Programmation Web – Avancé

JavaScript & Node.js

Gestion de la sécurité du browser

Introduction aux procédures de sécurité appliquées par votre browser

Pourquoi mon application frontend ne peut pas communiquer avec l'application backend ?

Sécurité des communications : SOP

- SOP = Same Origin Policy [93.]
- Appliquée par le browser
- But :
  - Restriction des interactions entre un document ou script chargé par une origine avec une ressource d'une autre origine
  - Isolation des documents ou scripts malicieux, réduction des attaques

Sécurité des communications : SOP

- Même origine entre deux URLs si même :
  - Protocole
  - Port
  - Host

Sécurité des communications : SOP

Type d'interaction	SOP Permission	Ressources
Cross-origin writes Requêtes vers une autre origine	Permis	Liens, redirection, soumission de formulaires
Cross-origin embedding	Permis	JavaScript via <script src="…"></script>, CSS via <link rel="stylesheet" href="…">, Images, Media, iframes…
Cross-origin reads Réponse d'une autre origine	Interdit	

Relaxer la sécurité via des CORS

- CORS = Cross Origin Resource Sharing [94.]

- Spécification par le serveur :
  - des origines pouvant lire ses ressources via un web browser / pouvant accéder à ses réponses
  - via des « HTTP headers »

Problème de sécurité : CORS trop large

NB : attaque XSS

Option A : Relaxer la sécurité via des CORS

- Installation du package cors [95.]: npm i cors
- Headers configurés au niveau du backend via Middleware
- Configuration et utilisation

```
var cors = require('cors');
let corsOptions = {
  origin: 'http://localhost:8080',
}
// enable CORS for all routes in the given router
app.use("/pizzas", cors(corsOptions), pizzaRouter);
```

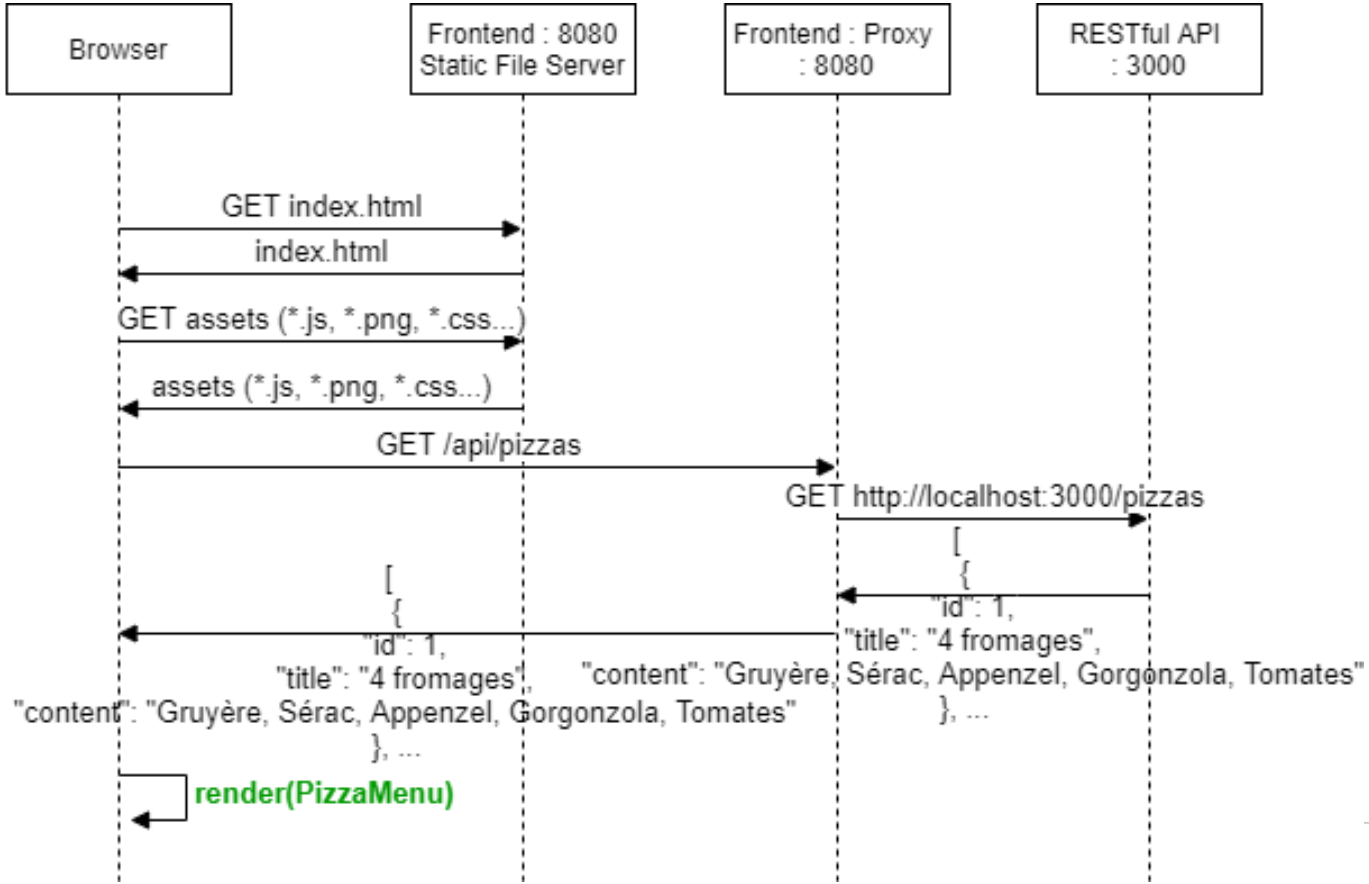
Option A : Relaxer la sécurité via des CORS

- DEMO : Création d'une RESTfull API pour une pizzeria : Part 7 – Gestion des CORS via l'autorisation d'une nouvelle origine
- DEMO : Création d'une SPA : Part 0 : consommation d'une opération non protégée de la RESTful API gérant les CORS
- OK, mais quid si l'API est non modifiable ?

Option B : Contourner le SOP via un proxy au niveau du frontend

- Transfert des requêtes du frontend vers vos Web API

Redirection des requêtes via un proxy



Option B : Contourner le SOP via un proxy au niveau du frontend

- Configuration de **Webpack devServer proxy** :

```
devServer: {
      proxy: {
      "/api": {
        target: "http://localhost:3000",
        pathRewrite: {'^/api' : ''}
      },
    },
```

- **GET /api/pizzas** devient
  GET http://localhost:3000/pizzas

Option B : Contourner le SOP via un proxy au niveau du frontend

- Proxy pour développement
  - Existence d'une multitude de proxys : **Webpack devServer** et son **proxy**, **VS Code proxy, Node** et son **proxy** ...
  - Proxy complet sous Node : **http-proxy-middleware** [96.]

Option B : Contourner le SOP via un proxy au niveau du frontend

- Dev proxy : http-proxy-middleware [96.]

```
const { createProxyMiddleware } = require("http-proxy-middleware");
app.use("/api",
  createProxyMiddleware({target: config.API_URL,
    changeOrigin: true,
    logLevel: "debug",
    /* If we wanted that the call to http://localhost/api were transformed to
       API_URL/ (instead of API_URL/api/)
  /*pathRewrite: {
      '^/api/': '/' // remove base path
    },*/
  })
);
```

Option B : Contourner le SOP via un proxy au niveau du frontend

- Dev proxy
  - Autre option : Configuration du serveur de développement de Node (package.json), e.g. :

```
"proxy": "http://localhost:3000",
```

Option B : Contourner le SOP via un proxy au niveau du frontend

- Proxy pour la production : voir les instructions de votre provider
  - Example : utilisation de **static.json** pour configurer un **static file server** et son **proxy** sous **heroku**

Option B : Contourner le SOP via un proxy

- **DEMO : Création d'une SPA : Part 1 : consommation d'opérations non protégées de la RESTful API grâce à un proxy**
  - Gestion de l'affichage de différentes pages dans une SPA pas trop lourd ?
  - Comment gérer l'affichage de la barre de navigation si l'utilisateur est authentifié ou pas ?
  - Que faire avec le token ?
  - Quid de l'URL ? Quid de l'historique ?