Training Day 11 Report

Date: 8 July 2025

Project: MERN Notes App

Topic Covered: Updating a Note by ID (PUT Route)

On the eleventh day, we extended the Notes API by adding the ability to update existing notes using their unique `_id`. This allows users to edit the title, content, or other properties of a note.

Understanding the PUT Request

- The client sends a **PUT** request to `/api/notes/:id` along with the updated data in the request body.
- The backend locates the note by ID, modifies its fields, and saves the updated version.

Implementing the Update Route

In `backend/routes/noteRoutes.js`, we add the following route handler:

Listing 1: PUT /api/notes/:id – Update an existing note

```
router.put('/:id', async (req, res) => {
  const { title, content } = req.body;

  try {
    const note = await Note.findById(req.params.id);

    if (!note) {
      return res.status(404).json({ message: 'Note not found' });
    }

    note.title = title || note.title;
    note.content = content || note.content;

    const updatedNote = await note.save();
    res.json(updatedNote);
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
});
```

Key points:

- Used `req.body` to extract updated fields from the request.
- Checked if the note exists before updating.
- Saved the modified note back to MongoDB and returned the updated object.

Testing the Endpoint

- Sent `PUT http://localhost:5000/api/notes/<noteID>` using Postman.

- Included a JSON body such as:

```
{
  "title": "Updated Note Title",
  "content": "This is the revised note content."
}
```

- Verified that the note's fields were successfully updated in the database.

Outcome of the Day

- Learned how to implement **PUT routes** in Express.
- Successfully updated notes by ID with new data.
- Backend now supports both reading and updating notes, moving closer to full CRUD functionality.