NodeJS : Web Application

- Express Generator
- **Formulaire de contact**
- Récupération des données
- Méthodes de requête HTTP
- Applications pratiques

Express Generator

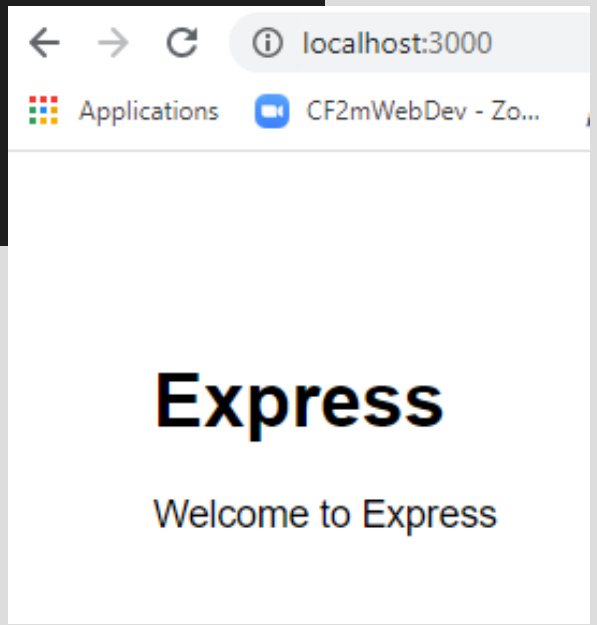
- **Nouveau projet** avec **Express Generator**
- Pas besoin de l’installer de nouveau si global
  - **express formulaireContact --view=ejs**
  - **npm install**
  - **set DEBUG=formulairecontact:* & npm start**
- L'arborescence des fichiers va être créée et le moteur de template choisi est **EJS**
- Les modules sont installés
- Démarrage en mode DEBUG

Installation de l'application

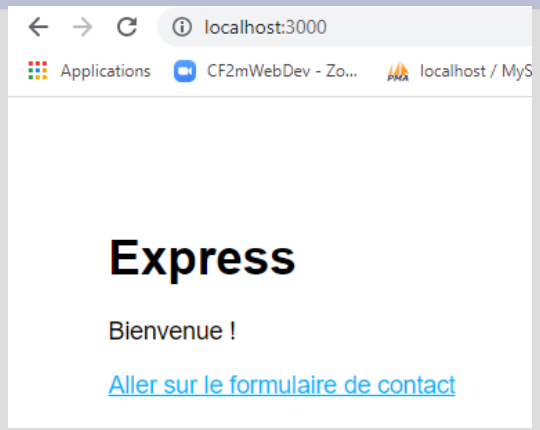
```
PS E:\nodeProjects\demo_bodyParser> express formulaireContact --view=ejs

   create : formulaireContact\
   create : formulaireContact\public\
   create : formulaireContact\public\javascripts\
   create : formulaireContact\public\images\
   create : formulaireContact\public\stylesheets\
   create : formulaireContact\public\stylesheets\style.css
   create : formulaireContact\routes\
   create : formulaireContact\routes\index.js
   create : formulaireContact\routes\users.js
   create : formulaireContact\views\
   create : formulaireContact\views\error.ejs
   create : formulaireContact\views\index.ejs
   create : formulaireContact\app.js
   create : formulaireContact\package.json
   create : formulaireContact\bin\
   create : formulaireContact\bin\www

   change directory:
     > cd formulaireContact

   install dependencies:
     > npm install

   run the app:
     > SET DEBUG=formulairecontact:* & npm start
```

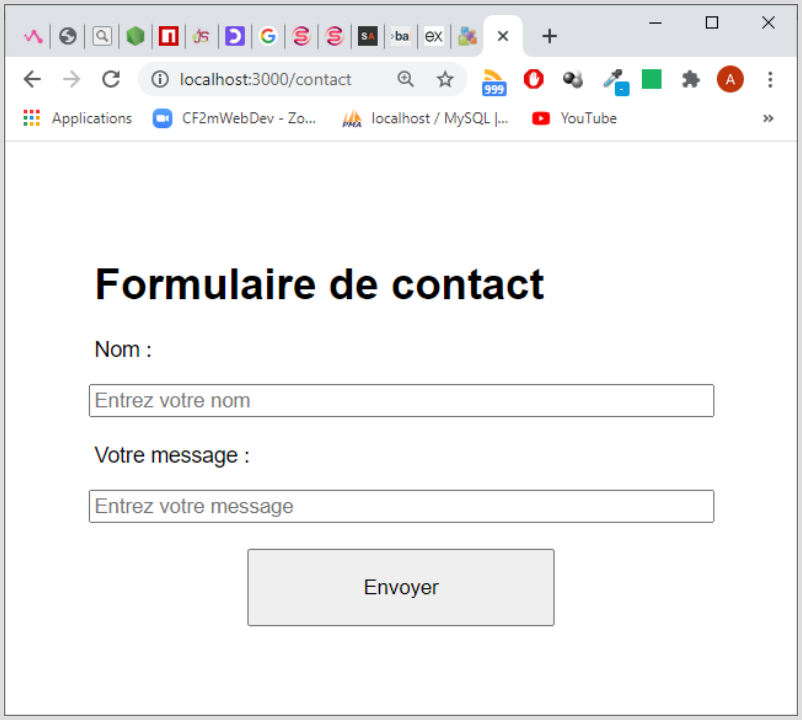
Démarrage de l'application

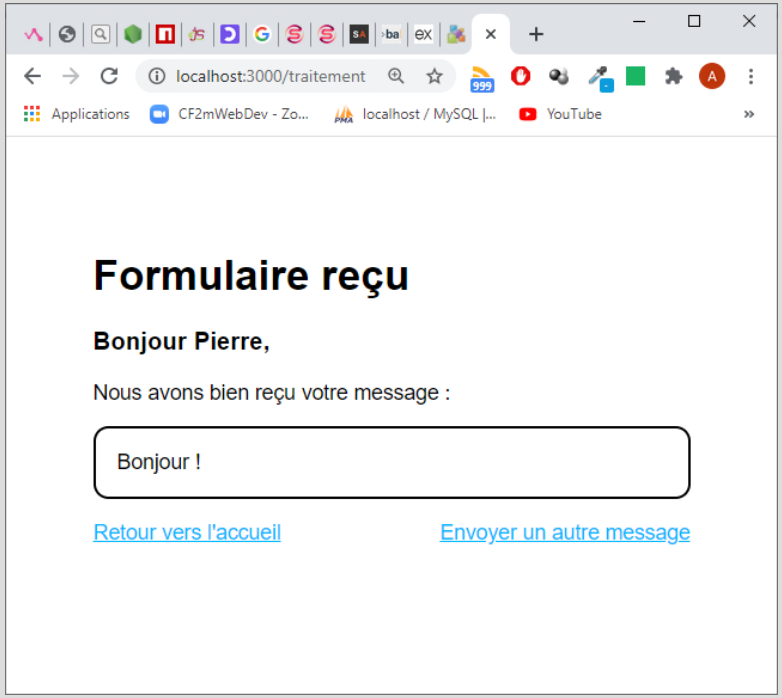
```
PS E:\nodeProjects\demo_bodyParser> cd f*
PS E:\nodeProjects\demo_bodyParser\formulaireContact> npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
added 54 packages from 38 contributors and audited 55 packages in 15.475s
found 0 vulnerabilities

PS E:\nodeProjects\demo_bodyParser\formulaireContact> SET DEBUG=formulairecontact:*
PS E:\nodeProjects\demo_bodyParser\formulaireContact> npm start

> formulairecontact@0.0.0 start E:\nodeProjects\demo_bodyParser\formulaireContact
> node ./bin/www

GET / 304 46.093 ms - -
GET /stylesheets/style.css 200 17.405 ms - 111
GET /favicon.ico 404 7.282 ms - 1193
Terminer le programme de commandes (O/N) ? o
PS E:\nodeProjects\demo_bodyParser\formulaireContact>
```



Affichage des pages







Modification de l’application

- Dans ***package.json***, ajouter dans "scripts"
  - **"devstart" : "nodemon ./bin/www"**
- Et redémarrer l'application avec
  - **npm run devstart**
- Pas de gestion des utilisateurs
  - Commenter les lignes 8 et 23 dans **app.js**
  - (Supprimer le fichier **routes/users.js**)
- Page d’accueil : **lien** vers le formulaire de contact
  - Modifier la vue dans **views/index.ejs**

views/index.ejs

```ejs
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><%= title %></title>
    <link rel='stylesheet' href='/stylesheets/style.css' />
  </head>
  <body>
    <h1><%= title %></h1>
    <p>Bienvenue !</p>
    <p><a href="/contact">Aller sur le formulaire de contact</a></p>
  </body>
</html>
```

Formulaire de contact

- Ajouter une page avec le **formulaire à remplir**
  - Nouvelle vue dans **views/contact_form.ejs**
- Ajouter une page pour **afficher les données**
  - Nouvelle vue dans **views/traiter_form.ejs**
- Adapter le **style** dans la CSS
  - Modifier **public/stylesheets/style.css**
- Modifier le comportement de l'application pour tenir compte de **l'envoi des données**

views/contact_form.ejs

```ejs
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><%= titre %></title>
    <link rel='stylesheet' href='/stylesheets/style.css' />
  </head>
  <body>
    <h1>Formulaire de contact</h1>
    <form name="contact" method="POST" action="/traitement">
        <label for="nom">Nom : </label>
        <input id="nom" name="nom" type="text" placeholder="Entrez votre nom" />
        <br />
        <label for="msg">Votre message : </label>
        <input id="msg" name="msg" type="text" placeholder="Entrez votre message" />
        <br />
        <input type="submit" value="Envoyer" />
    </form>
  </body>
</html>
```

views/traiter_form.ejs

```ejs
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><%= titre %></title>
    <link rel='stylesheet' href='/stylesheets/style.css' />
</head>
<body>
    <h1><%= titre %></h1>
    <h3>Bonjour <%= nom %>,</h3>
    <p>Nous avons bien reçu votre message :</p>
    <p class="reponse"><%= msg %></p>

    <a href="/" style="float: left;">Retour vers l'accueil</a>
    <a href="/contact" style="float: right;">Envoyer un autre message</a>
</body>
</html>
```

stylesheets/style.css

```css
body {
  padding: 50px;
  font: 14px "Lucida Grande", Helvetica, Arial, sans-serif;
}

a {
  color: #00B7FF;
}

/* Formulaire de contact */
form {
  display: flex;
  flex-direction: column;
  align-items: center;
}

label, input {
  width: 100%;
}

label {
  height: 30px;
}

input[type="submit"] {
  width: 50%;
  height: 50px;
}

.reponse {
  border: 2px solid black;
  border-radius: 10px;
  padding: 1em;
}
```

Récupération des données

- Pour gérer l'envoi des données, on doit :
  - créer de **nouvelles routes (URL)**
    - pour demander l'affichage du formulaire à remplir
    - pour recevoir les données introduites dans le formulaire et les traiter
  - utiliser les autres « **verbes HTTP** » avec les méthodes correspondantes de Express
  - exploiter les données envoyées avec le middleware **body-parser**
    (NB : il est intégré dans Express depuis la version 4.16)

body-parser

- Avec le middleware **body-parser**, on analyse le contenu de la requête POST reçue
- Le contenu est lu dans l'objet **Request** de Express via **req.body** après avoir exécuté :
  - app.use(express.json());
    // pour ***application/json***
    (données au format JSON)
  - app.use(express.urlencoded({ extended:false }));
    // pour ***application/x-www-form-urlencoded***
    (données venant d'un formulaire)

Verbe HTTP

- Quand le client (browser) envoie une **requête** à un serveur, celle-ci est composée de :
  - Verbe HTTP (action à réaliser)
  - URI (adresse de la ressource utilisée)
  - Version du protocole HTTP (1.1 ou 2)
  - Headers (en-têtes)
  - Body (contenu de la requête, les données)
- Les actions (méthodes de requête) ou verbes HTTP les plus courants sont : **GET, POST, PUT et DELETE**

Méthodes de requête

- **GET** : Pour demander une représentation de la ressource spécifiée. A utiliser uniquement afin de récupérer des données.
- **POST** : A utiliser pour envoyer des données vers la ressource indiquée.
- **PUT** : A utiliser pour remplacer toutes les représentations actuelles de la ressource visée.
- **DELETE** : A utiliser pour supprimer la ressource indiquée.

Routage dans Express

- A chacune de ces méthodes de requête HTTP correspond une méthode dans Express
  - app.get(…) : recevoir des données
  - app.post(…) : envoyer des données
  - app.put(…) : modifier des données
  - app.delete(…) : supprimer des données
- A utiliser pour les CRUD, les APIs,…
- En combinaison avec des bases de données

routes/index.js

```
var express = require('express');
var router = express.Router();
//var bodyParser = require('body-parser');    // avant Express v4.16

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

/* Afficher le formulaire de contact qui va envoyer les données */
router.get('/contact', function(req, res, next) {
  res.render('contact_form', {titre: "Contact"});
});

/* Utiliser le middleware intégré body-parser pour analyser la requête POST */
//var urlencodedParser = bodyParser.urlencoded({ extended: false });
var urlencodedParser = express.urlencoded({extended:false});

/* Recevoir les données et afficher la page */
router.post('/traitement', urlencodedParser, function(req, res, next){
  let lenom = req.body.nom;
  let lemessage = req.body.msg;

  res.render('traiter_form', {titre: "Formulaire reçu", nom: lenom, msg: lemessage});
});

module.exports = router;
```