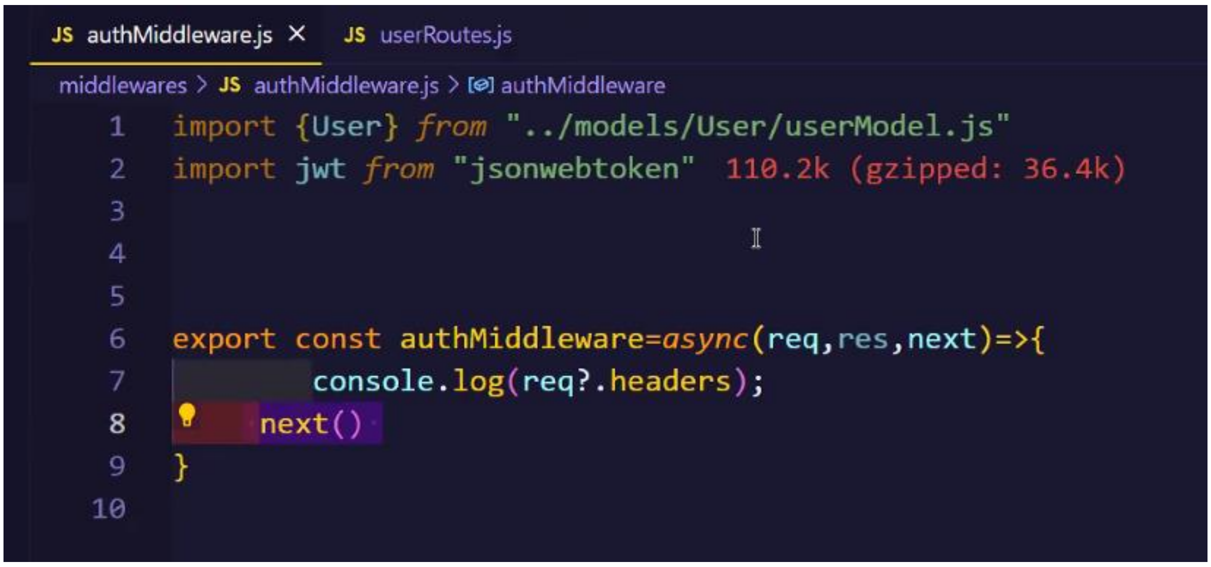
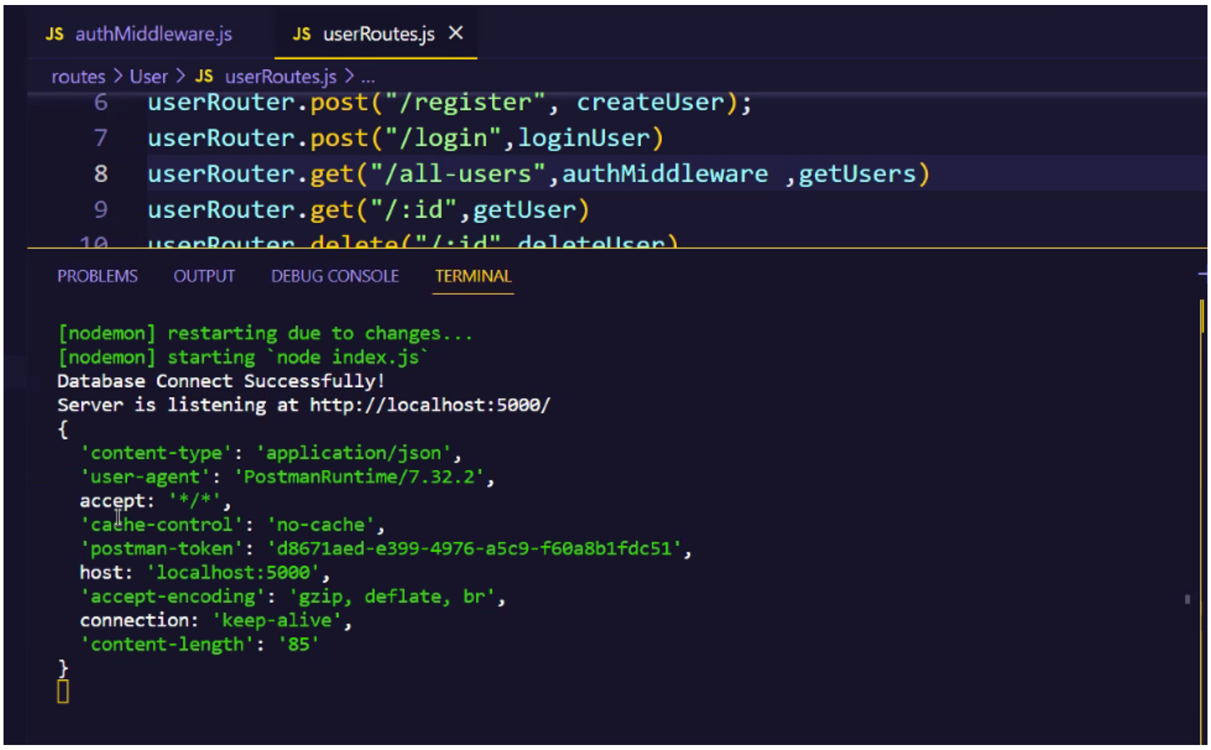
Lets learn Auth middleware

Inside middleware folder -create authMiddleware.js

We send token in headers



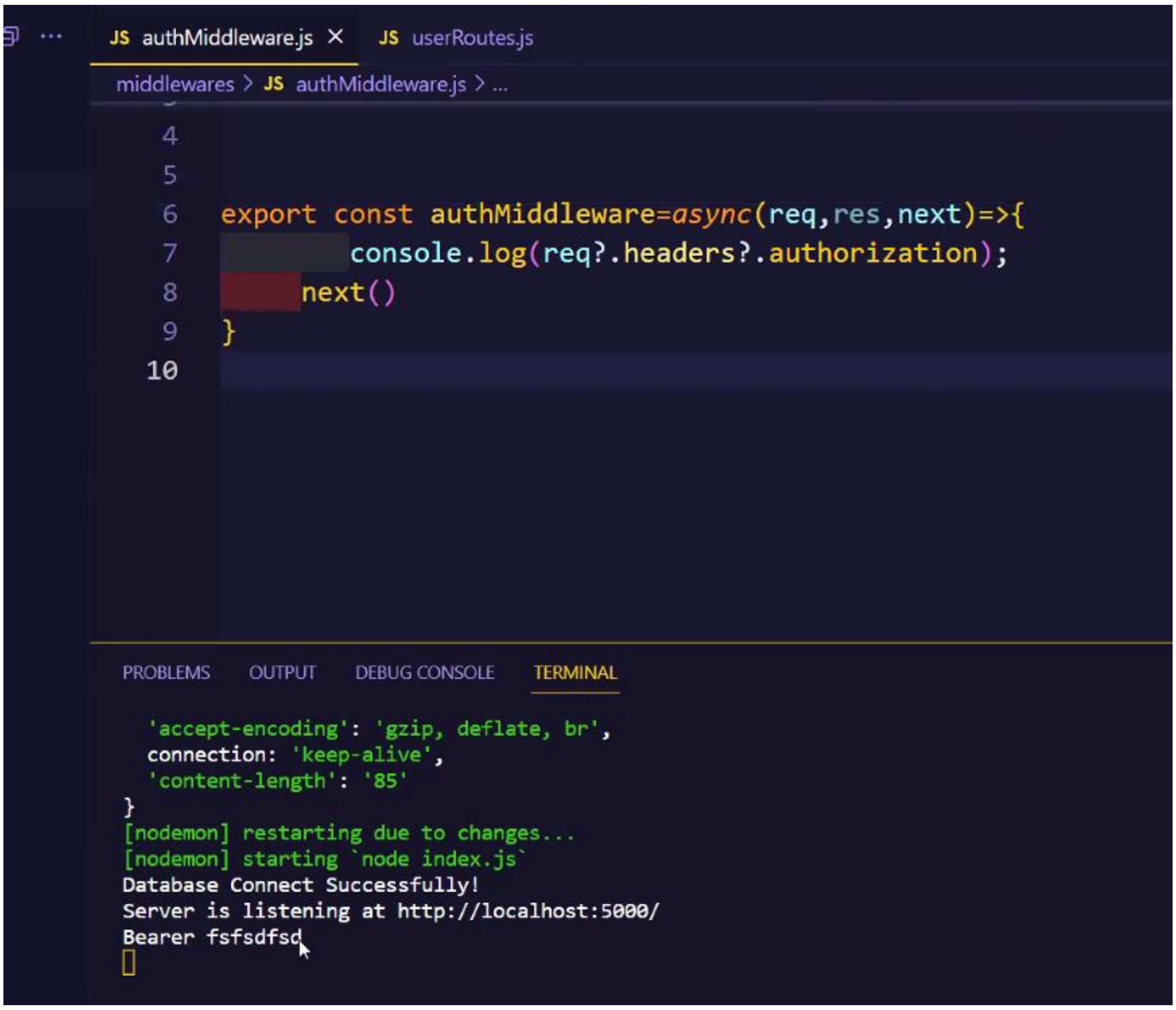
```javascript
import {User} from "../models/User/userModel.js"
import jwt from "jsonwebtoken"  110.2k (gzipped: 36.4k)



export const authMiddleware=async(req,res,next)=>{
        console.log(req?.headers);
    next()
}
```



```javascript
userRouter.post("/register", createUser);
userRouter.post("/login",loginUser)
userRouter.get("/all-users",authMiddleware ,getUsers)
userRouter.get("/:id",getUser)
userRouter.delete("/:id",deleteUser)
```

```
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Database Connect Successfully!
Server is listening at http://localhost:5000/
{
  'content-type': 'application/json',
  'user-agent': 'PostmanRuntime/7.32.2',
  accept: '*/*',
  'cache-control': 'no-cache',
  'postman-token': 'd8671aed-e399-4976-a5c9-f60a8b1fdc51',
  host: 'localhost:5000',
  'accept-encoding': 'gzip, deflate, br',
  connection: 'keep-alive',
  'content-length': '85'
}
```

We are passing middleware in a route above

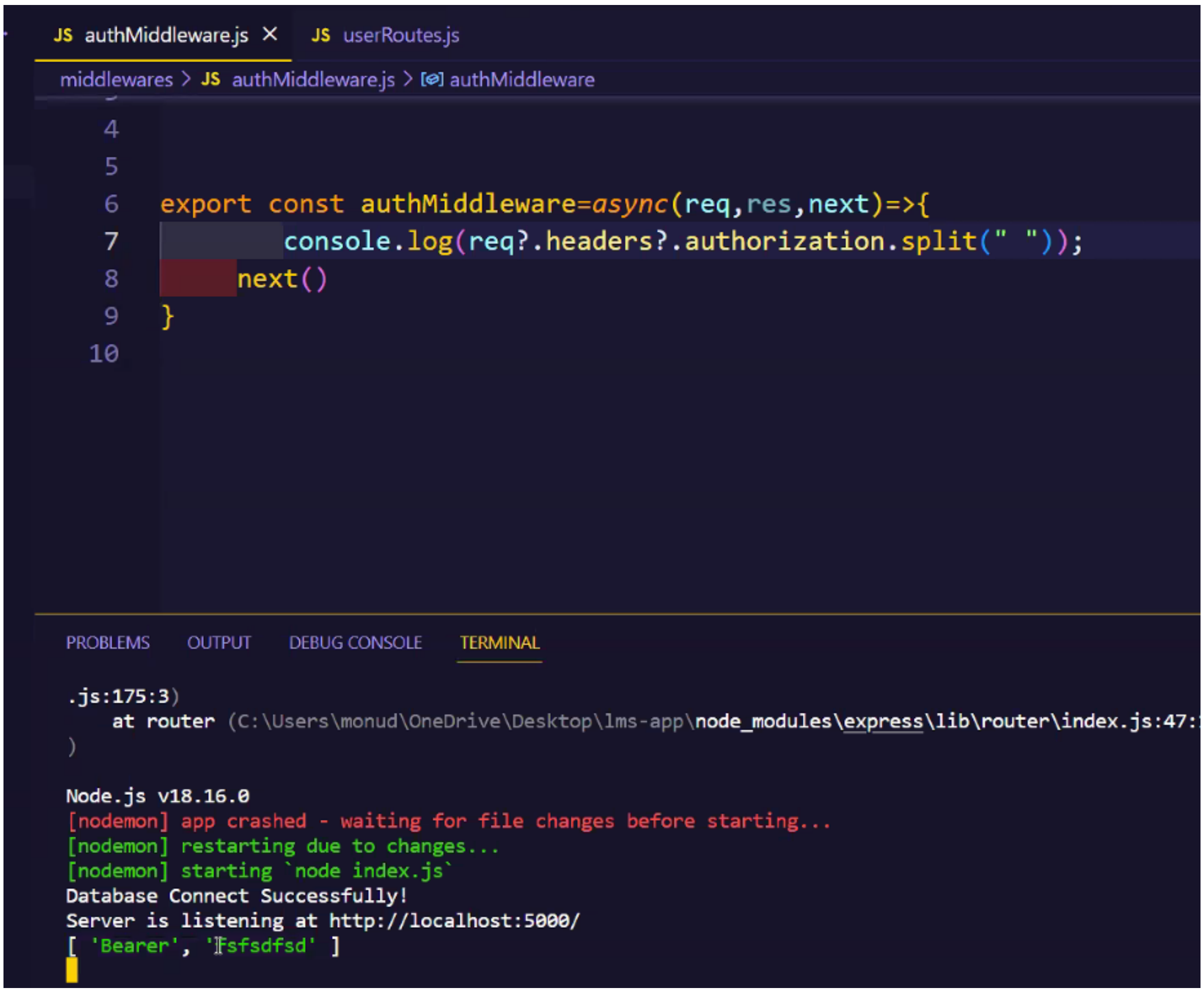


```javascript
export const authMiddleware=async(req,res,next)=>{
        console.log(req?.headers?.authorization);
    next()
}
```

```
  'accept-encoding': 'gzip, deflate, br',
  connection: 'keep-alive',
  'content-length': '85'
}
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Database Connect Successfully!
Server is listening at http://localhost:5000/
Bearer fsfsdfsd
```

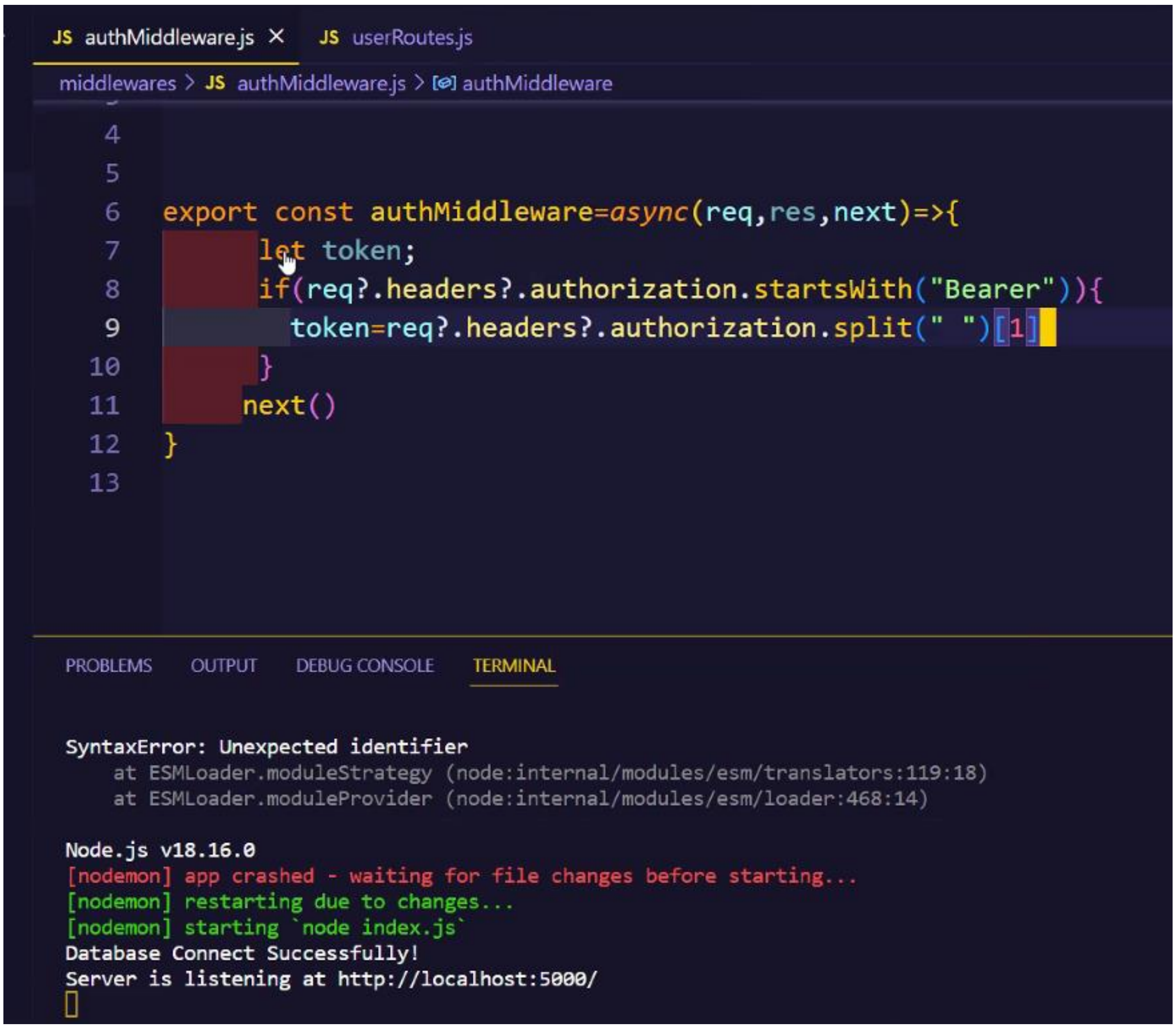
We sent the token from postman , we are getting it as we see above

We split the token , convert to array



```
export const authMiddleware=async(req,res,next)=>{
        console.log(req?.headers?.authorization.split(" "));
    next()
}
```

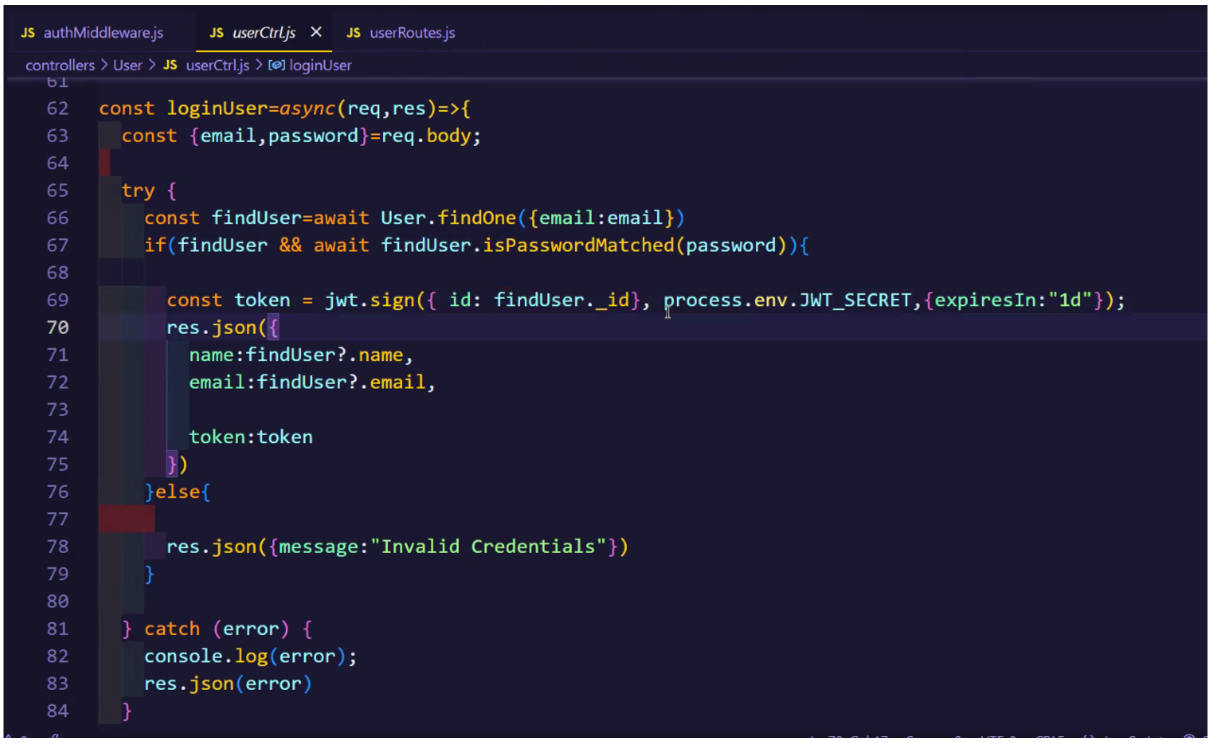
```
.js:175:3)
    at router (C:\Users\monud\OneDrive\Desktop\lms-app\node_modules\express\lib\router\index.js:47:
)

Node.js v18.16.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Database Connect Successfully!
Server is listening at http://localhost:5000/
[ 'Bearer', 'fsfsdfsd' ]
```



```javascript


export const authMiddleware=async(req,res,next)=>{
        let token;
        if(req?.headers?.authorization.startsWith("Bearer")){
          token=req?.headers?.authorization.split(" ")[1]
        }
      next()
}

```

```
SyntaxError: Unexpected identifier
    at ESMLoader.moduleStrategy (node:internal/modules/esm/translators:119:18)
    at ESMLoader.moduleProvider (node:internal/modules/esm/loader:468:14)

Node.js v18.16.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Database Connect Successfully!
Server is listening at http://localhost:5000/
```

We generate middleware as below

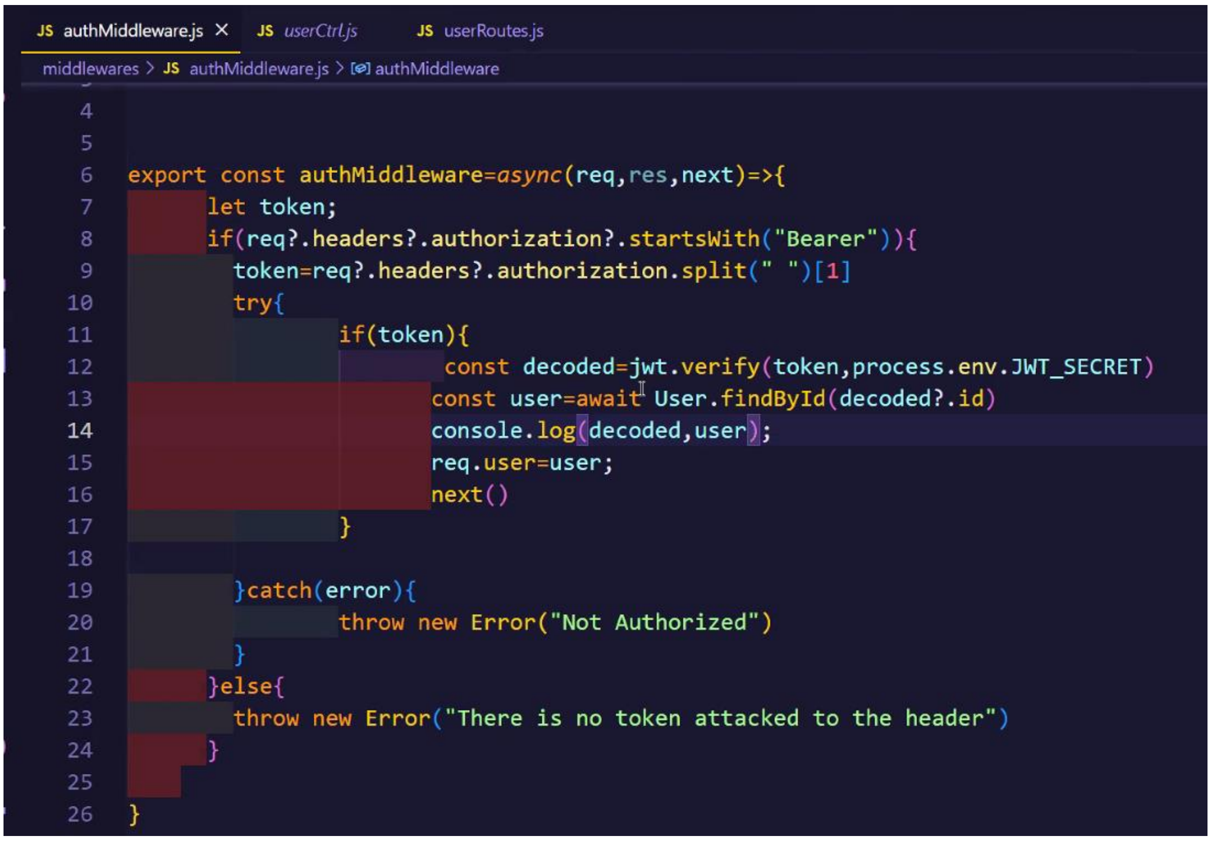


JS authMiddleware.js | JS userCtrl.js | JS userRoutes.js

controllers > User > JS userCtrl.js > loginUser

```
const loginUser=async(req,res)=>{
  const {email,password}=req.body;

  try {
    const findUser=await User.findOne({email:email})
    if(findUser && await findUser.isPasswordMatched(password)){

      const token = jwt.sign({ id: findUser._id}, process.env.JWT_SECRET,{expiresIn:"1d"});
      res.json({
        name:findUser?.name,
        email:findUser?.email,

        token:token
      })
    }else{

      res.json({message:"Invalid Credentials"})
    }

  } catch (error) {
    console.log(error);
    res.json(error)
  }
```

Then we use auth middleware



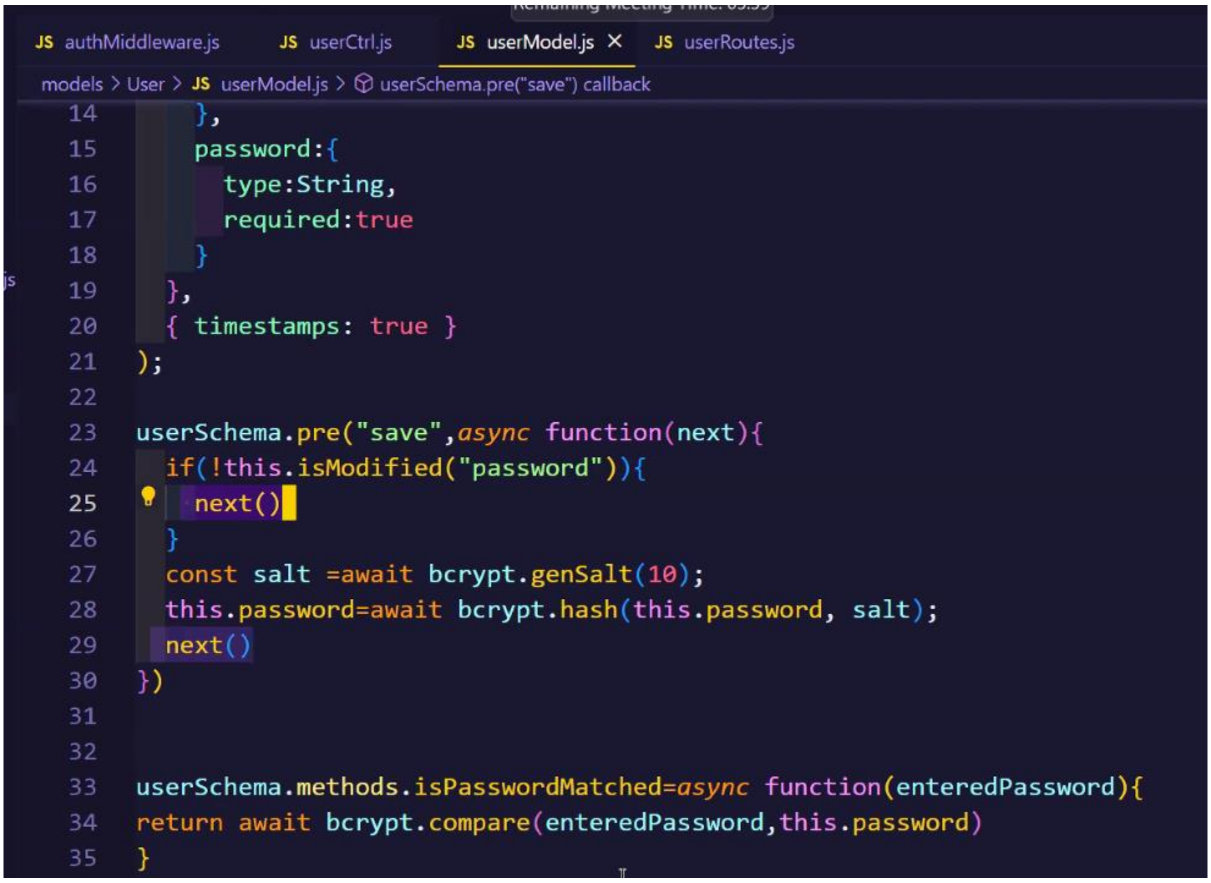
JS authMiddleware.js | JS userCtrl.js | JS userRoutes.js

middlewares > JS authMiddleware.js > authMiddleware

```


export const authMiddleware=async(req,res,next)=>{
       let token;
       if(req?.headers?.authorization?.startsWith("Bearer")){
         token=req?.headers?.authorization.split(" ")[1]
         try{
                 if(token){
                         const decoded=jwt.verify(token,process.env.JWT_SECRET)
                        const user=await User.findById(decoded?.id)
                        console.log(decoded,user);
                        req.user=user;
                        next()
                 }

         }catch(error){
                 throw new Error("Not Authorized")
         }
       }else{
         throw new Error("There is no token attacked to the header")
       }

}
```

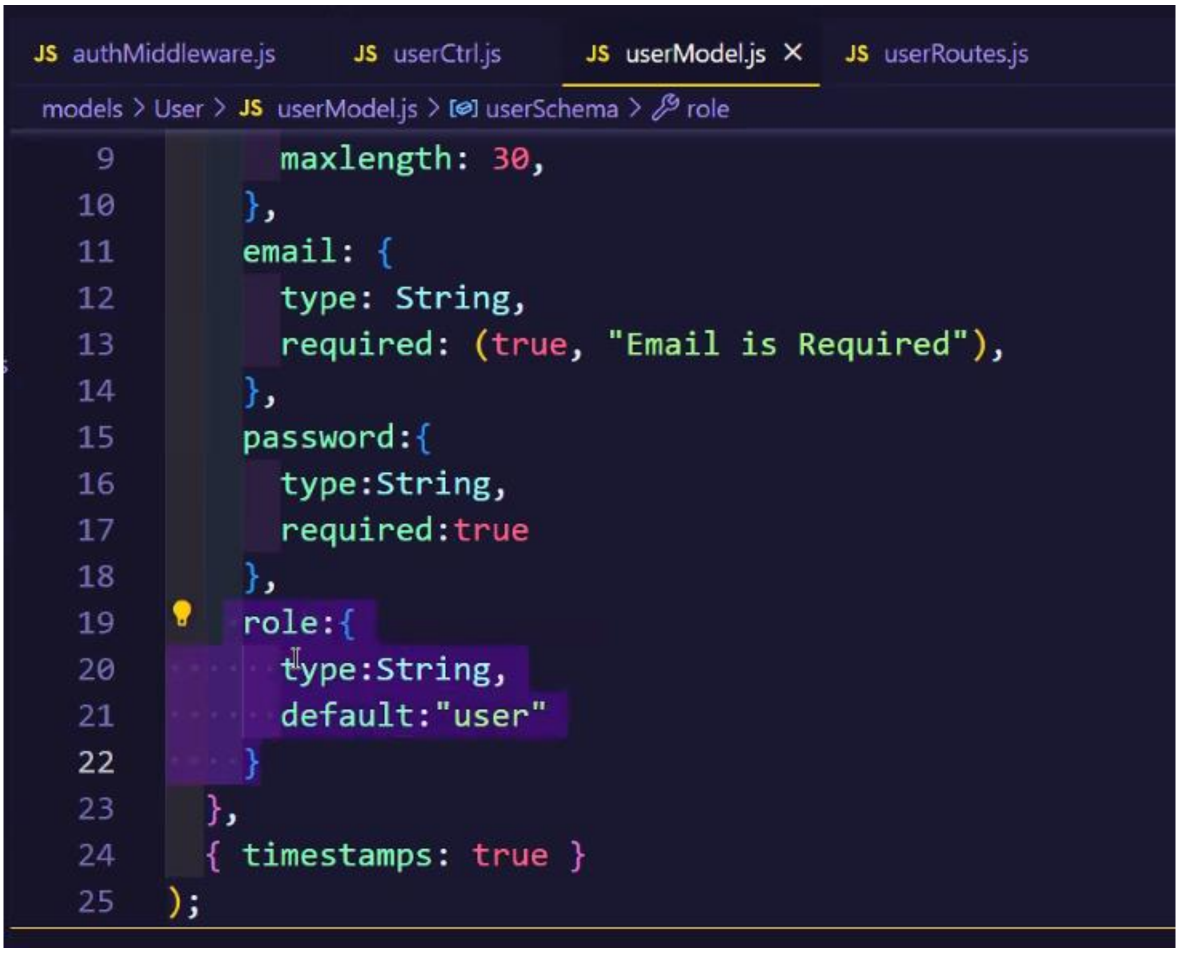
Where ever we want the user details , pass the authMiddleware in its route

When we are updating the details , then no need to hash again , then use below in the schema model

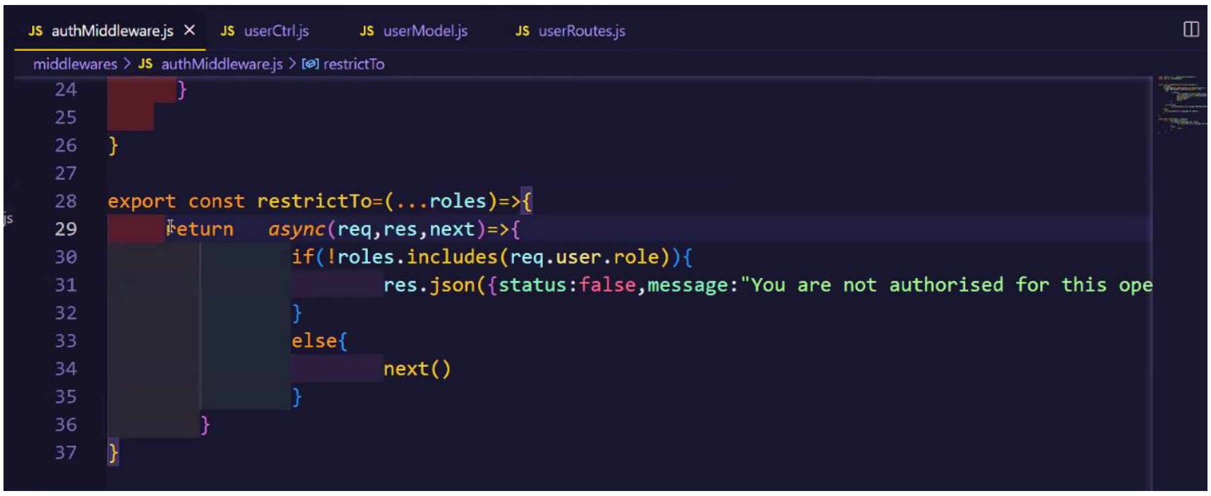

JS authMiddleware.js
JS userCtrl.js
JS userModel.js
JS userRoutes.js
models > User > JS userModel.js > userSchema.pre("save") callback

```
    },
    password:{
      type:String,
      required:true
    }
  },
  { timestamps: true }
);

userSchema.pre("save",async function(next){
  if(!this.isModified("password")){
    next()
  }
  const salt =await bcrypt.genSalt(10);
  this.password=await bcrypt.hash(this.password, salt);
  next()
})


userSchema.methods.isPasswordMatched=async function(enteredPassword){
return await bcrypt.compare(enteredPassword,this.password)
}
```

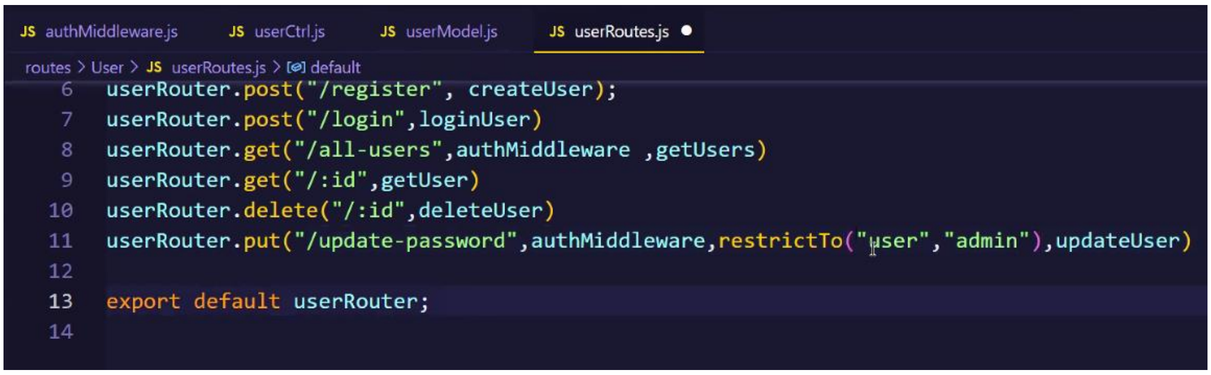
Now add a field in the userModel



```
      maxlength: 30,
    },
    email: {
      type: String,
      required: (true, "Email is Required"),
    },
    password:{
      type:String,
      required:true
    },
    role:{
      type:String,
      default:"user"
    }
  },
  { timestamps: true }
);
```



```
        }

}

export const restrictTo=(...roles)=>{
    return   async(req,res,next)=>{
                if(!roles.includes(req.user.role)){
                        res.json({status:false,message:"You are not authorised for this ope
                }
                else{
                        next()
                }
        }
}
```



```
userRouter.post("/register", createUser);
userRouter.post("/login",loginUser)
userRouter.get("/all-users",authMiddleware ,getUsers)
userRouter.get("/:id",getUser)
userRouter.delete("/:id",deleteUser)
userRouter.put("/update-password",authMiddleware,restrictTo("user","admin"),updateUser)

export default userRouter;
```

restrictTo – protected routes

authMiddleware