

Laporan Praktikum
Mata Kuliah Pemrograman Berbasis Framework
“Week 11”



Nama Penyusun:

Khosy Robbin Hood (1941720067) / TI3D

JURUSAN TEKNOLOGI INFORMASI
PROGRAM STUDI D-IV TEKNIK INFORMATIKA
2022

1. Setting react app dan firebase

1.1. Edit firebase/firebase.js

```
import firebase from 'firebase/compat/app';
import 'firebase/compat/auth';
import 'firebase/compat/firestore';
import { getAuth } from '@firebase/auth';

import { initializeApp } from "firebase/app";

import { getAnalytics } from "firebase/analytics";

const firebaseConfig = {
  apiKey: "AIzaSyDSEE48jrP5O63pOMGeqHH2DMDr1zbuWPo",
  authDomain: "pertemuan-11-16271.firebaseio.com",
  projectId: "pertemuan-11-16271",
  storageBucket: "pertemuan-11-16271.appspot.com",
  messagingSenderId: "995213747816",
  appId: "1:995213747816:web:01dd52a02f51d06756bddd"
};

export const myFirebase =
firebase.initializeApp(firebaseConfig);
const baseDb = myFirebase.firestore();
export const db = baseDb;
```

1.2. Edit actions/auth.js

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

export const LOGIN_REQUEST = "LOGIN_REQUEST";
export const LOGIN_SUCCESS = "LOGIN_SUCCESS";
export const LOGIN_FAILURE = "LOGIN_FAILURE";

export const LOGOUT_REQUEST = "LOGOUT_REQUEST";
export const LOGOUT_SUCCESS = "LOGOUT_SUCCESS";
export const LOGOUT_FAILURE = "LOGOUT_FAILURE";

export const VERIFY_REQUEST = "VERIFY_REQUEST";
export const VERIFY_SUCCESS = "VERIFY_SUCCESS";

const requestLogin = () => {
  return {
    type: LOGIN_REQUEST,
  };
};
```

```
};

const receiveLogin = (user) => {
  return {
    type: LOGIN_SUCCESS,
    user,
  };
};

const loginError = () => {
  return {
    type: LOGIN_FAILURE,
  };
};

const requestLogout = () => {
  return {
    type: LOGOUT_REQUEST,
  };
};

const receiveLogout = () => {
  return {
    type: LOGOUT_SUCCESS,
  };
};

const logoutError = () => {
  return {
    type: LOGOUT_FAILURE,
  };
};

const verifyRequest = () => {
  return {
    type: VERIFY_REQUEST,
  };
};

const verifySuccess = () => {
  return {
    type: VERIFY_SUCCESS,
  };
};
```

```

};

export const loginUser = (email, password) => (dispatch) =>
{
  dispatch(requestLogin());
  myFirebase
    .auth()
    .signInWithEmailAndPassword(email, password)
    .then((user) => {
      dispatch(receiveLogin(user));
    })
    .catch((error) => {
      //Do something with the error if you want!
      dispatch(loginError());
    });
};

export const logoutUser = () => (dispatch) => {
  dispatch(requestLogout());
  myFirebase
    .auth()
    .signOut()
    .then(() => {
      dispatch(receiveLogout());
    })
    .catch((error) => {
      //Do something with the error if you want!
      dispatch(logoutError());
    });
};

export const verifyAuth = () => (dispatch) => {
  dispatch(verifyRequest());
  myFirebase.auth().onAuthStateChanged((user) => {
    if (user !== null) {
      dispatch(receiveLogin(user));
    }
    dispatch(verifySuccess());
  });
};
};

```

1.3. Buka reducers/auth.js

```
import {
  LOGIN_REQUEST,
  LOGIN_SUCCESS,
  LOGIN_FAILURE,
  LOGOUT_REQUEST,
  LOGOUT_SUCCESS,
  LOGOUT_FAILURE,
  VERIFY_REQUEST,
  VERIFY_SUCCESS,
} from "../actions/auth";

export default (
  state = {
    isLoggingIn: false,
    isLoggingOut: false,
    isVerifying: false,
    loginError: false,
    logoutError: false,
    isAuthenticated: false,
    user: {},
  },
  action
) => {
  switch (action.type) {
    case LOGIN_REQUEST:
      return {
        ...state,
        isLoggingIn: true,
        loginError: false,
      };
    case LOGIN_SUCCESS:
      return {
        ...state,
        isLoggingIn: false,
        isAuthenticated: true,
        user: action.user,
      };
    case LOGIN_FAILURE:
      return {
        ...state,
        isLoggingIn: false,
        isAuthenticated: false,
      };
  }
}
```

```

        loginError: true,
      };
    case LOGOUT_REQUEST:
      return {
        ...state,
        isLoggingOut: true,
        logoutError: false,
      };
    case LOGOUT_SUCCESS:
      return {
        ...state,
        isLoggingOut: false,
        isAuthenticated: false,
        user: {},
      };
    case LOGOUT_FAILURE:
      return {
        ...state,
        isLoggingOut: false,
        logoutError: true,
      };
    case VERIFY_REQUEST:
      return {
        ...state,
        isVerifying: true,
        verifyingError: false,
      };
    case VERIFY_SUCCESS:
      return {
        ...state,
        isVerifying: false,
      };
    default:
      return state;
  }
};

```

1.4. Edit reducers/index.js

```

import { combineReducers } from "redux";
import auth from "../auth";
export default combineReducers({ auth });

```

1.5. Pilih src folder called configureStore.js dan edit

```
import { applyMiddleware, createStore } from "redux";
import thunkMiddleware from "redux-thunk";
import { verifyAuth } from "../actions/auth";
import rootReducer from "../reducers";

export default function configureStore(persistedState) {
  const store = createStore(
    rootReducer,
    persistedState,
    applyMiddleware(thunkMiddleware)
  );
  store.dispatch(verifyAuth());
  return store;
}
```

1.6. Pilih dan edit folder src dan root.js

```
import React from "react";
import { Provider } from "react-redux";
import { BrowserRouter as Router } from "react-router-dom";
import App from "../App";
import configureStore from "../configureStore";

const store = configureStore();

function Root() {
  return (
    <Provider store={store}>
      <Router>
        <App />
      </Router>
    </Provider>
  );
}

export default Root;
```

1.7. Pilih dan edit folder src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import Root from './root';
import reportWebVitals from './reportWebVitals';
```

```

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

```

1.8. Buat lah coding App.js

```

import React from "react";
import { Route, Switch } from "react-router-dom";
import { connect } from "react-redux";
import ProtectedRoute from "../components/ProtectedRoute";
import Home from "../components/Home";
import Login from "../components/Login";

function App(props) {
  const { isAuthenticated, isVerifying } = props;
  return (
    <Switch>
      <ProtectedRoute
        exact
        path="/"
        component={Home}
        isAuthenticated={isAuthenticated}
        isVerifying={isVerifying}
      />
      <Route path="/login" component={Login} />
    </Switch>
  );
}

function mapStateToProps(state) {

```



```

    return {
      isAuthenticated: state.auth.isAuthenticated,
      isVerifying: state.auth.isVerifying
    };
  }
}

export default connect(mapStateToProps) (App);

```

1.9. Buatlah file dan simpan components/Login.js

```

import React, { Component } from "react";
import { connect } from "react-redux";
import { Redirect } from "react-router-dom";
import { loginUser } from "../actions/auth";
import { withStyles } from "@material-ui/styles";
import Avatar from "@material-ui/core/Avatar";
import Button from "@material-ui/core/Button";
import TextField from "@material-ui/core/TextField";
import LockOutlinedIcon from
"@material-ui/icons/LockOutlined";
import Typography from "@material-ui/core/Typography";
import Paper from "@material-ui/core/Paper";
import Container from "@material-ui/core/Container";

const styles = () => ({
  "@global": {
    body: {
      backgroundColor: "#fff"
    }
  },
  paper: {
    marginTop: 100,
    display: "flex",
    padding: 20,
    flexDirection: "column",
    alignItems: "center"
  },
  avatar: {
    marginLeft: "auto",
    marginRight: "auto",
    backgroundColor: "#f50057"
  },
  form: {
    marginTop: 1

```

```

    },
    errorText: {
      color: "#f50057",
      marginBottom: 5,
      textAlign: "center"
    }
  });

class Login extends Component {
  state = { email: "", password: "" };
  handleEmailChange = ({ target }) => {
    this.setState({ email: target.value });
  };
  handlePasswordChange = ({ target }) => {
    this.setState({ password: target.value });
  };
  handleSubmit = () => {
    const { dispatch } = this.props;
    const { email, password } = this.state;
    dispatch(loginUser(email, password));
  };
  render() {
    const { classes, loginError, isAuthenticated } =
this.props;
    if (isAuthenticated) {
      return <Redirect to="/" />;
    } else {
      return (
        <Container component="main" maxWidth="xs">
          <Paper className={classes.paper}>
            <Avatar className={classes.avatar}>
              <LockOutlinedIcon />
            </Avatar>
            <Typography component="h1"
variant="h5">
              Sign in
            </Typography>
            <TextField
              variant="outlined"
              margin="normal"
              fullWidth
              id="email"
              label="Email Address"

```

```

        name="email"

onChange={this.handleEmailChange}
      />
      <TextField
        variant="outlined"
        margin="normal"
        fullWidth
        name="password"
        label="Password"
        type="password"
        id="password"

onChange={this.handlePasswordChange}
      />
      {loginError && (
        <Typography component="p"
className={classes.errorText}>
        Incorrect email or password.
      </Typography>
    )}
      <Button
        type="button"
        fullWidth
        variant="contained"
        color="primary"
        className={classes.submit}
        onClick={this.handleSubmit}
      >
        Sign In
      </Button>
    </Paper>
  </Container>
);
}
}
}

function mapStateToProps(state) {
  return {
    isLoggingIn: state.auth.isLoggingIn,
    loginError: state.auth.loginError,
    isAuthenticated: state.auth.isAuthenticated
  }
}

```

```

    };
}

export default
withStyles(styles)(connect(mapStateToProps)(Login));

```

1.10. Buka component/Home.js

```

import React, { Component } from "react";
import { connect } from "react-redux";
import { logoutUser } from "../actions/auth";

class Home extends Component {
  handleLogout = () => {
    const { dispatch } = this.props;
    dispatch(logoutUser());
  };

  render() {
    const { isLoggingOut, logoutError } = this.props;
    return (
      <div>
        <h1>This is your app's protected area.</h1>
        <p>Any routes here will also be
protected</p>
        <button
onClick={this.handleLogout}>Logout</button>
        {isLoggingOut && <p>Logging Out....</p>}
        {logoutError && <p>Error logging out</p>}
      </div>
    );
  }
}

function mapStateToProps(state) {
  return {
    isLoggingOut: state.auth.isLoggingOut,
    logoutError: state.auth.logoutError
  };
}

export default connect(mapStateToProps)(Home);

```

1.11. Buat File dalam folder /src/component/ dengan nama protectedRoute.js

```

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

```

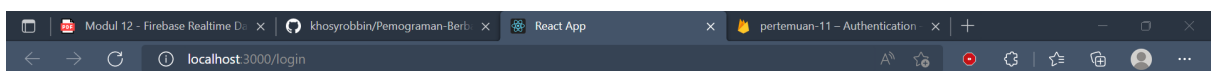
```


const ProtectedRoute = ({
  component: Component,
  isAuthenticated,
  isVerifying,
  ...rest
}) => (
  <Route
    {...rest}
    render={props =>
      isVerifying ? (
        <div />
      ) : isAuthenticated ? (
        <Component {...props} />
      ) : (
        <Redirect
          to={{
            pathname: "/login",
            state: { from: props.location }
          }}
        />
      )
    }
  />
);

export default ProtectedRoute;

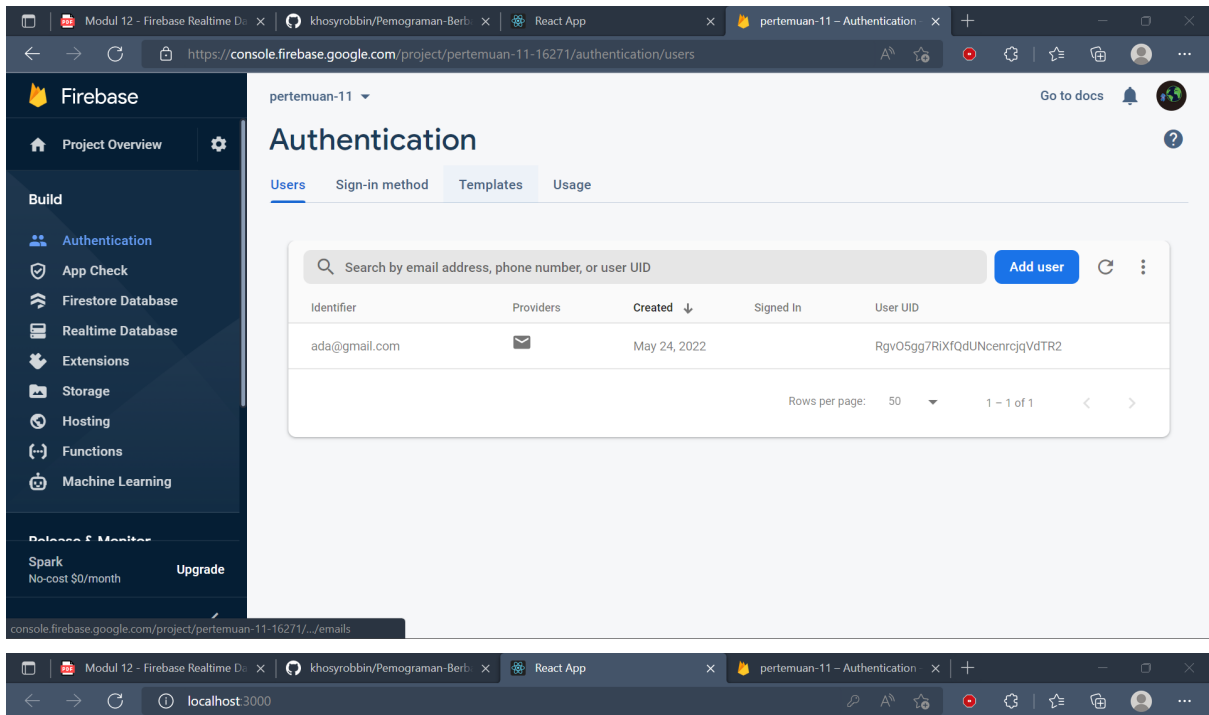
```

Hasil





Sign in



This is your app's protected area.

Any routes here will also be protected

[Logout](#)

Link Github: [khosyrobbin/Pemograman-Berbasis-Framework \(github.com\)](https://github.com/khosyrobbin/Pemograman-Berbasis-Framework)