**Code Snippet**

// postsController.js   
async function getSortedPosts(req, res) {   
const posts = await Posts.find(); // Mongoose model

posts.sort((a, b) => b.created - a.created); res.json(posts);

}

// router.js

router.get('/posts', async (req, res) => {

await getSortedPosts(req, res);

console.log('Done.');

});

1. Identify at least two problems (bugs or performance issues) here.

2. Propose corrections (code snippets or descriptions).

3. Explain in plain English why each fix matters.  
  
  
**Problem 1:** **Inefficient sorting in memory**

* posts.sort((a, b) => b.created - a.created);

This loads **all posts into memory**, **then** sorts them using JS.

**Bad for performance**: especially when the dataset is large.

**Problem 2: No error handling**

* const posts = await Posts.find();

No error handling here to handle the promise failure case properly, this should be written inside a try-catch block

Followings are the fixes;

1. Fix 1: Let MongoDB handle the sort efficiently

const posts = await Posts.find().sort({ created: -1 });

1. Fix 2: Add error handling

async function getSortedPosts(req, res) {

try {

const posts = await Posts.find().sort({ created: -1 });

res.json(posts);

} catch (err) {

console.error('Error fetching posts:', err);

res.status(500).json({ message: 'Server error' });

}

### }

* Why these fixes matters, explaination;

#### Sorting in the DB:

* **MongoDB is optimized for sorting**, especially with indexes (e.g., on created).
* Fetching pre-sorted results is **faster and memory-efficient**.
* Sorting in JavaScript means you're pulling unnecessary data into RAM.

#### Error handling:

* Without **try/catch**, any error (like DB connection issues) crashes the route.
* Catching errors ensures your app **responds gracefully** with a meaningful message and status code.

**Final Updated code**

// postsController.js

async function getSortedPosts(req, res) {

try {

const posts = await Posts.find().sort({ created: -1 });

res.json(posts);

} catch (err) {

console.error('Error fetching posts:', err);

res.status(500).json({ message: 'Server error' });

}

}

// router.js

router.get('/posts', getSortedPosts);