Database connection with mongodb

```
const pool = require('../config/db');const path = require('path');const fs = require('fs');const PDFDocument = require('pdfkit');

const studyFolderController = {  getStudyMaterials: async (req, res) => {    try {      const [materials] = await pool.execute(        'SELECT * FROM notes ORDER BY subject_name, created_at DESC'      );      res.json(materials);    } catch (error) {      res.status(500).json({ message: error.message });    }  },

  savePDFToStudyFolder: async (req, res) => {    try {      const { noteId } = req.params;      const [note] = await pool.execute(        'SELECT * FROM notes WHERE id = ?',        [noteId]      );

      if (!note[0]) {        return res.status(404).json({ message: 'Note not found' });      }

      const { subject_name, title, content } = note[0];            // Create directories if they don't exist      const studyDir = path.join(__dirname, `../uploads/study/${subject_name}`);      fs.mkdirSync(studyDir, { recursive: true });

      // Clean content by removing HTML tags and entities      let cleanContent = content        .replace(/<[^>]*>/g, '') // Remove HTML tags        .replace(/&amp;nbsp;/g, ' ') // Replace &amp;nbsp; with space        .replace(/&amp;amp;/g, '&') // Replace &amp;amp; with &        .replace(/&amp;lt;/g, '<') // Replace &lt; with <        .replace(/&amp;gt;/g, '>') // Replace &gt; with >        .replace(/&amp;quot;/g, '"') // Replace &quot; with "        .replace(/&amp;#39;/g, "'") // Replace &#39; with '        .trim(); // Remove extra whitespace

      // Generate PDF      const pdfPath = path.join(studyDir, `${title}.pdf`);      const doc = new PDFDocument({        margins: {          top: 50,          bottom: 50,          left: 50,          right: 50        }      });      const stream = fs.createWriteStream(pdfPath);

      doc.pipe(stream);            // Add title      doc.fontSize(24)        .font('Helvetica-Bold')        .text(title, {          align: 'center',          underline: true        });      doc.moveDown(2);            // Add content with proper formatting      doc.fontSize(12)        .font('Helvetica')        .text(cleanContent, {          align: 'left',          lineGap: 7,          paragraphGap: 10,          width: 500        });          doc.end();
```

```
    // Wait for PDF to finish writing    stream.on('finish', () => {      const relativePath = `/uploads/study/${subject_name}/${title}.pdf`;      res.json({ filePath: relativePath });    });

  } catch (error) {    res.status(500).json({ message: error.message });  }  },

  searchMaterials: async (req, res) => {  try {    const { searchTerm } = req.query;    const [materials] = await pool.execute(      'SELECT * FROM notes WHERE subject_name LIKE ? OR title LIKE ?',      [`%${searchTerm}%`, `%${searchTerm}%`]    );    res.json(materials);  } catch (error) {    res.status(500).json({ message: error.message });  }};

module.exports = studyFolderController;
```