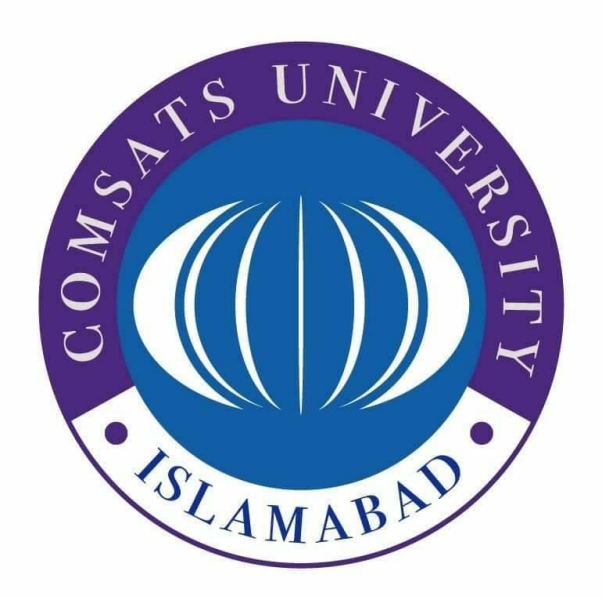


LAB ASSINGMENT NO 04

Advanced Web Technologies

Name: Uzair Ahmed

Reg No: SP23-BSE-053

FS Module:

```
// path
console.log('Current file:', __filename);
console.log('Current directory:', __dirname);
console.log('Current working dir:', process.cwd());

try {
  process.chdir(__dirname);
  console.log('Changed cwd to:', process.cwd());
} catch (err) {
  console.error('chdir error:', err.message);
}
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
Current file: C:\Users\sp23-bse-005\Desktop\ws\index.js
Current directory: C:\Users\sp23-bse-005\Desktop\ws
Current working dir: C:\Users\sp23-bse-005\Desktop\ws
Changed cwd to: C:\Users\sp23-bse-005\Desktop\ws
```

```
const fs = require('fs');
const path = require('path');
const file = path.join(__dirname, 'example.txt');

// Async
fs.readFile(file, { encoding: 'utf8' }, (err, data) => {
  if (err) return console.error('Async read error:', err.message);
  console.log('Async contents:\n', data);
});
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
Async contents:
 Hello! This is a new file.
```

```
const fs = require('fs');
const path = require('path');
const file = path.join(__dirname, 'example.txt');

// Sync
try {
  const data = fs.readFileSync(file, 'utf8');
  console.log('Sync contents:\n', data);
} catch (err) {
  console.error('Sync read error:', err.message);
}
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
Sync contents:
 Hello! This is a new file.
```

```
const fs = require('fs');
const path = require('path');
const file = path.join(__dirname, 'example.txt');

fs.open(file, 'w', (err, fd) => {
  if (err) return console.error('fs.open error:', err.message);
  console.log('File descriptor:', fd);
  fs.close(fd, () => {});
});

// writeFile
fs.writeFile(file, 'Hello world\n', (err) => {
  if (err) return console.error('writeFile error:', err.message);
  console.log('writeFile: wrote initial content');
});
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
File descriptor: 3
writeFile: wrote initial content
```

```
const fs = require('fs');
const path = require('path');
const file = path.join(__dirname, 'example.txt');

fs.open(file, 'w', (err, fd) => {
    if (err) return console.error('fs.open error:', err.message);
    console.log('File descriptor:', fd);
    fs.close(fd, () => { });
});

// appendFile (adds to end)
fs.appendFile(file, 'Appended line\n', (err) => {
    if (err) return console.error('appendFile error:', err.message);
    console.log('appendFile: appended content');
});
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
File descriptor: 3
appendFile: appended content
```

```
// write-wx.js
const fs = require('fs');
const file = __dirname + '/unique.txt';

fs.writeFile(file, 'first content', { flag: 'wx' }, (err) => {
  if (err) {
    if (err.code === 'EEXIST') return console.error('File already exists - not
overwriting');
    return console.error('writeFile error:', err);
  }
  console.log('Created unique.txt successfully');
});
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
Created unique.txt successfully
```

```
// delete-rename.js
const fs = require('fs');
const path = require('path');
const oldName = path.join(__dirname, 'example.txt');
const newName = path.join(__dirname, 'comsats.txt');

// Rename
fs.rename(oldName, newName, (err) => {
  if (err) return console.error('rename error:', err.message);
  console.log('File renamed to', newName);

});
```

Output:

```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
File renamed to C:\Users\sp23-bse-005\Desktop\ws\comsats.txt
```

```
const fs = require('fs');
const path = require('path');

const newName = path.join(__dirname, 'comsats.txt');

  // Delete
  fs.unlink(newName, (err) => {
    if (err) return console.error('unlink error:', err.message);
    console.log('File deleted:', newName);
  });
```

Output:

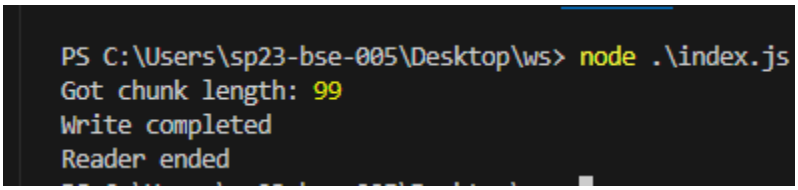
```
PS C:\Users\sp23-bse-005\Desktop\ws> node .\index.js
File deleted: C:\Users\sp23-bse-005\Desktop\ws\comsats.txt
```

```
// streams-read-write.js
const fs = require('fs');
const inFile = __dirname + '/largeInput.txt';
const outFile = __dirname + '/largeOutput.txt';


// // Writable stream
const writer = fs.createWriteStream(outFile);
writer.write('Start of file\n');
writer.end('Final line\n');
writer.on('finish', () => console.log('Write completed'));
writer.on('error', err => console.error('Writer error:', err));


// Readable stream
const reader = fs.createReadStream(inFile, { encoding: 'utf8', highWaterMark: 64
* 1024 });
reader.on('data', chunk => {
  console.log('Got chunk length:', chunk.length);
});
reader.on('end', () => console.log('Reader ended'));
reader.on('error', err => console.error('Reader error:', err));
```

Output:



Mongo DB with Mongoose:

Define Schema:

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

const url = 'mongodb://localhost:27017/testdb';

// Define a schema for files
const fileSchema = new mongoose.Schema({
  filename: { type: String, required: true, unique: true },
  content: { type: String, required: true }
```

```
});
```

Model Creation and Read files from local file system:

```
// Create a model
const File = mongoose.model('File', fileSchema);

// Read file from local filesystem (promisified)
function readFileAsync(path, encoding) {
  return new Promise((resolve, reject) => {
    fs.readFile(path, encoding, (err, data) => {
      if (err) reject(err);
      else resolve(data);
    });
  });
}
```

Read Files(Async) and Insert into Mongodb:

```
async function run() {
  try {
    const data = await readFileAsync('./example.txt', 'utf8');

    await mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology: true
});
    console.log('Connected successfully to MongoDB');

    // Insert the file document
    const doc = new File({ filename: 'example.txt', content: data });
    const savedDoc = await doc.save();

    console.log('File content inserted with _id:', savedDoc._id);

    await mongoose.connection.close();

    return true;
  } catch (err) {
    console.error('Error:', err.message || err);
    await mongoose.connection.close();
    return false;
  }
```

```
}
```

Output:

```
Connected successfully to MongoDB
File content inserted with _id: 68e0d8d4e9b192010c874acc

--- File Inserted Successfully ---
```

Read File form Mongo:

```
async function readFileFromMongo(filename) {
  try {
    await mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology: true
});

    const fileDoc = await File.findOne({ filename });

    if (fileDoc) {
      console.log(`\n[READ] Content of '${filename}':\n${fileDoc.content}\n`);
    } else {
      console.log(`\n[READ] File '${filename}' not found in MongoDB\n`);
    }

    await mongoose.connection.close();
  } catch (err) {
    console.error('Error:', err.message || err);
    await mongoose.connection.close();
  }
}
```

Output:

```
--- Reading original file ---


[READ] Content of 'example.txt':
Software Engineering Students
```

Renaming Files:

```
```

```
async function renameFile(oldFilename, newFilename) {
  try {
    await mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology: true
});

    const result = await File.updateOne(
      { filename: oldFilename },
      { filename: newFilename }
    );

    if (result.matchedCount === 0 && result.modifiedCount === 0) {
      console.log(`\n[RENAME] File '${oldFilename}' not found.\n`);
    } else if (result.modifiedCount === 1) {
      console.log(`\n[RENAME] File renamed from '${oldFilename}' to
'${newFilename}'.\n`);
    } else {
      console.log('\n[RENAME] No changes made.\n');
    }

    await mongoose.connection.close();
  } catch (err) {
    console.error('Error renaming file:', err.message || err);
    await mongoose.connection.close();
  }
}
```

Output:

```
--- Renaming file ---


[RENAME] No changes made.


--- Reading renamed file ---


[READ] Content of 'renamed_example.txt':
Software Engineering Students
```

Deleting files in Mongodb Collections:

```
async function deleteFile(filename) {
  try {
```

```
    await mongoose.connect(url, { useNewUrlParser: true, useUnifiedTopology: true
});

    const result = await File.deleteOne({ filename });

    if (result.deletedCount === 1) {
      console.log(`\n[DELETE] File '${filename}' deleted successfully.\n`);
    } else {
      console.log(`\n[DELETE] File '${filename}' not found.\n`);
    }

    await mongoose.connection.close();
  } catch (err) {
    console.error('Error deleting file:', err.message || err);
    await mongoose.connection.close();
  }
}
```

Output:

```
--- Deleting renamed file ---


[DELETE] File 'renamed_example.txt' deleted successfully.


--- Trying to read deleted file ---


[READ] File 'renamed_example.txt' not found in MongoDB
```

Main Function:

```
async function main() {
  console.log('--- Starting file insertion ---\n');
  const inserted = await run();
  if (inserted) {
    console.log('\n--- File Inserted Successfully ---\n');

    console.log('--- Reading original file ---\n');
    await readFileFromMongo('example.txt');
```

```
    console.log('\n--- Renaming file ---\n');
    await renameFile('example.txt', 'renamed_example.txt');

    console.log('\n--- Reading renamed file ---\n');
    await readFileFromMongo('renamed_example.txt');

    console.log('\n--- Deleting renamed file ---\n');
    await deleteFile('renamed_example.txt');

    console.log('\n--- Trying to read deleted file ---\n');
    await readFileFromMongo('renamed_example.txt');

    console.log('\n--- All operations complete ---\n');
  } else {
    console.log('File insertion failed, aborting further operations.');
  }
}

main();
```
