props) => (

<Link className={styles.container} {...props} />

)

Then add these lines

export default ({

component,

...props

}) => {

// allow specifying component to use; default to Link

const Component = component || Link

return <Component className={styles.container} {...props} />

}

Open [src/components/SingleplayerPlayerNames/index.js](https://github.com/Nedomas/el-backo-frontend-steps/commit/cfbe780#diff-f5aed14fa3b960a574b11829b5bd6e36)

Add this to the imports:

import Button from '../Button'

and add this attribute to the <Formik> tag

onSubmit={(variables) => (

console.log('Form submitted with:', variables)

)}

Then add this Button tag right before the </div> at the end of the file

<Button component='button' type='submit'>

Start game

</Button>

## Assignment 1-7 (3/27/2019)

**Time to start setting up the back end! What does the “back end” mean?**

We will use [Apollo Server](https://www.apollographql.com/docs/apollo-server/) as a backend server. It handles all the communication through a GraphQL endpoint.

If anything is unclear for the steps 2.3.1 - 2.3.5, check out the official [guide](https://www.apollographql.com/docs/apollo-server/getting-started.html).

2.3.1 - Create an empty backend directory (outside of the frontend directory) with (make sure to replace XXXX with your name):

C:\dev\el-backo-class\XXXX>mkdir back-end

**2.3.2** - Enter the directory, so the remaining work will take place within that directory:

C:\dev\el-backo-class\XXXX\back-end\>cd graphql-server-example

**2.3.3 -** Initialize the new directory as a Node.js project:

yarn init --yes

If the above steps all completed successfully, there should be a new package.json file in the directory. You can verify this by running

dir

**2.3.4** - Install Apollo Server and GraphQL

yarn add apollo-server graphql

**2.3.5**- Setup a hello world query and start the server with:  
  
Create [index.js](https://github.com/Nedomas/el-backo-backend-steps/commit/e6c18d2#diff-168726dbe96b3ce427e7fedce31bb0bc): (in the back-end\graphql-server-example folder)

This is the content:

const { ApolloServer, gql } = require('apollo-server')

const typeDefs = gql`

type Query {

hello: String

}

`

const resolvers = {

Query: {

hello: () => 'world',

},

}

const server = new ApolloServer({ typeDefs, resolvers })

server.listen().then(({ url }) => {

console.log(`🚀 Server ready at ${url}`)

})

Then run it:

node index.js

**2.3.6 -** Test the hello world query and the server by opening <http://localhost:4000/>

**2.3.7 -** Setup Prisma & Docker by following these steps:

npm install -g prisma

Make sure Docker is (1) running and (2) up to date

Create a “hello world” folder for Docker to play with (to start):

C:\dev\el-backo-class\XXXX\back-end\>mkdir hello-world

C:\dev\el-backo-class\XXXX\back-end\>cd hello-world  
C:\dev\el-backo-class\XXXX\back-end\hello-world\>

Then we create a Docker Compose file.

C:\dev\el-backo-class\XXXX\back-end\hello-world\>notepad docker-compose.yml

This opens Notepad.

Paste this into that file:

version: '3'

services:

prisma:

image: prismagraphql/prisma:1.26

restart: always

ports:

- "4466:4466"

environment:

PRISMA\_CONFIG: |

port: 4466

databases:

default:

connector: postgres

host: postgres

port: 5432

user: prisma

password: prisma

migrations: true

postgres:

image: postgres:10.5

restart: always

environment:

POSTGRES\_USER: prisma

POSTGRES\_PASSWORD: prisma

volumes:

- postgres:/var/lib/postgresql/data

volumes:

postgres:

Now make a folder called [prisma](https://github.com/Nedomas/el-backo-backend-steps/commit/ac5d762cc51dd3775eb0e29e4f483539b5f655ec#diff-3d29739d18d2b4e16305ff3c49cccece) in the back-end folder

Make a file named [datamodel.prisma](https://github.com/Nedomas/el-backo-backend-steps/commit/ac5d762cc51dd3775eb0e29e4f483539b5f655ec#diff-3d29739d18d2b4e16305ff3c49cccece) in this prisma folder

Paste this into that file:

type User {

id: ID! @unique

name: String!

}

In the prisma folder run these two commands:

prisma deploy  
prisma generate