Install React

Create-react-app .

Install dependencies

npm install @material-ui/core@1.4.3 d3-ease@1.0.3 firebase@5.0.4 react-firebase-file-uploader@2.4.1 react-move@2.7.0 react-reveal@1.2.2 react-router-dom@4.3.1 –save

created HOC {props.children} for seeing both HOC/Layout and routes.js data

Way to use HOC

**HOC component (src/HOC/Layout.js)**

import React from "react";

const Layout = props => {

  return (

    <div>

Header

      {props.children}

      Footer

    </div>

  );

};

export default Layout;

src/Routes.js

import React from "react";

import Layout from "./Hoc/Layout";

const Routes = props => {

  return <Layout>hello</Layout>;

};

export default Routes;

* This trick is used to create header and body i.e. having same header and changing body data according to pages

TRICK

You can either use

<Home /> - this will show in all url

Or use router for component – this will show in only / url (i.e. whatever path defined)

<Switch>

        <Route exact component={Home} path="/" />

      </Switch>

    </Layout>

* For the animation we are using react-move

<https://github.com/react-tools/react-move>

* TRICK importing index.js component

import Matches from "./matches/index";

or

import Matches from "./matches";

Both are same and you can Call it like <Matches />

* Add special properties for props – Reusable Component

src\Components\home\matches\index.js

<Tag

          bck="#0e1731"

          size="50 px"

          color="#ffffff"

          add={{ // for overwriting style perv defined eg. material ui

            color: "red"

          }}

        >

          Matches

        </Tag>

src\Components\ui\misc.js

export const Tag = props => {

  const template = (

    <div

      style={{

        background: props.bck,

        fontSize: props.size,

        color: props.color,

        padding: "5px 10px",

        display: "inline-block",

        fontFamily: "Righteous",

        ...props.add

      }}

    >

      {props.children}

    </div>

  );

  if (props.link) {

    return <Link to={props.linkto}>{template}</Link>;

  } else {

    return template;

  }

};

npm install -g firebase-tools

Adding Firebase (Videos)

Initializing Firebase - <https://www.loom.com/share/ee9dfcb00dbc42bd8971bcb916fb8418>

Populating Data in Firebase - <https://www.loom.com/share/8c9de52911f54dbe8c6eb31220f33d1b>

Link

<https://console.firebase.google.com/u/0/project/m-city-4c8ad/settings/general/web:ODAzMjcxOTctMTAxNS00ZmM5LTk5YTAtOTdkZWQ5NmE2NGMx>

<https://firebase.google.com/docs/hosting/reserved-urls?authuser=0>

<https://firebase.google.com/docs/web/setup?authuser=0>

src/firebase.js

import firebase from "firebase/app";

import "firebase/app";

import "firebase/database";

// Your web app's Firebase configuration

var firebaseConfig = {

  apiKey: "AIzaSyCROOna6NC7F5H22k1mnZ7VkarHdRHmOcc",

  authDomain: "m-city-4c8ad.firebaseapp.com",

  databaseURL: "https://m-city-4c8ad.firebaseio.com",

  projectId: "m-city-4c8ad",

  storageBucket: "m-city-4c8ad.appspot.com",

  messagingSenderId: "1017221765198",

  appId: "1:1017221765198:web:6da7e6ce9a4c15ea1b8b1f",

  measurementId: "G-RVPCYPH668"

};

// Initialize Firebase

firebase.initializeApp(firebaseConfig);

// Initialize DB Alias

const firebaseDB = firebase.database();

// fetching data

firebaseDB

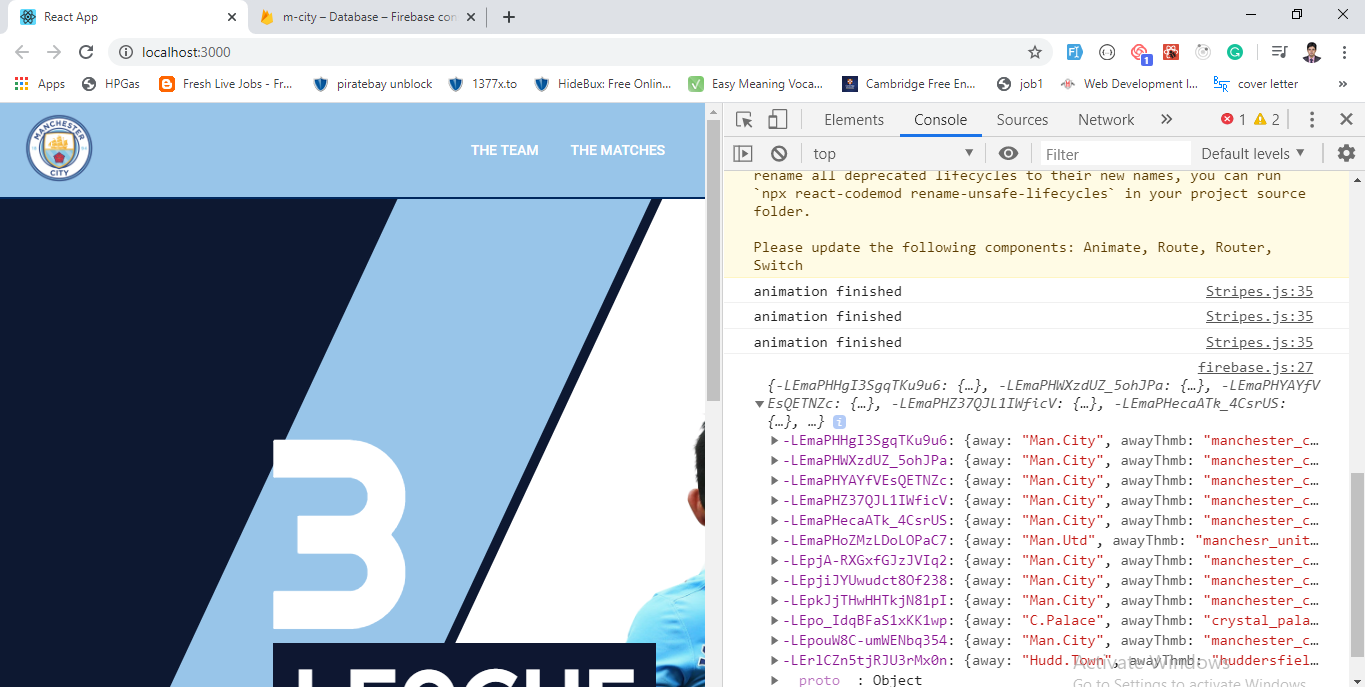
  .ref("matches")

  .once("value")

  .then(snapshot => {

    console.log(snapshot.val());

  });



Last 6 data from firebase

Firebase.js

// Initialize Firebase

firebase.initializeApp(firebaseConfig);

// Initialize DB Alias

const firebaseDB = firebase.database();

const firebaseMatches = firebaseDB.ref("matches");

export { firebase, firebaseMatches };

Blocks.js

import { firebaseMatches } from "../../../firebase";

componentDidMount() {

    firebaseMatches

      .limitToLast(6)

      .once("value")

      .then(snapshot => {

        console.log(snapshot.val());

      });

  }

React Reveal

1. For animation down effect

import Slide from "react-reveal/Slide";

showMatches = matches =>

    matches

      ? matches.map((match, i) => (

          <Slide bottom key={match.id}>

            <div className="item">

              <div className="wrapper">

                <MatchesBlock match={match} />

              </div>

            </div>

          </Slide>

        ))

      : null;

1. Triggering animation when specific div is rendered in browser – console will be logged once triggered

Src/home/meetPlayers/index.js

import Reveal from "react-reveal/Reveal";

<Reveal

        onReveal={() => {

          console.log("reveal");

        }}

      >

anydata

</Reveal>

React Move

Home/meetPlayers/cards.js

state = {

    show: this.props.show,

    cards: [

      {

        bottom: 90,

        left: 300

      },

      {

        bottom: 60,

        left: 200

      },

      {

        bottom: 30,

        left: 100

      },

      {

        bottom: 0,

        left: 0

      }

    ]

  };

STEP1:

showAnimateCards = () => {

      this.state.cards.map((card, i)=>{

          <Animate>

              {({})=>{

                  return (

                      <div>hello</div>;

                  )

              }}

          </Animate>

      });

  }

Finally:

showAnimateCards = () =>

    this.state.cards.map((card, i) => (

      <Animate

        key={i}

        show={this.props.show}

        start={{

          left: 0,

          bottom: 0

        }}

        enter={{

          left: [card.left],

          bottom: [card.bottom],

          timing: { duration: 500, ease: easePolyOut }

        }}

      >

        {({ left, bottom }) => {

          return (

            <div

              style={{

                position: "absolute",

                left,

                bottom

              }}

            >

              hello

            </div>

          );

        }}

      </Animate>

    ));

TRICK TO MAKE REUSABLE COMPONENT

STEP1: In your src make ui/yourReusableComponent.js

import React from "react";

const formFields = () => {

  return <div>FORM FIELDS</div>;

};

export default formFields;

STEP2: Import in the component you want to use

Enroll.js (File using reusable component)

import React, { Component } from "react";

import Fade from "react-reveal/Fade";

import FormFields from "../../ui/formFields";

import { validate } from "../../ui/misc";

class Enroll extends Component {

  // config for email input

  state = {

    formError: false,

    formSuccess: "",

    formdata: {

      email: {

        element: "input",

        value: "",

        config: {

          name: "email\_input",

          type: "email",

          placeholder: "Enter your email"

        },

        validation: {

          // list of validation to check

          required: true,

          email: true

        },

        valid: false,

        validationMessage: ""

      }

    }

  };

  submitForm() {}

  updateForm(element) {

    // We do not directly change formdata state, make a copy of it and update in the end

    const newFormdata = { ...this.state.formdata };

    const newElement = { ...newFormdata[element.id] };

    newElement.value = element.event.target.value;

    // check for validation

    let validData = validate(newElement);

    // updating state for validation message

    newElement.valid = validData[0];

    newElement.validationMessage = validData[1];

    // console.log(newElement);

    newFormdata[element.id] = newElement;

    this.setState({

      formdata: newFormdata

    });

  }

  render() {

    return (

      <Fade>

        <div className="enroll\_wrapper">

          <form onSubmit={event => this.submitForm(event)}>

            <div className="enroll\_title">Enter your email</div>

            <div className="enroll\_input">

              <FormFields

                id={"email"}

                formdata={this.state.formdata.email}

                change={element => this.updateForm(element)}

              />

            </div>

          </form>

        </div>

      </Fade>

    );

  }

}

export default Enroll;

formField.js (Reusable Component)

import React from "react";

const FormFields = ({ formdata, id, change }) => {

  const showError = () => {

    let errorMessage = (

      <div className="error\_label">

        {formdata.validation && !formdata.valid

          ? formdata.validationMessage

          : null}

      </div>

    );

    return errorMessage;

  };

  const renderTemplate = () => {

    let formTemplate = null;

    switch (formdata.element) {

      case "input":

        formTemplate = (

          <div>

            <input

              {...formdata.config}

              value={formdata.value}

              onChange={event => change({ event, id })}

            />

            {showError()}

          </div>

        );

        break;

      default:

        formTemplate = null;

    }

    return formTemplate;

  };

  return <div>{renderTemplate()}</div>;

};

export default FormFields;

Firebase DB

Firebase.js

Check if data is there

import firebase from "firebase/app";

import "firebase/app";

import "firebase/database";

// Your web app's Firebase configuration

var firebaseConfig = {

  apiKey: "AIzaSyCROOna6NC7F5H22k1mnZ7VkarHdRHmOcc",

  authDomain: "m-city-4c8ad.firebaseapp.com",

  databaseURL: "https://m-city-4c8ad.firebaseio.com",

  projectId: "m-city-4c8ad",

  storageBucket: "m-city-4c8ad.appspot.com",

  messagingSenderId: "1017221765198",

  appId: "1:1017221765198:web:6da7e6ce9a4c15ea1b8b1f",

  measurementId: "G-RVPCYPH668"

};

// Initialize Firebase

firebase.initializeApp(firebaseConfig);

// Initialize DB Alias

const firebaseDB = firebase.database();

const firebaseMatches = firebaseDB.ref("matches");

const firebasePromotions = firebaseDB.ref("promotions");

export { firebase, firebaseMatches, firebasePromotions };

// fetching data

// firebaseDB

//   .ref("matches")

//   .once("value")

//   .then(snapshot => {

//     console.log(snapshot.val());

//   });

Enroll.js

import { firebasePromotions } from "../../../firebase";

// ======= Insert into firebase DB: Start ======

      firebasePromotions

        .orderByChild("email")

        .equalTo(dataToSubmit.email)

        .once("value")

        .then(snapshot => {

          console.log(snapshot.val());

        });

      // ======= Insert into firebase DB: End ======

// check already exist and insert

// ======= Insert into firebase DB: Start ======

      firebasePromotions

        .orderByChild("email")

        .equalTo(dataToSubmit.email)

        .once("value")

        .then(snapshot => {

          if (snapshot.val() === null) {

            firebasePromotions.push(dataToSubmit); // insert to firebase db

            this.resetFormSuccess(true);

          } else {

            this.resetFormSuccess(false); // already exist case

          }

        });

      // ======= Insert into firebase DB: End ======

Firebase Auth Setup - <https://www.loom.com/share/85ccc187c800497cba7626889e3a38eb>

Using Firebase Auth in code

Src/firebase.js

import firebase from "firebase/app";

import "firebase/app";

import "firebase/database";

import "firebase/auth";

// Your web app's Firebase configuration

var firebaseConfig = {

  apiKey: "AIzaSyCROOna6NC7F5H22k1mnZ7VkarHdRHmOcc",

  authDomain: "m-city-4c8ad.firebaseapp.com",

  databaseURL: "https://m-city-4c8ad.firebaseio.com",

  projectId: "m-city-4c8ad",

  storageBucket: "m-city-4c8ad.appspot.com",

  messagingSenderId: "1017221765198",

  appId: "1:1017221765198:web:6da7e6ce9a4c15ea1b8b1f",

  measurementId: "G-RVPCYPH668"

};

// Initialize Firebase

firebase.initializeApp(firebaseConfig);

// Initialize DB Alias

const firebaseDB = firebase.database();

const firebaseMatches = firebaseDB.ref("matches");

const firebasePromotions = firebaseDB.ref("promotions");

export { firebase, firebaseMatches, firebasePromotions };

// fetching data

// firebaseDB

//   .ref("matches")

//   .once("value")

//   .then(snapshot => {

//     console.log(snapshot.val());

//   });

Src/signin/index.js

import React, { Component } from "react";

import { firebase } from "../../firebase";

submitForm(event) {

    event.preventDefault();

    let dataToSubmit = {};

    let formIsValid = true;

    // looping through above state to get input values

    for (let key in this.state.formdata) {

      dataToSubmit[key] = this.state.formdata[key].value;

      formIsValid = this.state.formdata[key].value;

    }

    if (formIsValid) {

      // ===== Checking in firebase for valid email and password for Auth: Start       firebase

        .auth()

        .signInWithEmailAndPassword(dataToSubmit.email, dataToSubmit.password)

        .then(() => {

          this.props.history.push("/dashboard");

          // console.log("User is authenticated...");

        })

        .catch(error => {

          this.setState({

            formError: true

          });

        });

      // ====== Checking in firebase for valid email and password for Auth: Start

    } else {

      this.setState({

        formError: true

      });

    }

  }

Firebase signout

import { firebase } from "../../../firebase";

firebase

      .auth()

      .signOut()

      .then(

        () => {

          console.log("Logged out successful.");

        },

        error => {

          console.log("Error logging out");

        }

      );

Redirect using React Router

this.props.history.push("/dashboard"); // Redirects to routes

Private Routes

For using private routes, we can use firebase auth().onAuthStateChanged() method

Index.js (endpoint of React App)

import React from "react";

import ReactDOM from "react-dom";

import "./Resources/css/app.css";

import { BrowserRouter } from "react-router-dom";

import Routes from "./routes";

import { firebase } from "./firebase";

// main

const App = props => {

  return (

    <BrowserRouter>

      <Routes {...props} />

    </BrowserRouter>

  );

};

firebase.auth().onAuthStateChanged(user => {

  // console.log(user);

  ReactDOM.render(<App user={user} />, document.getElementById("root"));

});

//ReactDOM.render(<App />, document.getElementById("root"));

// If you want your app to work offline and load faster, you can change

// unregister() to register() below. Note this comes with some pitfalls.

// Learn more about service workers: https://bit.ly/CRA-PWA

Routes.js

import React from "react";

import Layout from "./Hoc/Layout";

import { Switch, Route } from "react-router-dom";

import Home from "./Components/home/index";

import SignIn from "./Components/signin";

import Dashboard from "./Components/admin/Dashboard";

import PrivateRoutes from "./Components/authRoutes/privateRoutes";

const Routes = props => {

  console.log(props);

  return (

    <Layout>

      <Switch>

        <PrivateRoutes

          {...props} // this props contains user null or data

          exact

          component={Dashboard}

          path="/dashboard"

        />

        {/\* <Route exact component={Dashboard} path="/dashboard" /> \*/}

        <Route exact component={Home} path="/" />

        <Route exact component={SignIn} path="/sign\_in" />

      </Switch>

    </Layout>

  );

};

export default Routes;

PrivateRoutes.js

import React from "react";

import { Route, Redirect } from "react-router-dom";

const PrivateRoutes = ({ user, component: Comp, ...rest }) => {

  return (

    <Route

      {...rest}

      component={props =>

        user ? <Comp {...props} user={user} /> : <Redirect to="/sign\_in" />

      }

    />

  );

};

export default PrivateRoutes;

Public Routes

Routes.js

import React from "react";

import Layout from "./Hoc/Layout";

import { Switch, Route } from "react-router-dom";

import Home from "./Components/home/index";

import SignIn from "./Components/signin";

import Dashboard from "./Components/admin/Dashboard";

import PrivateRoutes from "./Components/authRoutes/privateRoutes";

import PublicRoutes from "./Components/authRoutes/publicRoutes";

const Routes = props => {

  console.log(props);

  return (

    <Layout>

      <Switch>

        <PrivateRoutes

          {...props}

          exact

          component={Dashboard}

          path="/dashboard"

        />

        <PublicRoutes {...props} exact component={Home} path="/" />

        <PublicRoutes {...props} exact component={SignIn} path="/sign\_in" />

        {/\* <Route exact component={Dashboard} path="/dashboard" />

        <Route exact component={Home} path="/" />

        <Route exact component={SignIn} path="/sign\_in" /> \*/}

      </Switch>

    </Layout>

  );

};

export default Routes;

publicRoutes.js

import React from 'react';

import { Route, Redirect } from 'react-router-dom';

const PublicRoutes = ({

    user,

    component: Comp,

    ...rest

}) => {

    return <Route {...rest} component={(props)=>(

        rest.restricted ?

            ( user ?

                <Redirect to="/dashboard"/>

                :

                <Comp {...props} user={user}/>

            )

        :

        <Comp {...props} user={user}/>

    )}/>

};

export default PublicRoutes;

Material UI

Copy source code

<https://material-ui.com/components/tables/>

Firebase get data based on ID

// fetching matches collection data by match id

      firebaseDB

        .ref(`matches/${matchId}`)

        .once("value")

        .then(snapshot => {

          const match = snapshot.val();

          getTeams(match, "Edit Match");

        });

Insert / Update in Firebase

if(this.state.formType === 'Edit Match'){

                firebaseDB.ref(`matches/${this.state.matchId}`)

                .update(dataToSubmit).then(()=>{

                    this.successForm('Updated correctly');

                }).catch((e)=>{

                    this.setState({ formError: true })

                })

            } else {

                firebaseMatches.push(dataToSubmit).then(()=>{

                    this.props.history.push('/admin\_matches');

                }).catch((e)=>{

                    this.setState({ formError: true })

                })

            }

React firebase file uploader

* For this – npm install react-firebase-file-uploader
* Check src/component/ui/fileuploader.js (reusable component)

Will be used in addEditPlayer.js

(src\Components\admin\players\addEditPlayers.js)

// This method is auto called before render - 2

  static getDerivedStateFromProps(props, state) {

    console.log("App - Get Derived state from Props");

    console.log(props, state);

    return null; // you must return state or props or null

  }

Fileuploader documentation (react-firebase-file-uploader = package)

<https://github.com/fris-fruitig/react-firebase-file-uploader>

Firebase Storage

firebasePlayers.storage().ref(this.props.dir)}

For uploading

 <FileUploader

              accept="image/\*"

              name="image"

              randomizeFilename

              storageRef={firebase.storage().ref(this.props.dir)}

              onUploadStart={this.handleUploadStart}

              onUploadError={this.handleUploadError}

              onUploadSuccess={this.handleUploadSuccess}

            />

handleUploadStart = () => {

    this.setState({

      isUploading: true

    });

  };

  handleUploadError = () => {

    this.setState({

      isUploading: false

    });

  };

  handleUploadSuccess = filename => {

    console.log(filename); // image name

    this.setState({

      name: filename,

      isUploading: false

    });

// GETTING image download URL

    // firebase provides only name, we need to get download image URL

    firebase

      .storage()

      .ref(this.props.dir)

      .child(filename)

      .getDownloadURL()

      .then(url => {

        console.log(url);

        this.setState({ fileURL: url });

      });

  };

// updating nested state

  // this method is invoked from fileuploader.js (as props)

  resetImage = () => {

    const newFormdata = { ...this.state.formdata };

    newFormdata["image"].value = "";

    newFormdata["image"].valid = false;

    this.setState({

      defaultImg: "",

      formdata: newFormdata

    });

  };

Player

Getting data in form

componentDidMount() {

    const playerId = this.props.match.params.id;

    if (!playerId) {

      // Add case

      this.setState({

        formType: "Add player"

      });

    } else {

      // Edit case - getting player db + image url from storage from firebase

      firebaseDB

        .ref(`players/${playerId}`)

        .once("value")

        .then(snapshot => {

          const playerData = snapshot.val();

          firebase

            .storage()

            .ref("players")

            .child(playerData.image)

            .getDownloadURL()

            .then(url => {

              this.updateFields(playerData, playerId, "Edit player", url);

            });

        });

    }

  }

On submit

submitForm(event) {

    event.preventDefault();

    let dataToSubmit = {};

    let formIsValid = true;

    for (let key in this.state.formdata) {

      dataToSubmit[key] = this.state.formdata[key].value;

      formIsValid = this.state.formdata[key].valid && formIsValid;

    }

    // on form submit

    if (formIsValid) {

      if (this.state.formType === "Edit player") {

        // Edit player

        firebaseDB

          .ref(`players/${this.state.playerId}`)

          .update(dataToSubmit)

          .then(() => {

            this.successForm("Updated correctly");

          })

          .catch(err => {

            this.setState({ formError: true });

          });

      } else {

        // Add player

        firebasePlayers

          .push(dataToSubmit)

          .then(() => {

            this.props.history.push("/admin\_players"); // redirecting to player dashboard

          })

          .catch(err => {

            this.setState({

              formError: true

            });

          });

      }

    } else {

      this.setState({

        formError: true

      });

    }

  }

Firebase hosting - <https://www.loom.com/share/4273fd4b6fa34f3d9484d3bd8bbb824e>

<https://www.loom.com/share/4273fd4b6fa34f3d9484d3bd8bbb824e>

Project Overview - <https://www.loom.com/share/dc4e7bdafa9d4d28ab9fdefbce28bc03>