

## FULLSTACK REALTIME CHAT APP

1. Create a folder “server”. Then run this command to create package.json file “npm init”. Then Install all Necessary Dependencies:

Dependency	Version
bcryptjs	^3.0.2
cloudinary	^2.6.1
cookie-parser	^1.4.7
dotenv	^16.5.0
express	^5.1.0
express-fileupload	^1.5.1
jsonwebtoken	^9.0.2
mongoose	^8.16.0
socket.io	^4.8.1

2. Modify scripts in package.json file.

3. Create a config/config.env file and add the following environmental variables to this file.

- MONGO\_URI
- PORT
- CLOUDINARY\_CLOUD\_NAME
- CLOUDINARY\_API\_KEY
- CLOUDINARY\_API\_SECRET
- JWT\_SECRET\_KEY
- JWT\_EXPIRE
- COOKIE\_EXPIRE
- NODE\_ENV
- FRONTEND\_URL

4. Now create a new file app.js and setup express, dotenv, cookie-parser, cors and fileUpload.

5. Now create server.js file and listen to your app.

6. Create a “temp” folder and leave it.

7. Now create a database/db.js file and connect to your database and call this function in your app.js file.

8. Create a middleware/catchAsyncError.middleware.js file to handle all async Errors and its benefit is that you don't have to use try-catch block everywhere in your code.

9. Now create a folder “models”.

10. Now create two files for your database models in this folder:

- user.model.js
- message.model.js

## ← FULLSTACK REALTIME CHAT APP.docx

12. Now add empty 3 functions to your controller.

- signup
- signin
- signout
- getUser
- updateProfile

13. Now create a folder routes. Create a new file user.routes.js. And setup all your user routes. Then import your routes in you app.js file.

14. Now create your signup function and then create a new file utils/jwtToken.js and write code for your jsonwebtoken and test it on postman.

15. Now create your signin function and test it on postman.

16. Now create your signout function and test it on postman.

17. Now create a middleware/auth.middleware.js.

18. Now create your getUser function and test it on postman.

19. Now create your updateProfile function and test it on postman.

20. Now create a new file controllers/message.controller.js.

21. Now add 3 empty function to this file.

- getAllUsers
- getMessages
- sendMessage

22. Now create a new file message.routes.js and setup all your routes. Then import your routes in your app.js file.

23. Now create your getAllUsers function and test it on postman.

24. Now create your getMessages function and test it on postman.

25. Now create your sendMessage function and while implementing send message functionality you will need to setup socket.io so create a new file utils/socket.js and setup your sockets and then initialize your socket server in your server.js file and then finally you can test it on postman.

## BACKEND COMPLETED