TD7 : Projet Doodle

Cet exercice de synthèse consiste à développer une application Web permettant d'inviter des participants à des évènements, et de recueillir leur réponse (confirmation de présence ou non).

repository github

Contraintes

L'application sera composée de deux modules :

Backend

Le backend sera constitué d'une API REST responsable de l'interrogation de la base de données (MariaDB).

Ce backend pourra être interrogé depuis le frontend web, ou depuis une application mobile.

Il sera développé en utilisant un micro-framework PHP (SLIM).

FrontEnd

Le Front sera développé en HTML/CSS, vanilla JS + Bootstrap pour la partie CSS

L'API fetch sera utilisée pour les requêtes HTTP vers l'api backend.

Fonctionnalités de l'application

Création d'un événement :

- Titre de l'événement
- Description
- Dates et heures proposées

Participation à un événement :

- Confirmation ou refus de la participation
- Liste des participants et leurs réponses

Gestion des événements :

- CRUD pour les événements (création, lecture, mise à jour, suppression)

Organisation du travail en équipe

Backend Team :

- Mise en place de l'API REST
- Création des routes et des contrôleurs
- Gestion de la base de données

Injection de PDO

```
return function (ContainerBuilder $containerBuilder) {
    $containerBuilder->addDefinitions([
        LoggerInterface::class => function (ContainerInterface $c) {
            $settings = $c->get(SettingsInterface::class);

            $loggerSettings = $settings->get('logger');
            $logger = new Logger($loggerSettings['name']);

            $processor = new UidProcessor();
            $logger->pushProcessor($processor);

            $handler = new StreamHandler($loggerSettings['path'],
$loggerSettings['level']);
            $logger->pushHandler($handler);

            return $logger;
        },
        'db' => function (\Psr\Container\ContainerInterface $c) {
            $db = [
                'dbname' => 'doodle',
                'user'   => 'root',
                'pass'   => '',
                'host'   => '127.0.0.1'
            ];

            $connection = new PDO("mysql:host=" . $db['host'] .
";port=3306;dbname=" . $db['dbname'], $db['user'], $db['pass']);
            $connection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            $connection->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,
PDO::FETCH_ASSOC);

            return $connection;
        },
    ]);
};
```

Usage db

```
    $app->get('/', function (Request $request, Response $response) {
        $db = $this->get('db');
        $stmt = $db->query('SELECT * FROM utilisateur');
```

```
        $users = $stmt->fetchAll();
        $response->getBody()->write(json_encode($users));
        return $response->withHeader('Content-Type', 'application/json');
    });
```

Frontend Team :

- Création des formulaires et de l'interface utilisateur
- Intégration avec l'API via Fetch API
- Gestion des interactions et des réponses des utilisateurs

```
const API_URL = 'http://localhost:8080/';

//Get request to the server using fetch
apiGet = async(resource) => {
    const url = `${API_URL}${resource}`;
    const response = await fetch(url);
    return await response.json();
};

//Post request to the server using fetch
apiPost = async (resource, data) => {
    const url = `${API_URL}${resource}`;
    const response = await fetch(url, {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(data)
    });
    return await response.json();
};
```