

TDM07 de Technologies Web: Express.js

ASI4 - INSA Rouen

CORRECTION

Forum ITI (4) : NodeJS

L'objectif de l'exercice est de développer le code nécessaire à l'affichage de messages en mode anonyme, dans le forum ITI (en d'autres termes, un livre d'or!).

Vous développerez une page affichant :

- un formulaire identique à celui réalisé au cours du TDM sur Javascript et jQuery (vous pouvez récupérer une correction en annexe),
- la liste des messages précédemment déposés.

Les différents messages seront sauvegardés dans un fichier côté serveur.

Le script devra afficher en dessous du formulaire un tableau qui contiendra tous les messages formatés comme suit : sur la première ligne, la date où le message a été enregistré suivi du nom avec l'email entre parenthèse, et sur la deuxième ligne le message enregistré.

Pensez également à mettre à jour la CSS pour un affichage plus propre de la liste de messages.

Livre d'or



Coordonnées :

Name:

E-mail:

Etudiant : ☒ Oui ☐ Non

Année :

Message :

mon message

Je souhaitez recevoir une copie du message : ☒

Poster le message

Effacer

Liste des messages postés

11/05/2026 17:24:48	Bob
Salut	
11/05/2026 17:47:43	Sam
Bonjour	

Afin de faciliter le développement, vous pourrez suivre les différentes étapes suivantes :

- Créez un serveur node.js permettant de gérer des ressources statiques ; testez-le avec le formulaire HTML.
- Ajoutez un chemin (par exemple '/messages'), qui enregistre les informations envoyées en POST par le formulaire, dans un fichier texte côté serveur. Vérifiez bien que les enregistrements se font correctement.
- Ajoutez un chemin en GET pour afficher la liste des messages.
- Formatez les messages en HTML et décommentez l'iframe du fichier HTML afin d'ajouter la liste.

Note : les iframes sont un peu ... désuètes et vieillottes ! Vous apprendrez à les remplacer en TW2.

Remarques

1. Continuez à vérifier vos pages (<http://validator.w3.org/> ou <http://www.xmlvalidation.com/>).
2. Vous aurez accès à la correction de ce TDM au format PDF, une fois celle-ci finalisée.
3. **Déposez votre compte-rendu sur moodle sous la forme d'un fichier PDF nommé TDM06-login.pdf, chez chacune des 2 personnes du binôme.**

Correction

- 'npm init'
- 'npm install --save node-static'
- optionnel : 'npm install --save-dev nodemon'

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Livre d'or</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="Ressources/style.css" rel="stylesheet" type="text/css" />
  </head>
  <body>
    <div id="page">
      <div id="top">
        
        <h1>Livre d'or</h1>
      </div>
      <p id="probleme"></p>
      <form name="formulaire" action="http://localhost:8080/messages" method="POST">
        <fieldset>
          <legend>Coordonnées :</legend>
          <label for="name">Name: </label><input placeholder="Saisissez votre nom" id="name" name="name" type="text" size="30" class="a-valider" /><br />
          <label for="email">E-mail: </label><input placeholder="Saisissez votre email" id="email" name="email" type="text" size="30" class="a-valider" /><br />
          <label>Etudiant : </label>
          <input type="radio" id="etudiantinsaoui" name="etudiantinsa" value="Oui" /><label class="labelradio" for="etudiantinsaoui"> Oui</label>
          <input type="radio" id="etudiantinsanon" name="etudiantinsa" value="Non" /> <label class="labelradio" for="etudiantinsanon"> Non</label><br />
          <label for="annee">Année : </label>
          <select name="annee" id="annee">
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
            <option value="5">5</option>
          </select>
        </fieldset>
        <fieldset>
          <legend>Message :</legend>
          <textarea rows="4" cols="50" id="message" class="a-valider" name="message"></textarea><br/>
          <label for="copie">Je souhaitez recevoir une copie du message : </label><input type="checkbox" name="copie" id="copie" checked="checked" /><br/>
        </fieldset>
        <input type="submit" value="Poster le message" />
        <input type="reset" value="Effacer" />
      </form>
    </div>
    <iframe name="messages" type="text/plain" src="/messages" width=530 frameborder=0></iframe>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
    <script src="Scripts/script-jQuery.js"></script>
  </body>
</html>
```

server.js

```
const http = require('http');
const fs = require('fs');
const Static = require('node-static');
const querystring = require('querystring');
//const url = require('url');

const HOSTNAME = '127.0.0.1';
const PORT = 8080;
const MESSAGE_FILE = 'messages.txt';
const PUBLIC_FILE_SERVER = new Static.Server('./public');

const SERVER = http.createServer((req, res) => {
  if (req.method === 'GET' && req.url === '/messages') {
    fs.readFile(MESSAGE_FILE, 'utf8', (err, data) => {
      if (err) {
        console.error(err);
        res.writeHead(500, { 'Content-Type': 'text/plain; charset=utf-8' });
        res.end('Erreur lors de la lecture des messages');
        return;
      }
      res.writeHead(200, { 'Content-Type': 'text/html; charset=utf-8' });
      let messages = '<!DOCTYPE html>\n';
      messages += '<html>
        <head>
          <title>Liste de messages</title>
```

```

    <meta equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="Ressources/style.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
      <h2>Liste des messages postés</h2>
      <table>';
    messages += data;
    messages += '</table></body></html>';
    res.end(messages);
  });
  return;
}

if (req.method === 'POST' && req.url === '/messages') {
  let formContent = '';
  req.on('data', chunk => {
    formContent += chunk;
  });

  req.on('end', () => {
    let data;
    try {
      data = querystring.parse(formContent);
    } catch (err) {
      res.writeHead(400, { 'Content-Type': 'text/plain; charset=utf-8' });
      res.end('Contenu formulaire invalide');
      return;
    }

    if ('name' in data && data.name !== "" && 'message' in data && data.message !== "") {
      let date = new Date().toLocaleString();
      //const formattedMessage = `${data.name} : ${data.message}\n`;
      const formattedMessage = `<tr><td class="date">${date}</td><td class="auteur">${
data.name}</td></tr>
                                <tr><td colspan="2"><pre>${data.message}</pre></td></tr>
>\n`;
      fs.appendFile(MESSAGE_FILE, formattedMessage, err => {
        if (err) {
          res.statusCode = 500;
          res.end('Erreur lors de l\'écriture du message');
          return;
        }
      });
    } else {
      res.statusCode = 400;
      res.end('Nom ou message vide');
      return;
    }
    //res.writeHead(200, { 'Content-Type': 'text/plain; charset=utf-8' });
    res.end('Message ajouté');
  });
  return;
}

PUBLIC_FILE_SERVER.serve(req, res);
});

SERVER.listen(PORT, HOSTNAME, () => {
  console.log('Server running at http://${HOSTNAME}:${PORT}/');
});

```