

getAll users from db

```
JS userController.js X JS userModel.js
APP > controllers > JS userController.js > getUsers
16     newUser,
17   });
18   } catch (err) {
19     res.status(500).json({ err: err.message });
20   }
21 };
22
23 const getUsers = async (req, res) => {
24   try {
25     const users = await User.find();
26     res.json(users);
27   } catch (err) {
28     res.json({
29       err: err.message,
30     });
31   }
32 };
33 export { createUser, getUsers };
34
```

Add the Route

```
JS userController.js JS userRoutes.js 1 X JS dbConnect.js
APP > routes > JS userRoutes.js > ...
You, 5 minutes ago | 1 author (You)
1   import express from 'express';
2
3   const userRouter = express.Router();
4
5   import { createUser, getUsers } from '../controllers/userController.js';
6
7   userRouter.post('/create-user', createUser);
8   userRouter.get('/all-users', getUsers);
9
10  export default userRouter;
11
```

getUser based on Id:

```
JS userController.js × JS userRoutes.js 1 JS dbConnect.js
APP > controllers > JS userController.js > [🔍] getUser
29     err: err.message,
30   });
31 }
32 };
33
34 const getUser = async (req, res) => {
35   try {
36     const { id } = req.params;
37     // const user = await User.findById(id);
38     const user = await User.findOne({ _id: id });
39     res.json(user);
40   } catch (err) {
41     res.json({ err: err.message });
42   }
43 };
44
45 export { createUser, getUsers, getUser };
46
```

Create route for getUser

```
JS userController.js JS userRoutes.js 1 X JS dbConnect.js
APP > routes > JS userRoutes.js > ...
You, 4 minutes ago | 1 author (You)
1 import express from 'express';
2
3 const userRouter = express.Router();
4
5 import {
6   createUser,
7   getUsers,
8   getUser,
9 } from '../controllers/userController.js';
10
11 userRouter.post('/create-user', createUser);
12 userRouter.get('/all-users', getUsers);
13 userRouter.get('/user/:id', getUser);
14 export default userRouter;
15
```

Deleteuser:

```
JS userController.js X JS userRoutes.js 1 JS dbConnect.js
APP > controllers > JS userController.js > deleteUser
41 } catch (err) {
42   res.json({ err: err.message });
43 }
44 };
45
46 const deleteUser = async (req, res) => {
47   try {
48     const id = req.params.id;
49     const deletedUser = await User.findByIdAndDelete(id);
50     res.json({
51       res: deletedUser,
52       msg: 'user deletion successful',
53     });
54   } catch (err) {
55     res.json({ err: err.message });
56   }
57 };
58 export { createUser, getUsers, getUser, deleteUser };
59
```

```
JS userController.js JS userRoutes.js 1 X JS dbConnect.js
APP > routes > JS userRoutes.js > ...
You, 5 minutes ago | 1 author (You)
1 import express from 'express';
2
3 const userRouter = express.Router();
4
5 import {
6   createUser,
7   getUsers,
8   getUser,
9   deleteUser,
10 } from '../controllers/userController.js';
11
12 userRouter.post('/create-user', createUser);
13 userRouter.get('/all-users', getUsers);
14 userRouter.get('/user/:id', getUser);
15 userRouter.delete('/user-delete/:id', deleteUser);
16 export default userRouter;
17
```

Update User:

```
JS userController.js X JS userRoutes.js 1 JS dbConnect.js
APP > controllers > JS userController.js > ...
55 res.json({ err: err.message });
56 }
57 };
58
59 const updateUser = async (req, res) => {
60   try {
61     const id = req.params.id;
62     const updatedUser = await User.findByIdAndUpdate(id, req.body, {
63       new: true,
64     });
65
66     res.json(updatedUser);
67   } catch (err) {
68     res.json(err);
69   }
70 };
71 export { createUser, getUsers, getUser, deleteUser, updateUser };
72
```

```
JS userController.js JS userRoutes.js 1 X JS dbConnect.js
APP > routes > JS userRoutes.js > ...
You, 3 minutes ago | 1 author (You)
1 import express from 'express';
2
3 const userRouter = express.Router();
4
5 import {
6   createUser,
7   getUsers,
8   getUser,
9   deleteUser,
10  updateUser,
11 } from '../controllers/userController.js';
12
13 userRouter.post('/create-user', createUser);
14 userRouter.get('/all-users', getUsers);
15 userRouter.get('/user/:id', getUser);
16 userRouter.delete('/user-delete/:id', deleteUser);
17 userRouter.put('/user-update/:id', updateUser);
18 export default userRouter;
19
```

Create Blog Schema

```
JS blogRoutes.js  JS index.js 1  JS blogController.js  JS blogModel.js 1 X
APP > models > JS blogModel.js > [?] blogSchema > [?] postedBy > [?] ref
1  import mongoose from 'mongoose';
2
3  const blogSchema = new mongoose.Schema(
4    {
5      title: {
6        required: true,
7        type: String,
8      },
9      desc: {
10        required: true,
11        type: String,
12      },
13      postedBy: {
14        type: mongoose.Schema.Types.ObjectId,
15        ref: 'User',
16        required: true,
17      },
18    },
19    { timestamps: true }
20  );
21
22  export const blog = mongoose.model('Blog', blogSchema);
```

```
JS blogRoutes.js  JS index.js 1  JS blogController.js X
APP > controllers > JS blogController.js > ...
1  import { blog } from '../models/blogModel.js';
2
3  export const postBlog = async (req, res) => {
4    try {
5      const post = await blog.create(req.body);
6      res.json(post);
7    } catch (err) {
8      res.json({ err: err.message });
9    }
10  };
11
```

```
JS blogRoutes.js X JS index.js 1 JS blogController.js
APP > routes > JS blogRoutes.js > ...
1 import express from 'express';
2
3 const blogRouter = express.Router();
4
5 import { postBlog } from '../controllers/blogController.js';
6
7 blogRouter.post('/post', postBlog);
8
9 export default blogRouter;
10
```

```
JS blogRoutes.js JS index.js 1 X JS blogController.js
APP > JS index.js > ...
1 import express from 'express';
2 import * as dotenv from 'dotenv';
3 import userRouter from './routes/userRoutes.js';
4 dotenv.config();
5 import dbConnect from './config/dbConnect.js';
6 import blogRouter from './routes/blogRoutes.js';
7 const PORT = process.env.PORT || 4000;
8 const app = express();
9 app.use(express.json());
10
11 dbConnect();
12
13 app.use('/api/v1', userRouter);
14 app.use('/api/v1', blogRouter);
15
16 app.get('/', (req, res) => {
17   res.send('Hello from server');
18 });
19
20 app.listen(PORT, () => {
21   console.log(`server is running at localhost://${PORT}`);
22 });
```

getBlog

```
JS blogRoutes.js JS index.js 1 JS blogController.js X JS blogModel.js 1
APP > controllers > JS blogController.js > [X] getBlog > [X] posts
1  import { blog } from '../models/blogModel.js';
2
3  export const postBlog = async (req, res) => {
4    try {
5      const post = await blog.create(req.body);
6      res.json(post);
7    } catch (err) {
8      res.json({ err: err.message });
9    }
10 };
11
12 export const getBlog = async (req, res) => {
13   try {
14     const posts = await blog.find().populate('postedBy');
15     res.json(posts);
16   } catch (err) {
17     res.json(err);
18   }
19 };
20
```

```
JS blogRoutes.js X JS index.js 1 JS blogController.js JS blogModel.js 1
APP > routes > JS blogRoutes.js > ...
1  import express from 'express';
2
3  const blogRouter = express.Router();
4
5  import { getBlog, postBlog } from '../controllers/blogController.js';
6
7  blogRouter.post('/post', postBlog);
8  blogRouter.get('/getposts', getBlog);
9  export default blogRouter;
10
```

Read
Update
Delete
relation

