

# Web application development

(Introduction to Basic React)

Instructor: Tran Vinh Khiem

September 1st, 2022



*Smart Software System Team*

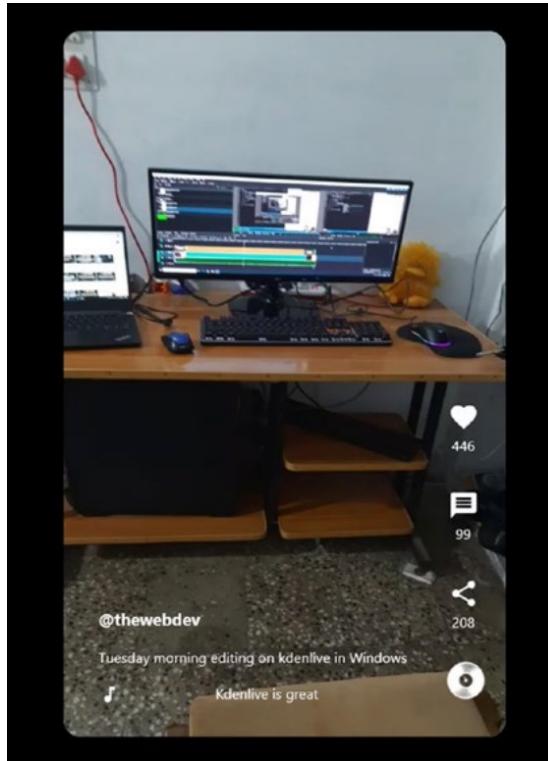


*“We love what we do and we do what our clients  
love & work with great clients all over the world  
to create thoughtful and purposeful websites.”*

— ProWeb365



# Basic React – Exercise 1 – Create reels app



# Basic React – Exercise 1 – Create reels app - FE

```
<import React, { useState, useEffect } from 'react'
import './App.css';
import Video from './components/Video';
import axios from './components/axios';

function App() {
  const [videos, setVideos] = useState([])

  useEffect(() => {
    async function fetchData() {
      const res = await axios.get("/v2/posts")
      setVideos(res.data)
      return res
    }
    fetchData()
  }, [])

  return (
    <div className="app">
      <div className="app__videos">
        {videos.map(({ url, channel, description, song, likes, shares, messages }) => (
          <Video
            key={url}
            url={url}
            channel={channel}
            description={description}
            song={song}
            likes={likes}
            shares={shares}
            messages={messages}
          />
        )));
      </div>
    </div>
  );
}

export default App;
```

```
html{
  scroll-snap-type: y mandatory;
}

.app{
  height: 100vh;
  background-color: black;
  display: grid;
  place-items: center;
}

.app__videos{
  position: relative;
  height: 800px;
  border-radius: 20px;
  overflow: scroll;
  width: 80%;
  max-width: 500px;
  scroll-snap-type: y mandatory;
}

.app__videos::-webkit-scrollbar{
  display: none;
}

.app__videos{
  -ms-overflow-style: none;
  scrollbar-width: none;
}
```

# Basic React – Exercise 1 – Create reels app - FE

```
import React, { useRef, useState } from 'react'
import './Video.css'
import VideoFooter from './VideoFooter'
import VideoSidebar from './VideoSidebar'

const Video = ({ url, channel, description, song, likes, shares, messages }) => {
  const [playing, setPlaying] = useState(false)
  const videoRef = useRef(null)
  const handleVideoPress = () => {
    if(playing){
      videoRef.current.pause()
      setPlaying(false)
    } else {
      videoRef.current.play()
      setPlaying(true)
    }
  }

  return (
    <div className="video">
      <video
        src={url}
        className="video__player"
        loop
        ref={videoRef}
        onClick={handleVideoPress}
      >
      </video>
      <VideoFooter channel={channel} description={description} song={song} />
      <VideoSidebar likes={likes} shares={shares} messages={messages} />
    </div>
  )
}

export default Video
```

```
.video{
  position: relative;
  background-color: white;
  width: 100%;
  height:100%;
  scroll-snap-align: start;
}

.video__player{
  object-fit: fill;
  width: 100%;
  height: 100%;
}
```

# Basic React – Exercise 1 – Create reels app - FE



```
import React from 'react'
import './VideoFooter.css'
import MusicNoteIcon from '@material-ui/icons/MusicNote'
import Ticker from 'react-ticker'

const VideoFooter = ({ channel, description, song }) => {
  return [
    <div className="videoFooter">
      <div className="videoFooter__text">
        <h3>@{channel}</h3>
        <p>{description}</p>
        <div className="videoFooter__ticker">
          <MusicNoteIcon className="videoFooter__icon" />
          <Ticker mode="smooth">
            {({ index }) => (
              <>
                <p>{song}</p>
              </>
            )}
          </Ticker>
        </div>
      </div>
      
    </div>
  ]
}

export default VideoFooter
```

```
.videoFooter{
  position: relative;
  color: white;
  bottom: 150px;
  margin-left: 40px;
  display: flex;
}

.videoFooter__text{
  flex: 1;
}

.videoFooter__text > h3{
  padding-bottom: 20px;
}

.videoFooter__text > p{
  padding-bottom: 20px;
}

.videoFooter__icon{
  position: absolute;
}

.videoFooter__ticker > .ticker{
  height: fit-content;
  margin-left: 30px;
  width: 60%;
}

.videoFooter__record{
  animation: spinTheRecord infinite 5s linear;
  height: 50px;
  filter: invert(1);
  position: absolute;
  bottom: 0;
  right: 20px;
}
```

# Basic React – Exercise 1 – Create reels app - FE

```
Import React, { useState } from 'react'
import './VideoSidebar.css'
import FavoriteIcon from '@material-ui/icons/Favorite'
import FavoriteBorderIcon from '@material-ui/icons/FavoriteBorder'
import MessageIcon from '@material-ui/icons/Message'
import ShareIcon from '@material-ui/icons/Share'

const VideoSidebar = ({ likes, shares, messages }) => {
  const [liked, setLiked] = useState(false)

  return (
    <div className="videoSidebar">
      <div className="videoSidebar__button">
        { liked ? <FavoriteIcon fontSize="large" onClick={e => setLiked(false)} /> : <FavoriteBorderIcon fontSize="large" onClick={e => setLiked(true)} /> }
        <p>{liked ? +likes + 1 : likes}</p>
      </div>
      <div className="videoSidebar__button">
        <MessageIcon fontSize="large" />
        <p>{messages}</p>
      </div>
      <div className="videoSidebar__button">
        <ShareIcon fontSize="large" />
        <p>{shares}</p>
      </div>
    </div>
  )
}

export default VideoSidebar
```

```
.videoSidebar{
  position: absolute;
  top: 50%;
  right: 10px;
  color: white;
}

.videoSidebar__button{
  padding: 20px;
  text-align: center;
}
```

# Basic React – Exercise 1 – Create reels app - FE

```
import axios from 'axios'

const instance = axios.create({
  baseURL: "your-backend-heroku-url"
})

export default instance
```

# Basic React – Exercise 1 – Create reels app - BE

```
import mongoose from 'mongoose'

const shortVideoSchema = mongoose.Schema({
  url: String,
  channel: String,
  description: String,
  song: String,
  likes: String,
  shares: String,
  messages: String
})

export default mongoose.model('shortVideos', shortVideoSchema)
```

# Basic React – Exercise 1 – Create reels app - BE

```
import express from 'express'
import mongoose from 'mongoose'
import Cors from 'cors'
import Videos from './dbModel.js'
//App Config
const app = express()
const port = process.env.PORT || 9000
const connection_url = 'your-mongodb-api-url'

//Middleware
app.use(express.json())
app.use(Cors())

//DB Config
mongoose.connect(connection_url, {
  useNewUrlParser: true,
  useCreateIndex: true,
  useUnifiedTopology: true
})

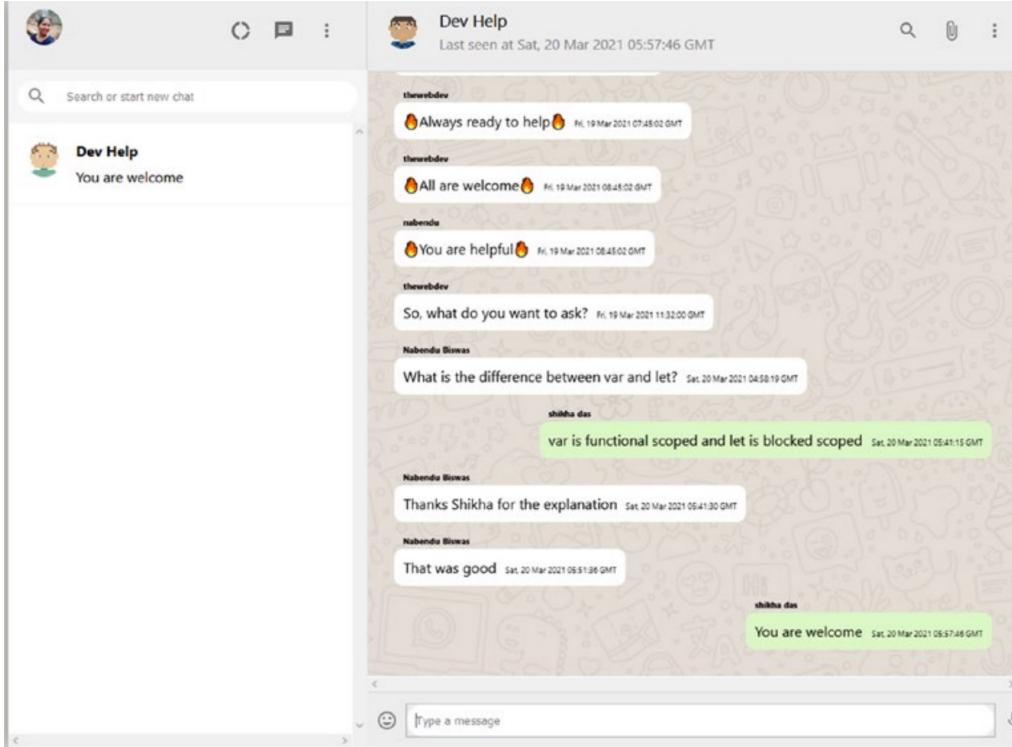
//API Endpoints
app.get("/", (req, res) => res.status(200).send("He so lo , ho so ly ly"))

app.post('/v2/posts', (req, res) => {
  const dbVideos = req.body
  Videos.create(dbVideos, (err, data) => [
    if(err)
      res.status(500).send(err)
    else
      res.status(201).send(data)
  ])
})
```

```
app.get('/v2/posts', (req, res) => {
  Videos.find((err, data) => {
    if(err) {
      res.status(500).send(err)
    } else {
      res.status(200).send(data)
    }
  })
})
//Listener
app.listen(port, () => console.log(`Listening on localhost: ${port}`))
```



# Basic React – Exercise 2 – Create messaging app





# Basic React – Exercise 2 – Create messaging app - Structure

```
└── messaging-app-frontend
    ├── .firebase
    ├── public
    └── src
        └── components
            ├── axios.js
            ├── Chat.css
            ├── Chat.js
            ├── Login.css
            ├── Login.js
            ├── reducer.js
            ├── Sidebar.css
            ├── Sidebar.js
            ├── SidebarChat.css
            ├── SidebarChat.js
            ├── StateProvider.js
            ├── App.css
            ├── App.js
            ├── firebase.js
            ├── index.css
            ├── index.js
            └── .firebaserc
                └── .gitignore
```

```
└── messaging-app-backend
    ├── .gitignore
    ├── dbMessages.js
    ├── package-lock.json
    ├── package.json
    └── server.js
```



# Basic React – Exercise 2 – Create messaging app - FE

```
import React, { useEffect, useState } from 'react'
import './App.css';
import Chat from './components/Chat';
import Sidebar from './components/Sidebar';
import Pusher from 'pusher-js';
import axios from './components/axios';
import Login from './components/Login';
import { useStateValue } from './components/StateProvider';

function App() {
  const [messages, setMessages] = useState([])
  const [user], dispatch = useStateValue()

  useEffect(() => {
    axios.get("/messages-sync").then(res => {
      setMessages(res.data)
    })
  }, [])

  useEffect(() => {
    const pusher = new Pusher('9e297c1b3f7413a26cce', {
      cluster: 'ap2'
    });

    const channel = pusher.subscribe('messages');
    channel.bind('inserted', (data) => {
      setMessages([...messages, data])
    });

    return () => {
      channel.unbind_all()
      channel.unsubscribe()
    }
  }, [messages])

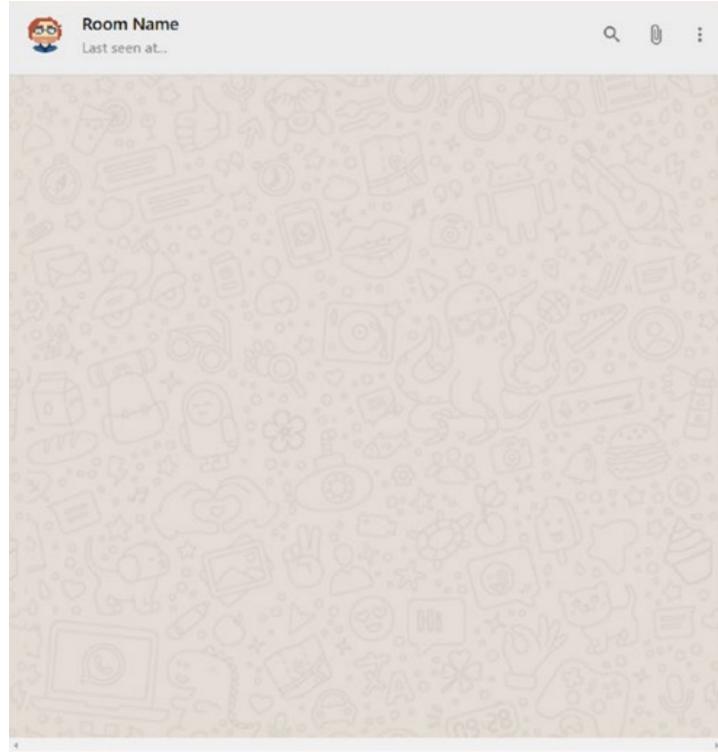
  console.log(messages)
}
```

```
return (
  <div className="app">
    { !user ? <Login /> : (
      <div className="app__body">
        <Sidebar messages={messages} />
        <Chat messages={messages} />
      </div>
    )}
  </div>
);

export default App;
```



# Basic React – Exercise 2 – Create messaging app - FE - Chat





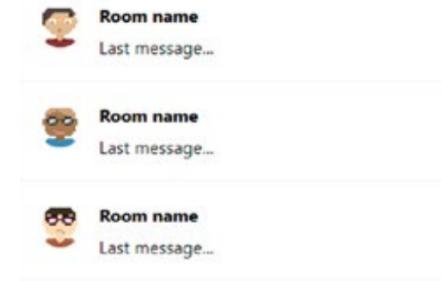
# Basic React – Exercise 2 – Create messaging app – FE - Chat

```
1 import React, { useEffect, useState } from 'react'
2 import { Avatar, IconButton } from '@material-ui/core'
3 import { AttachFile, MoreVert, SearchOutlined, InsertEmoticon } from '@material-ui/icons'
4 import MicIcon from '@material-ui/icons/Mic'
5 import './Chat.css'
6 import axios from './axios'
7 import { useStateValue } from './StateProvider'
8
9 const Chat = ({ messages }) => {
0  const [seed, setSeed] = useState("")
1  const [input, setInput] = useState("")
2  const [{ user }, dispatch] = useStateValue()
3
4  const sendMessage = async (e) => {
5    e.preventDefault()
6    await axios.post('/messages/new', {
7      message: input,
8      name: user.displayName,
9      timestamp: new Date().toUTCString(),
0      received: true
1    })
2    setInput("")
3  }
4
5  useEffect(() => {
6    |  setSeed(Math.floor(Math.random() * 5000))
7  }, [])
8
9
return (
0  <div className="chat">
1    <div className="chat_header">
2      <Avatar src={(`https://avatars.dicebear.com/api/human/b${seed}.svg`)} />
3      <div className="chat_headerInfo">
4        <h3>Dev Help</h3>
5        <p>Last seen at (" ")<br/>| ({messages[messages.length -1].timestamp})</p>
6      </div>
7    </div>
8
9  </div>
```

```
0  <div className="chat_headerRight">
1    <IconButton>
2      <SearchOutlined />
3    </IconButton>
4    <IconButton>
5      <AttachFile />
6    </IconButton>
7    <IconButton>
8      <MoreVert />
9    </IconButton>
0  </div>
1
</div>
<div className="chat_body">
2   {messages.map(message => (
3     <p className={`chat_message ${message.name === user.displayName ? 'chat_reversed' : ''}`}>
4       <span className="chat_name">{message.name}</span>
5       <span className="chat_timestamp">
6         {message.timestamp}
7       </span>
8     </p>
9   )));
0
1  </div>
2  <div className="chat_footer">
3    <InsertEmoticon />
4    <form>
5      <input
6        value={input}
7        onChange={e => setInput(e.target.value)}
8        placeholder="Type a message"
9        type="text"
0      />
1      <button onClick={sendMessage} type="submit">Send a message</button>
2    </form>
3    <MicIcon />
4  </div>
5
6  </div>
7
8  export default Chat
9
```



# Basic React – Exercise 2 – Create messaging app – FE – Side bar





# Basic React – Exercise 2 – Create messaging app – FE – Side bar

```
import React from 'react'
import './Sidebar.css'
import DonutLargeIcon from '@material-ui/icons/DonutLarge'
import ChatIcon from '@material-ui/icons/Chat'
import MoreVertIcon from '@material-ui/icons/MoreVert'
import { Avatar, IconButton } from '@material-ui/core'
import { SearchOutlined } from '@material-ui/icons'
import SidebarChat from './SidebarChat'
import { useStateValue } from './StateProvider';

const Sidebar = ({ messages }) => {
    const [ user ], dispatch = useStateValue()

    return (
        <div className="sidebar">
            <div className="sidebar__header">
                <Avatar src={user?.photoURL} />
                <div className="sidebar__headerRight">
                    <IconButton>
                        <DonutLargeIcon />
                    </IconButton>
                    <IconButton>
                        <ChatIcon />
                    </IconButton>
                    <IconButton>
                        <MoreVertIcon />
                    </IconButton>
                </div>
            </div>
            <div className="sidebar__search">
                <div className="sidebar__searchContainer">
                    <SearchOutlined />
                    <input placeholder="Search or start new chat" type="text" />
                </div>
            </div>
            <div className="sidebar__chats">
                <SidebarChat messages={messages} />
            </div>
        </div>
    )
}
```



# Basic React – Exercise 2 – Create messaging app – FE – Side bar chat

```
import React, { useEffect, useState } from 'react'
import { Avatar } from '@material-ui/core'
import './SidebarChat.css'

const SidebarChat = ({ messages }) => {
  const [seed, setSeed] = useState("")

  useEffect(() => {
    setSeed(Math.floor(Math.random() * 5000))
  }, [])

  return (
    <div className="sidebarChat">
      <Avatar src={`your-avatar-icon-api`} />
      <div className="sidebarChat__info">
        <h2>Dev Help</h2>
        <p>{messages[messages.length -1]?.message}</p>
      </div>
    </div>
  )
}

export default SidebarChat
```



# Basic React – Exercise 2 – Create messaging app – FE – State context

```
import React, { createContext, useContext, useReducer } from "react"

export const StateContext = createContext()

export const StateProvider = ({ reducer, initialState, children }) => (
  <StateContext.Provider value={useReducer(reducer, initialState)}>
    {children}
  </StateContext.Provider>
)

export const useStateValue = () => useContext(StateContext)
```



# Basic React – Exercise 2 – Create messaging app – FE – Reducer

```
export const initialState = { user: null }

export const actionTypes = {
  SET_USER: "SET_USER"
}

const reducer = (state, action) => {
  console.log(action)
  switch(action.type) {
    case actionTypes.SET_USER:
      return {
        ...state,
        user: action.user
      }
    default:
      return state
  }
}

export default reducer
```



# Basic React – Exercise 2 – Create messaging app – FE – Login

```
import React from 'react'
import { Button } from '@material-ui/core'
import './Login.css'
import { auth, provider } from '../firebase'
import { actionTypes } from './reducer'
import { useStateValue } from './StateProvider'

const Login = () => {
  const [{}, dispatch] = useStateValue()

  const signIn = () => {
    auth.signInWithPopup(provider)
      .then(result => {
        dispatch({
          type: actionTypes.SET_USER,
          user: result.user
        })
      })
      .catch(error => alert(error.message))
  }

  return (
    <div className="login">
      <div className="login__container">
        
        <div className="login__text">
          <h1>Sign in to Messaging App</h1>
        </div>
        <Button onClick={signIn}>Sign In with Google</Button>
      </div>
    </div>
  )
}

export default Login
```

```
.login{
  background-color: #f8f8f8;
  height: 100vh;
  width: 100vw;
  display: grid;
  place-items: center;
}

.login__container{
  padding: 100px;
  text-align: center;
  background-color: white;
  border-radius: 10px;
  box-shadow: -1px 4px 20px -6px rgba(0, 0, 0, 0.75);
}

.login__container > img {
  object-fit: contain;
  height: 100px;
  margin-bottom: 40px;
}

.login__container > button {
  margin-top: 50px;
  text-transform: inherit !important;
  background-color: #0a8d48 !important;
  color: white;
}
```



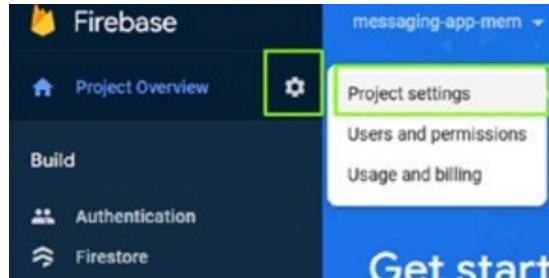
# Basic React – Exercise 2 – Create messaging app – FE – Login– Add google authentication

The screenshot shows the Firebase console's Authentication section. The left sidebar has 'Authentication' selected. The main area is titled 'Authentication' with the sub-instruction: 'Authenticate and manage users from a variety of providers without server-side code'. A 'Get started' button is highlighted with a red box. Below it, there's a note about Google Sign-In being automatically configured for iOS and web apps, with a 'Enable' toggle switch. A 'Project public-facing name' field contains 'project' and a 'Project support email' field contains '3@gmail.com'. At the bottom, there are sections for 'Whitelist client IDs from external projects (optional)' and 'Web SDK configuration', both with dropdown menus. A 'Save' button at the bottom right is also highlighted with a red box.

The screenshot shows the 'Sign-in method' tab of the Firebase Authentication settings. It lists three providers: 'Email/Password' (Disabled), 'Phone' (Disabled), and 'Google' (Disabled). The 'Google' row is highlighted with a blue border. A blue pen icon is located to the right of the 'Google' row.



# Basic React – Exercise 2 – Create messaging app – FE – Login– Add firebase to your project



The screenshot shows the Firebase Project Overview interface. On the left, there's a sidebar with icons for Project Overview, Build, Authentication, and Firestore. The 'Project Overview' icon is selected. A dropdown menu is open over the 'Build' icon, showing options: 'Project settings' (which is highlighted with a green border), 'Users and permissions', and 'Usage and billing'. At the bottom of this sidebar is a large blue 'Get started' button. To the right, the main content area has a header 'messaging-app-mem' with a dropdown arrow. Below it, the message 'There are no apps in your project' is displayed, followed by 'Select a platform to get started'. There are four circular icons for 'iOS', 'Android', 'Web', and 'Cloud Functions'. The 'Web' icon is highlighted with a blue border. At the bottom left, there's a modal window titled 'Add Firebase to your web app' with a 'Register app' button. Below this, there's a note about setting up Firebase Hosting with a checked checkbox and a link to 'Learn more'.

x Add Firebase to your web app

1 Register app

Also set up **Firebase Hosting** for this app. [Learn more](#)

Hosting can also be set up later. It's free to get started at any time.



# Basic React – Exercise 2 – Create messaging app – FE – Login– Add firebase to your project

```
import firebase from 'firebase/app';
import 'firebase/auth';           // for authentication
import 'firebase/storage';        // for storage
import 'firebase/database';       // for realtime database
import 'firebase/firestore';      // for cloud firestore

const firebaseConfig = {
  apiKey: "",
  authDomain: "",
  projectId: "",
  storageBucket: "",
  messagingSenderId: "",
  appId: ""
};

const firebaseApp = firebase.initializeApp(firebaseConfig)
const db = firebaseApp.firebaseio()
const auth = firebase.auth()
const provider = new firebase.auth.GoogleAuthProvider()

export { auth, provider }
export default db
```



# Basic React – Exercise 2 – Create messaging app – FE – axios

```
import axios from 'axios'

const instance = axios.create({
  baseURL: "your-api-backend"
})

export default instance
```



# Basic React – Exercise 2 – Create messaging app – BE – DB

```
import mongoose from 'mongoose'

const messagingSchema = mongoose.Schema({
  message: String,
  name: String,
  timestamp: String,
  received: Boolean
})

export default mongoose.model('messagingmessages', messagingSchema)
```



# Basic React – Exercise 2 – Create messaging app – BE – Create pusher – Real time DB

PUSHER  
A MessageBird company

Products ▾ Developers ▾ User stories Blog Pricing ▾

Sign in Sign up

Powering realtime experiences for mobile and web

Bi-directional hosted APIs that are flexible, scalable and easy to use. We create and maintain complex messaging infrastructure so you can build the realtime features your

Pubsub Notifications

PHP Node Ruby ASP Java Python Go

```
1 $pusher->trigger('my-channel', 'my-event', [
2   'message' => 'Hello world'
3 ]);
```

Create your Channels app

Name your app ⓘ   

Select a cluster ⓘ   

Create apps for multiple environments? ⓘ

Choose your tech stack (optional)

Front end   

Back end   

Create app Cancel



# Basic React – Exercise 2 – Create messaging app – BE – Server

```
import express from 'express'
import mongoose from 'mongoose'
import Cors from 'cors'
import Messages from './dbMessages.js'
import Pusher from 'pusher'

//App Config
const app = express()
const port = process.env.PORT || 9000
const connection_url = 'your-mongodb'

const pusher = new Pusher({
  appId: "your-app-id",
  key: "your-key",
  secret: "your",
  cluster: "your",
  useTLS: true
});

//Middleware
app.use(express.json())
app.use(Cors())

//DB Config
mongoose.connect(connection_url, {
  useNewUrlParser: true,
  useCreateIndex: true,
  useUnifiedTopology: true
})

//API Endpoints
const db = mongoose.connection
db.once("open", () => {
  console.log("DB Connected")
  const msgCollection = db.collection("messagingmessages")
  const changeStream = msgCollection.watch()
```

```
changeStream.on('change', change => {
  console.log(change)

  if(change.operationType === "insert") {
    const messageDetails = change.fullDocument
    pusher.trigger("messages", "inserted", {
      name: messageDetails.name,
      message: messageDetails.message,
      timestamp: messageDetails.timestamp,
      received: messageDetails.received
    })
  } else {
    console.log('Error trigerring Pusher')
  }
})

app.get("/", (req, res) => res.status(200).send("Hello UIT guys"))

app.post('/messages/new', (req, res) => {
  const dbMessage = req.body
  Messages.create(dbMessage, (err, data) => {
    if(err)
      res.status(500).send(err)
    else
      res.status(201).send(data)
  })
})

app.get('/messages/sync', (req, res) => {
  Messages.find((err, data) => {
    if(err)
      res.status(500).send(err)
    else
      res.status(200).send(data)
  })
})

//Listener
app.listen(port, () => console.log(`Listening on localhost: ${port}`))
```



# Q & A



**Thank you for cooperating  
Gét gó**

*“Coming together is a beginning;  
Keeping together is progress;  
Working together is success.”*

- HENRY FORD