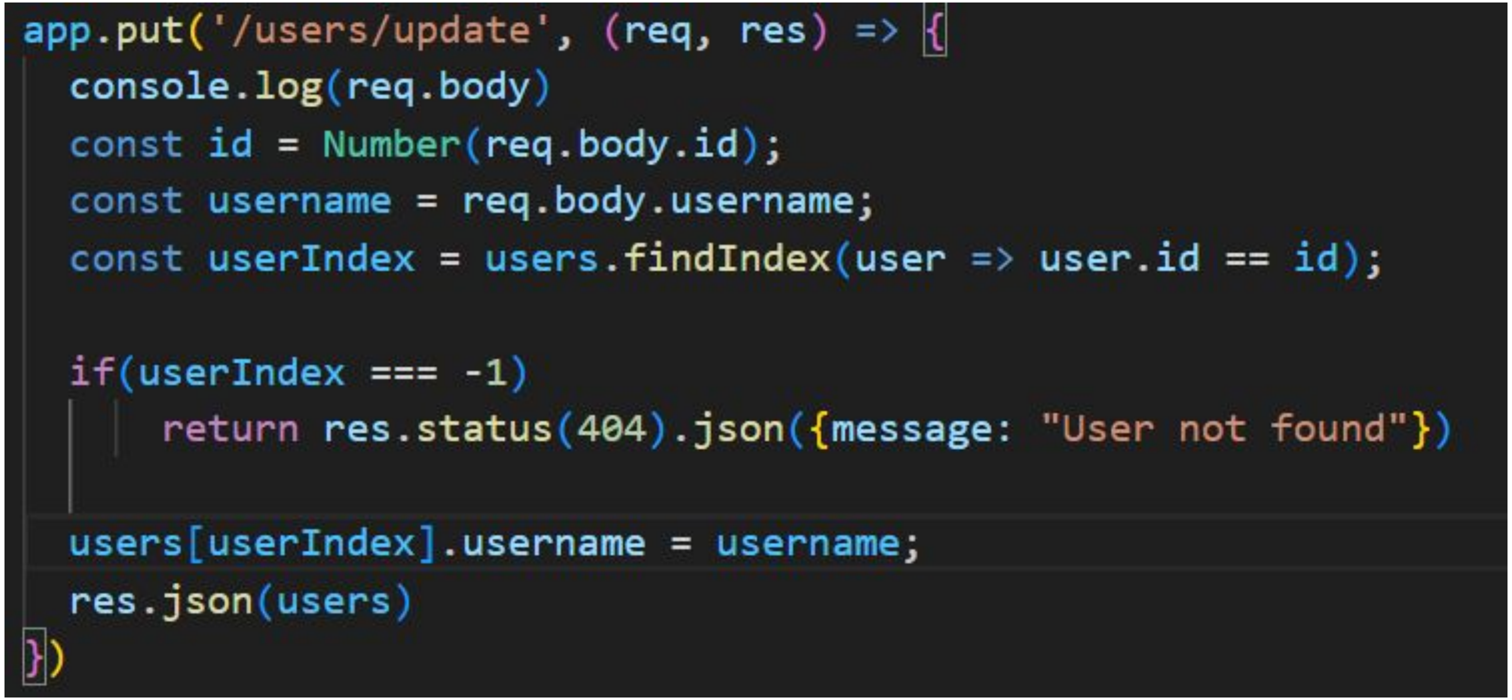
Express - 4

PUT, DELETE and handling routes

Create a PUT request to update users data

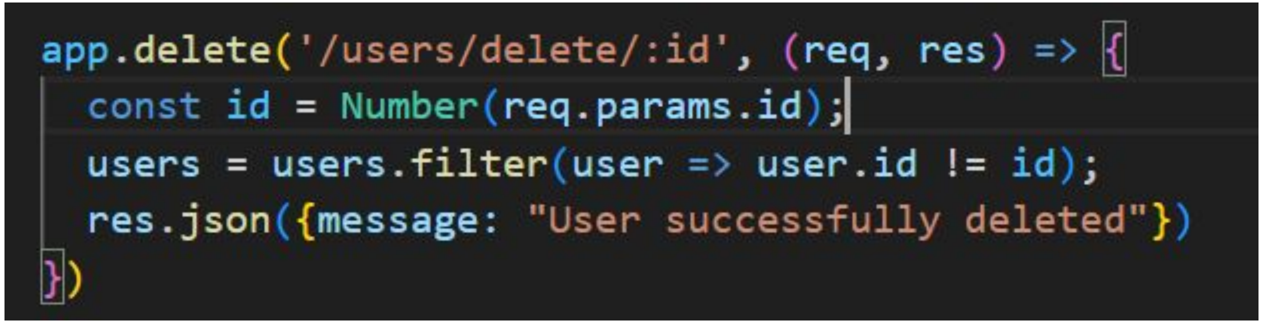


```
app.put('/users/update', (req, res) => {
  console.log(req.body)
  const id = Number(req.body.id);
  const username = req.body.username;
  const userIndex = users.findIndex(user => user.id == id);

  if(userIndex === -1)
      return res.status(404).json({message: "User not found"})

  users[userIndex].username = username;
  res.json(users)
})
```

Create a DELETE request to delete a user’s data



```
app.delete('/users/delete/:id', (req, res) => {
  const id = Number(req.params.id);
  users = users.filter(user => user.id != id);
  res.json({message: "User successfully deleted"})
})
```

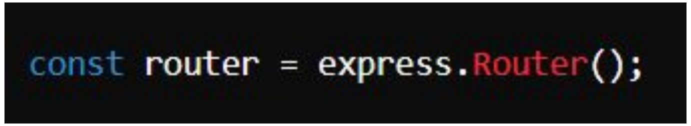
Building Modular and Scalable Routes with express.Router()

What is express.Router()

- **Definition:**
    - **express.Router()** is a class in Express.js that helps create modular, mountable route handlers.
    - It provides a way to organize routes in a more structured manner.
- **Key Features**
    - Segmentation or Splitting of route handling into modules.
    - Supports middleware just like the main express application object.
    - Enables cleaner and more maintainable code.

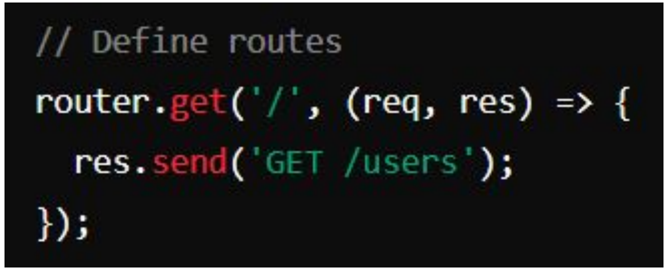
Steps to use express.Router()

1. In a new file create express.Router() object and save it in a variable



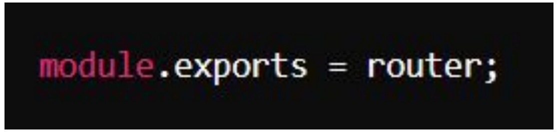
```javascript
const router = express.Router();
```

2. Now instead of app we will use this router variable to create routes.



```javascript
// Define routes
router.get('/', (req, res) => {
  res.send('GET /users');
});
```

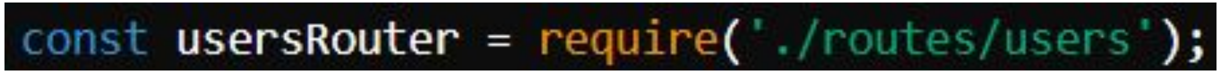
3. Now we will default export this router variable



```javascript
module.exports = router;
```

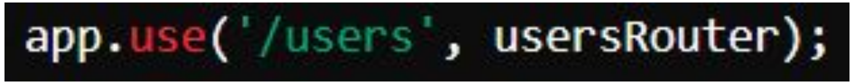
Mounting a router

1. Now import the exported router from the respective file



```
const usersRouter = require('./routes/users');
```

2. Now use the router mounting syntax to use the router



```
app.use('/users', usersRouter);
```

This will make the routes defined in usersRouter accessible under /users path.