Day04 Logs 04

part 1 – Install Morgan

1. Copy the folder called **Day04-Deployment-01** from **Day04** paste it and rename it to **Day04-Logs-04**.
2. Run **mpn install** **FIRST** on the new folder then run the following command to install Morgan

**npm install morgan --save**

1. Require morgan in the http\_server.js file

|  |
| --- |
| **const express = require('express');**  **const bodyParser= require('body-parser');**  **const Morgan = require('morgan');**  **const port = 8000;**  **const app = express();**  **app.use(bodyParser.urlencoded({extended:false}));** |

1. Since Morgan requires access to the file system, lets include that as well in the main file

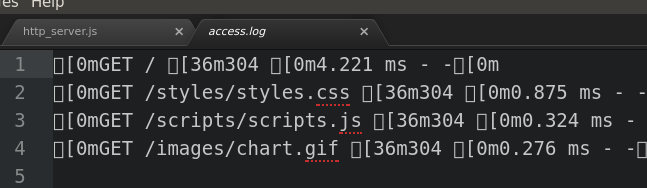
|  |
| --- |
| **const express = require('express');**  **const bodyParser= require('body-parser');**  **const Morgan = require('morgan');**  **const fs = require('fs');**  **const path = require('path');**  **const port = 8000;**  **const app = express();** |

1. Now we can use the Morgan module to create a morgan object and create a log file.

|  |
| --- |
| **const Morgan = require('morgan');**  **const fs = require('fs');**  **const path = require('path');**  **const m = new Morgan(**  **'dev',**  **{stream:fs.createWriteStream('./access.log')}**  **);** |

1. Connect our app to morgan, but do this **after** the app object is created.

|  |
| --- |
| **const port = 8000;**  **const app = express();**  **app.use(bodyParser.urlencoded({extended:false}));**  **app.use(m);**  **const router = express.Router();** |

1. Run the application, you should now see an empty file called access.log in the file system of your OS. See image below:  
   

part 2 – Separate Morgan

1. Create a new .js file inside of the controllers folder, call it morgan\_logs.js
2. Add the following lines of code to this new file. Notice that it is the same lines in the http\_server.js file

|  |
| --- |
| **const Morgan = require('morgan');**  **const fs = require('fs');**  **const path = require('path');**  **const m = new Morgan(**  **'dev',**  **{stream:fs.createWriteStream('./access.log')}**  **);** |

1. Now back in the http\_server file, remove the corresponding lines

|  |
| --- |
| **const express = require('express');**  **const bodyParser= require('body-parser');**  **const port = 8000;**  **const app = express();**  **app.use(bodyParser.urlencoded({extended:false}));**  **const router = express.Router();**  **const routes = require('./routes/routes');**  **routes(app);**  **//**  **app.listen(port, () => console.log("Listening " + port));** |

1. Back in the morgan\_logs.js file, add the exports line to make this function visible to other files

|  |
| --- |
| **const Morgan = require('morgan');**  **const fs = require('fs');**  **const path = require('path');**  **const m = new Morgan(**  **'dev',**  **{stream:fs.createWriteStream('./access.log')}**  **);**  **module.exports = m;** |

1. In the http\_server.js file, require the new morgan\_logs.js file

|  |
| --- |
| **const express = require('express');**  **const bodyParser= require('body-parser');**  **const port = 8000;**  **const app = express();** |

1. Now we can use morgan as an object and pass it to app as middleware

|  |
| --- |
| **const express = require('express');**  **const bodyParser= require('body-parser');**  **const morgan = require('./controllers/morgan\_logs');**  **const port = 8000;**  **const app = express();**  **app.use(bodyParser.urlencoded({extended:false}));**  **app.use(morgan);**  **const router = express.Router();**  **const routes = require('./routes/routes');**  **routes(app);**  **//**  **app.listen(port, () => console.log("Listening " + port));** |

This is the entire http\_server.js file

1. Test the app, it should function the same as before but now all logging details have been separated into a separate file