INF3005 – Programmation web avancée

Introduction

Jacques Berger

Objectifs

Présenter l'architecture d'une application web

Prérequis

HTTP

Architecture

L'architecture est l'ensemble des composants de haut niveau d'un système logiciel

L'architecture se distingue par la nature de ses composants et des communications entre eux

Architecture

Une application web peut mettre en oeuvre différents types d'architecture

Client-Serveur
n-tiers
Microservices

Architecture

Dans ce cours, nous allons principalement utiliser l'architecture client-serveur

Client-Serveur

L'architecture client-serveur est la base des échanges sur le web

2 rôles

- Un client
- Un serveur

Client-Serveur

Le serveur offre des fonctionnalités et des données

Le client veut consommer ces fonctionnalités et ces données

Client-Serveur

Un client avec très peu de logique d'affaires est un client léger (thin client)

Un client avec beaucoup de logique d'affaires est un client lourd (fat client)

Client-Serveur

Le client envoie des requêtes au serveur

Le serveur répond aux requêtes

Le serveur gère plusieurs clients simultanément

Le client ne traite qu'avec un seul serveur

Application web

Dans une application web classique, le client est le fureteur

Le client reçoit du serveur un ensemble de données avec les technologies suivantes :

- HTML
- CSS
- Javascript

Application web

Le client interprète ces données et les exécute au besoin

Tout ce qui est exécuté dans le fureteur est considéré comme le client

Application web

Le serveur est un serveur web

Le serveur web écoute les demandes de connexion sur le port 80 (http)

Pour chaque requête, il produit une réponse qui est souvent en format HTML

La réponse est envoyée au fureteur

Application web

Le serveur web exécute du code interprété ou compilé

Ex : PHP, Node.js, Python, Java, etc.

Application web

Le fureteur communique avec le serveur web en utilisant le protocole HTTP

Application web

Frontend : Tout ce qui est exécuté par le fureteur

Backend : Tout ce qui est exécuté par le serveur web

Application web

Tout le code du frontend est ouvert, c'est-à-dire que n'importe qui peut y accéder et le consulter

Si on ne veut pas publier un bout de code, il doit être fait dans le backend

Base de données

Le frontend est limité en ressources

Notamment, le frontend ne peut pas utiliser une base de données

Toute communication avec une base de données doit être faite par le backend

Base de données

Si le frontend veut lire les données d'une base de données, elle doit faire une requête au backend

Ensuite, le backend doit faire la requête à la BD et envoyer le résultat au fureteur dans un format qu'il comprendra

Formulaires

Les formulaires HTML envoient automatiquement les données des champs au backend, lorsque le formulaire possède un *action* et une *method*

Les formulaires HTML ne supporte que 2 méthodes : GET et POST

HTTP supporte 9 méthodes au total : GET, POST, HEAD, PUT, DELETE, PATCH, OPTIONS, TRACE, CONNECT

Formulaires

Lorsque le formulaire envoie les données avec la méthode GET, les données sont encodées comme des paramètres dans l'URL

Formulaires

Lorsque le formulaire envoie les données avec la méthode POST, les données sont encodées dans le body de la requête HTTP

Formulaires

Dans les deux cas, le serveur web peut lire les données dans la requête HTTP, mais pas au même endroit

La différence majeure entre les deux méthodes réside dans la visibilité des valeurs du formulaire pour l'utilisateur

Formulaires

Peu importe la méthode utilisée par le formulaire, les données sont échangées en clair (sans aucune encryption)

Formulaires

Lors d'un POST, le serveur aura tendance à retourner un nouveau document HTML au fureteur

Post-Redirect-Get

Post-Redirect-Get est un design pattern pour l'échange de requêtes entre le frontend et le backend

L'objectif du PRG est d'éviter que la page résultante d'un POST soit actualisée par l'utilisateur, ce qui provoquerait un renvoie du POST et possiblement un doublon dans la BD

Post-Redirect-Get

Avec PRG, le serveur web va demander une redirection au fureteur après un POST au lieu de lui renvoyer un document HTML

Ainsi, le fureteur effectuera la redirection et comme l'URL va changer, même si l'utilisateur actualise la page, ça sera sans effet

Post-Redirect-Get

Déroulement de l'échange :

Le fureteur envoie un POST
Le serveur traite le POST et renvoie une redirection
Le fureteur fait un GET sur la nouvelle URL