|  |  |
| --- | --- |
| Saïden | 26/09/2023 |
| Tidjine | L2infoT7 |

**Rapport TP5**

* **Partie 1 : Serveur HTTP natif Node.js**

**Question 1.1** donner la liste des en-têtes de la réponse HTTP du serveur.

HTTP/1.1 200 OK

Date: Tue, 26 Sep 2023 00:42:41 GMT

Connection: keep-alive

Keep-Alive: timeout=5

Transfer-Encoding: chunked

HTTP/1.1 200 OK

Date: Tue, 26 Sep 2023 00:42:41 GMT

Connection: keep-alive

Keep-Alive: timeout=5

Transfer-Encoding: chunked

**Question 1.2** donner la liste des en-têtes qui ont changé depuis la version précédente.

HTTP/1.1 200 OK

Content-Type: application/json

Date: Tue, 26 Sep 2023 00:50:17 GMT

Connection: keep-alive

Keep-Alive: timeout=5

Content-Length: 20

HTTP/1.1 200 OK

Content-Type: application/json

Date: Tue, 26 Sep 2023 00:50:17 GMT

Connection: keep-alive

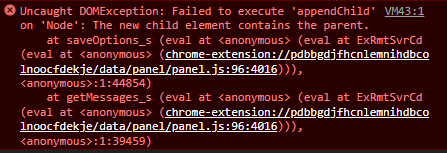
Keep-Alive: timeout=5

Content-Length: 20

**Question 1.3** que contient la réponse reçue par le client ?

|  |
| --- |
| <!DOCTYPE html>  <head>  <title>My Website</title>  <style>  html {  margin: 0;  padding: 0;  border: 0;  width: 100%;  height: 100%;    }    body {  width: 100%;  height: 100%;  position: relative;  background-color: rgb(236, 152, 42);  display: flex;  flex-direction: column;  justify-content: center;  align-items: center;  }    .center {  color: white;  font-family: Helvetica, sans-serif;  text-align: center;  }    h1 {  font-size: 144px;  }    p {  font-size: 64px;  }  </style>  </head>  <body>  <div class="center">  <h1>Hello Again!</h1>  <p>This is served from a file</p>  </div>  </body>  </html> |

**Question 1.4** quelle est l’erreur affichée dans la console ? Retrouver sur <https://nodejs.org/api> le code d’erreur affiché.



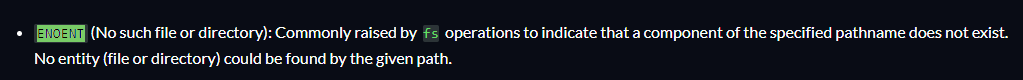
Erreur console-navigateur

⏬



Erreur console-VScode

⏬



Erreur répertorier-sur-node.org

[here](https://nodejs.org/api/errors.html#common-system-errors)

**Question 1.5** donner le code de requestListener() modifié *avec gestion d’erreur* en async/await.

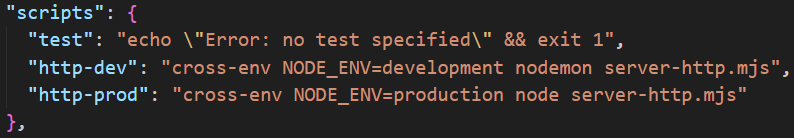
|  |
| --- |
| import http from "node:http";    const host = "localhost";  const port = 8000;    import fs from "node:fs/promises";    function ReadFile(index,format) {  let result = fs.readFile(index,format);  return result;  };    async function requestListener(\_request, response) {  try {  const contents = await ReadFile("index.html", "utf8");  response.setHeader("Content-Type", "text/html");  response.writeHead(200);  return response.end(contents);  } catch (error) {  const contents = console.error();  return response.end(contents);  }  };    const server = http.createServer(requestListener);  server.listen(port, host, () => {  console.log(`Server is running on http://${host}:${port}`);  }); |

**Question 1.6** indiquer ce que cette commande a modifié dans votre projet.

|  |  |
| --- | --- |
| **npm install cross-env --save** | **npm install nodemon --save-dev** |
| * Cette commande installe le package ‘cross-env’ dans notre projet. * ‘--save’ permet la dépendance directe entre le package importer et notre package.json. | * Cette commande installe le package ‘nodemon’ dans notre projet. * ‘**--save-dev**’ permet l’enregistrement du package en tant que dépendance de développement de notre fichier package.json. |

|  |  |
| --- | --- |
| **Résultat ‘npm run http-dev'** | **Résultat ‘**npm run http-prod' |
|  |  |

**Question 1.7** quelles sont les différences entre les scripts http-dev et http-prod ?



|  |  |
| --- | --- |
| **http-dev** | **http-prod** |
| Exécute notre application dans un environnement de développement :   * Redémarrage automatique avec ‘nodemon’ permet un meilleur débogage et suppression du cache. * Permet l’activation de diverses fonctionnalités de développement :  |  | | --- | | * Débogage avancé * Port de dev * Gestion du cache |  * Permet une mise a jours des modifications de l’application en 'direct’. | Exécute notre application dans un environnement de production :   * Permet l’activation de diverses options d’optimisation de la :  |  | | --- | | * Sécurité * Gestion d’erreurs * Mise en cache |  * Redémarrage manuel. * Mise a jours méthodiques des modifications de l’application. |

**Question 1.8** donner les codes HTTP reçus par votre navigateur pour chacune des quatre pages précédentes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Routes** | http://localhost:8000/index.html | http://localhost:8000/random.html | http://localhost:8000/ | http://localhost:8000/dont-exist |
| **Code HTTP** | 200 | 200 | 404 | 404 |

* **Partie 2 : Framework Express**

**Question 2.1** donner les URL des documentations de chacun des modules installés par la commande précédente.

1. Express.js : [Expressjs.com](https://expressjs.com)
2. Http-errors : [npmjs.com](https://www.npmjs.com/package/http-errors)
3. Loglevel : [npmjs.com](https://www.npmjs.com/package/loglevel)
4. Morgan : [npmjs.com](https://www.npmjs.com/package/morgan)

**Question 2.2** vérifier que les trois routes fonctionnent.

|  |  |  |  |
| --- | --- | --- | --- |
| **Routes** | http://localhost:8000 | http://localhost:8000/random | localhost:8000/random/100 |
| **Etats** | ✅ | ➖ | ❌ |
| **Screen** |  |  |  |

**Question 2.3** lister les en-têtes des réponses fournies par Express. Lesquelles sont nouvelles par rapport au serveur HTTP ?