

Class Assignment

Name: Gaurav Deshmukh

USN: 72232433C

Code :

(1) convert text into uppercase from input file to output file.

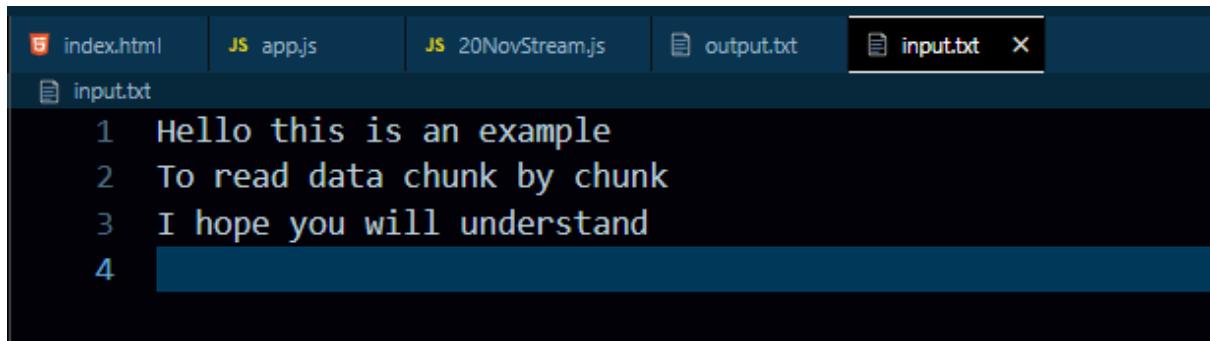
The screenshot shows a code editor interface with several tabs at the top: index.html, app.js, 20NovStream.js (which is the active tab), output.txt, and input.txt. The 20NovStream.js tab contains the following code:

```
174 const fs=require('fs')
175 const {Transform}= require('stream')
176
177 class TransformUppercase extends Transform{
178
179   constructor(){
180     super();
181   }
182
183   _transform(chunk, encoding, callback){
184     let upperCaseData = chunk.toString().toUpperCase();
185     this.push(upperCaseData);
186     callback();
187   }
188 }
189
190 const inputFile='input.txt'
191 const outFile='output.txt'
192
193 const readStream=fs.createReadStream(inputFile,{encoding:'utf8'})
194 const transformStream= new TransformUppercase()
195 const writeStream=fs.createWriteStream(outFile)
196
197 readStream
198 .pipe(transformStream)
199 .pipe(writeStream)
200 .on('finish', () => {
201   console.log('All text converted to uppercase successfully!');
202 });
203
```

Below the code editor, there are tabs for OUTPUT, DEBUG CONSOLE, and PORTS, with OUTPUT being selected. At the bottom, there is a terminal window with the following content:

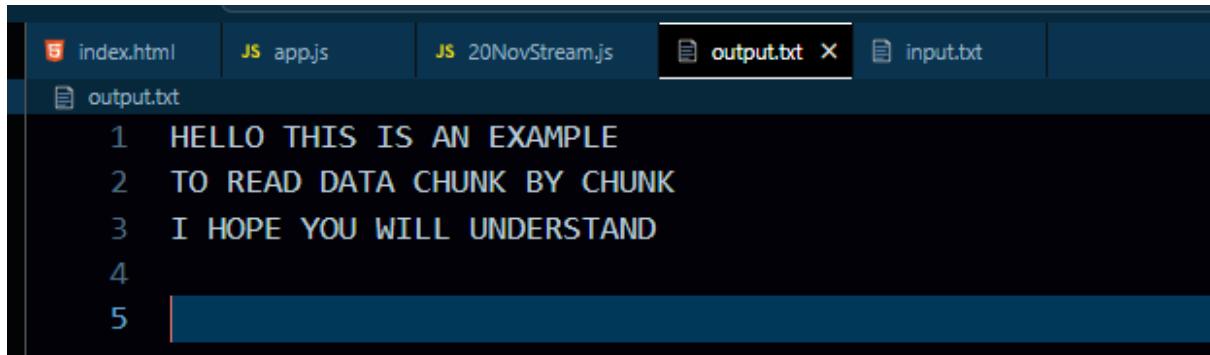
- PS C:\Users\gaura\OneDrive\Desktop\FINACLE TRAINING\Node.js and V8JS\Day 17 19-11-2025> node 20NovStream
 All text converted to uppercase successfully!
- PS C:\Users\gaura\OneDrive\Desktop\FINACLE TRAINING\Node.js and V8JS\Day 17 19-11-2025> []

Input File:



```
index.html app.js 20NovStream.js output.txt input.txt
input.txt
1 Hello this is an example
2 To read data chunk by chunk
3 I hope you will understand
4
```

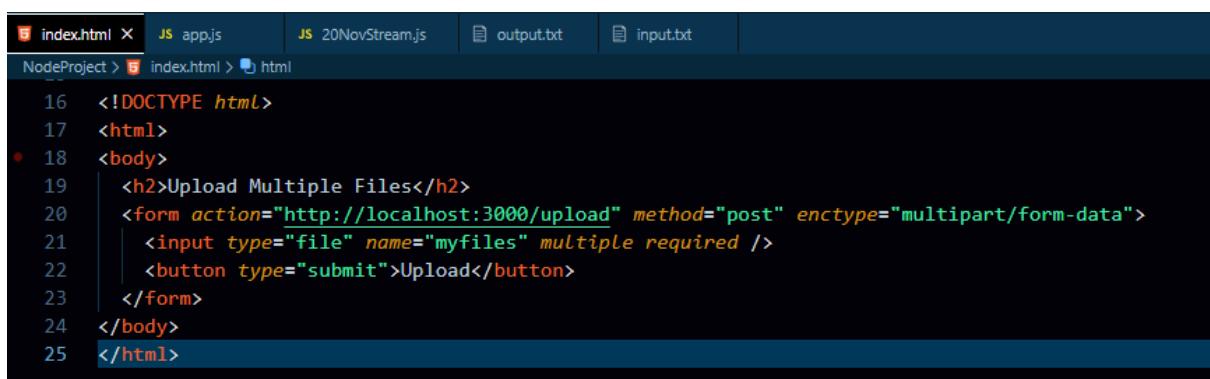
Output File:



```
index.html app.js 20NovStream.js output.txt input.txt
output.txt
1 HELLO THIS IS AN EXAMPLE
2 TO READ DATA CHUNK BY CHUNK
3 I HOPE YOU WILL UNDERSTAND
4
5
```

(2) code for uploading multiple files

Index.html:



```
index.html app.js 20NovStream.js output.txt input.txt
NodeProject > index.html
16  <!DOCTYPE html>
17  <html>
18  <body>
19  | <h2>Upload Multiple Files</h2>
20  | <form action="http://localhost:3000/upload" method="post" enctype="multipart/form-data">
21  | | <input type="file" name="myfiles" multiple required />
22  | | <button type="submit">Upload</button>
23  | </form>
24  </body>
25  </html>
```

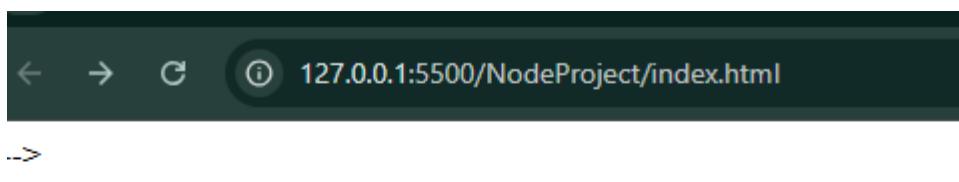
App.js:

The screenshot shows a code editor with the file 'app.js' open. The code implements a Node.js application using Express and Multer to handle file uploads. It sets up a static file server for uploads, defines a Multer disk storage strategy, and handles POST requests to '/upload' to upload multiple files. It also handles GET requests to '/download/:filename' to download files. The terminal below shows the application running at port 3000 and successfully uploading two files.

```
56 const app = express();
57 const PORT = 3000;
58
59 app.use('/uploads', express.static(path.join(__dirname, 'uploads')));
60
61 const storage = multer.diskStorage({
62   destination: (req, file, cb) => cb(null, 'uploads/'),
63   filename: (req, file, cb) => cb(null, Date.now() + path.extname(file.originalname))
64 });
65
66 const upload = multer({ storage });
67
68 app.post('/upload', upload.array('myfiles', 10), (req, res) => {
69   if (!req.files || req.files.length === 0) {
70     return res.status(400).send('No files were uploaded.');
71   }
72
73   console.log(`${req.files.length} files uploaded.`);
74
75   let responseHtml = '<h2>Files uploaded successfully!</h2><ul>';
76
77   req.files.forEach(file => {
78     responseHtml += `<li><a href="/uploads/${file.filename}">${file.originalname}</a> (${file.filename})</li>`;
79     console.log(`File uploaded: ${file.filename}`);
80   });
81
82   responseHtml += '</ul>';
83   res.send(responseHtml);
84 });
85
86 app.get('/download/:filename', (req, res) => {
87   const filePath = path.join(__dirname, 'uploads', req.params.filename);
88   res.download(filePath, (err) => {
89     if (err) {
90       console.error('Error downloading:', err);
91       res.status(500).send('Error in download.');
92     }
93   });
94 });

OUTPUT DEBUG CONSOLE PORTS
TERMINAL
PS C:\Users\gaura\OneDrive\Desktop\FINACLE TRAINING\Node.js and VBJS\Day 17 19-11-2025\NodeProject> node app
Server running at http://localhost:3000
2 files uploaded.
File uploaded: 1763705203153.pdf
File uploaded: 1763705203168.jpg
```

Output:



Upload Multiple Files

Choose Files 2 files

← → ⌂

localhost:3000/upload

Files uploaded successfully!

- [EWS.pdf](#) (1763705203153.pdf)
- [passport photo.jpg](#) (1763705203160.jpg)