Create a file named server.js and set up your Express server along with MongoDB connection:

```
const express = require('express'); const
bodyParser = require('body-parser'); const
mongoose = require('mongoose');

const app = express(); const PORT =
process.env.PORT || 3000;

mongoose.connect('mongodb://localhost:27017/my_database', {
useNewUrlParser: true,     useUnifiedTopology: true
}).then(() => {     console.log('Connected to
MongoDB');
}).catch((err) => {     console.error('Error connecting to
MongoDB:', err);
});

app.use(bodyParser.json()); app.listen(PORT, () => {
console.log(`Server is running on http://localhost:${PORT}`);
});
```

Create a file named models/item.js to define the Mongoose schema for your items:

```
const mongoose = require('mongoose');
```

```
const itemSchema = new mongoose.Schema({
  name: { type: String, required: true },
description: { type: String }
});

module.exports = mongoose.model('Item', itemSchema);
```

In server.js, define routes for each CRUD operation:

```
app.post('/items', async (req, res) => {
  try {
    const { name, description } = req.body;    const newItem =
await Item.create({ name, description });
res.status(201).json(newItem);
  } catch (error) {    console.error('Error creating
item:', error);    res.status(500).json({ error: 'Error
creating item' });
  }
});
app.get('/items', async (req, res) => {
  try {
    const items = await Item.find();    res.json(items);
} catch (error) {    console.error('Error fetching items:',
error);    res.status(500).json({ error: 'Error fetching
items' });
  }
});
app.get('/items/:id', async (req, res) => {
try {
```

```
    const item = await Item.findById(req.params.id);
    if (!item) {
      return res.status(404).json({ error: 'Item not found' });
    }
    res.json(item);    } catch (error) {
console.error('Error fetching item:', error);
res.status(500).json({ error: 'Error fetching item' });
  }
});
app.put('/items/:id', async (req, res) => {
try {
    const { name, description } = req.body;       const updatedItem = await
Item.findByIdAndUpdate(req.params.id, { name, description }, { new: true });       if (!updatedItem)
{          return res.status(404).json({ error: 'Item not found' });
    }
    res.json(updatedItem);
  } catch (error) {       console.error('Error updating
item:', error);       res.status(500).json({ error: 'Error
updating item' });
  }
});
app.delete('/items/:id', async (req, res) => {
try {
    const deletedItem = await Item.findByIdAndDelete(req.params.id);
if (!deletedItem) {          return res.status(404).json({ error: 'Item not
found' });
    }
    res.sendStatus(204);
  } catch (error) {
```

```
        console.error('Error deleting item:', error);
res.status(500).json({ error: 'Error deleting item' });
    }
});
```