

**Certification 37805 Développeur Web Examen -**

**Bloc3 développement avancé via Framework. Certificateur : Webecom**

**Il vous est possible sur ce bloc de réaliser soit l’application avec un Framework back (ex :Symfony, Laravel...) Soit de réaliser l’application orientée Framework front (ex: Node, React...)**

**Sujet 1 en utilisant un Framework Back :**

**Objectif :**

Wacdo souhaite disposer d'une application pour gérer le et les affectations des collaborateurs dans les différents restaurants

Il s'agit donc de gérer :

• les collaborateurs

• les restaurants

• les fonctions

• les affectations

L'application est utilisée uniquement par des collaborateurs en ayant les autorisations (au niveau de la table des collaborateurs, on gère une information pour savoir si il a le droit d'utiliser l'application, et si oui un mot de passe pour se connecter)

**Technologies et Outils:**

Framework : Utilisation d’un framework back pour le développement de l'application.

**Langage de Programmation :**

-Utilisation d’un langage serveur pour le développement des fonctionnalités de l'application (ex : PHP)

- Utilisation d'un moteur de template (ex : Twig)

**Base de Données :**

- Utilisation d'une base de données SQL (par exemple, MySQL, PostgreSQL).

- Gestion des entités avec un ORM (ex : Doctrine)

**Sécurité :**

Mise en œuvre de mesures de sécurité nécessaires, y compris l'authentification et l'autorisation des utilisateurs.

**Modèle de données :**

Le modèle de données, que vous pouvez enrichir si nécessaire, comporte les objets suivants (nous parlons ici du modèle conceptuel)

Collaborateur

• nom

• prénom

• email

• date de première embauche

• administrateur (true / false)

• mot de passe (pour les administrateurs)

Restaurant

• nom

• adresse

• code postal

• ville

Fonction (les postes existants chez Wacdo)

• intitulé du poste (exemple : équipier polyvalent, manager, etc.…)

Affectation (affectation des équipiers sur les postes et les restaurants)

• collaborateur (objet collaborateur)

• restaurant (objet restaurant où le collaborateur est affecté

• poste (objet poste)

• début (date de début d'affectation)

• fin (date de fin d'affectation ou vide si l'affectation est active)

Fonctionnalités de l'Application :

L'application n'est utilisable que si l’utilisateur est identifié, à partir d'un compte "collaborateur" ayant le droit "administrateur" (et un mot de passe). Le menu principal comporte les options suivantes :

• gestion des restaurants

• gestion des collaborateurs

• gestion des fonctions

• recherche des affectations

**Gestion des restaurants :**

On arrive sur la liste des restaurant, avec un formulaire pour rechercher et filtrer (par nom, par code postal, par ville.

On a un bouton pour créer un restaurant.

Les éléments de la liste sont cliquables, pour avoir le détail du restaurant, incluant la liste des collaborateurs en poste dans ce restaurant (poste en cours). Cette liste est filtrable par poste, par nom, par date de début d'affectation.

Sur le détail, un bouton “modifier”, permet de voir l'historique des affectations (filtrable) et d’’affecter un nouveau collaborateur.

**Gestion des fonctions :**

Permet de voir la liste des différentes fonctions

Un bouton permet de créer une fonction et chaque fonction est éditable.

**Gestion des collaborateurs :**

Dirige sur la vue comportant la liste des collaborateurs, avec un formulaire pour rechercher et filtrer (par nom, prénom, email)

Un bouton permet de créer un collaborateur, et un bouton permet de rechercher les collaborateurs non affectés.

Les éléments de la liste sont cliquables, pour avoir le détail du collaborateur, incluant la ou les affections en cours, et l'historique des affectations. Cette liste est filtrable par poste, par date de début d'affectation.

Sur le détail, un bouton permet de modifier le collaborateur, pour l'affecter à un nouveau poste.

Les affectations en cours sont modifiables.

**Recherche des affectations :**

Permet d’afficher la liste des affectations, avec un formulaire pour rechercher et filtrer par poste, par date de début et de fin, par ville.

**Validation des données :**

-Une étape ne peut pas être finalisées s'il manque des informations essentielles

-Les champs de formulaires doivent être vérifies (téléphone, adresse, nom , prénom)

**Tests et Validation :**

Avant le déploiement de l'application, une série de tests devra être effectuée pour s'assurer que l'application répond aux spécifications mentionnées ci-dessus. Cela inclut des tests d'interface utilisateur, des tests fonctionnels et des tests de sécurité.

**Livrables Attendus :**

Le candidat construit son environnement de développement en installant un framework et les dépendances inhérentes aux fonctionnalités demandées. Le candidat développe l'intégralité de l’application demandée dans l’environnement du framework

Il présente son application fonctionnelle et déployée sur un serveur. Il argumente le fonctionnement global du framework, ses spécificités, son architecture et les choix des dépendances installées.

Le jury demande au candidat des modifications ou ajouts de code pour solutionner une problématique inattendue dans l’instant.

**--------------------------------------------------------------------------------------------------------------**

**Sujet 2 en utilisant un Framework front :**

**Objectif :**

Pour les besoins de la nouvelle application mobile de commande en ligne nous travaillerons pour la chaîne de restaurants Mac-Donald’s

Vous devrez intégrer l'interface de recherche d'un restaurant Mac-Donald sur une carte, en utilisant React dans un environnement Vite.js. Vous développerez la fonctionnalité du choix d’un restaurant à partir du prototype fourni.

**Prototype fourni :**

**https://www.figma.com/design/LQaFXIxxoouIdauOPd4biC/react-sujet?node-id=0-1&t=dJbk7F0xrmGBiEfy-1**

**Description du parcours utilisateur :**

- L’utilisateur arrive sur l’écran avec une carte, un champ de recherche et un overlay lui précisant qu’aucun restaurant n’est encore sélectionné

- Il entre la ville dans laquelle il souhaite rechercher un restaurant

- La liste des villes trouvées par l’API correspondant la recherche s’affiche sous le champ de recherche

- L’utilisateur choisis la bonne ville en cliquant sur une proposition

- La carte se centre sur la ville sélectionnée et la liste des Mac Donald de la ville s’affiche sur la carte sous forme de markers

- Lorsque l’on clique sur un marker, un popup s’ouvre affichant l’adresse exacte du restaurant et un bouton “sélectionner”

- Une fois que l’utilisateur clique sur sélectionner, l’overlay affiche le restaurant sélectionner et un bouton “continuer”

**Spécificités techniques :**

Vous utiliserez React et l’outil de développement Vite.js , et prendrez soin de bien organiser votre projet, ainsi que de documenter vos composants

Pour la carte vous utiliserez la librairie react-leaflet : **https://react-leaflet.js.org/**

Pour les informations géographiques (rechercher une ville, rechercher les restaurant dans une ville) vous interrogerez l’api Nominatim en vous référant à la documentation (évitez d’interroger l’api sur des événements onChange pour limiter le nombre de requêtes) : **https://nominatim.org/release-docs/develop/api/Search/**

Si le temps vous le permet, vous pourrez améliorer la fonctionnalité en proposant à l’utilisateur de directement centrer la carte sur la ville dans laquelle il se trouve et afficher les restaurants proches de lui

**Tests et Validation :**

Chaque composant doit être testé individuellement, ainsi que la totalité de l'application qui doit assurer :

- Une architecture correcte des composants et des assets

- Le découpage schématique des différents composants que vous avez identifié.

**Livrables Attendus :**

Le candidat construit son environnement de développement en installant un framework. Le candidat développe l'intégralité de l’application demandée dans l’environnement du framework

Il présente son application fonctionnelle et déployée sur un serveur. Il argumente le fonctionnement global du framework, ses spécificités, son architecture.

Le jury demande au candidat des modifications ou ajouts de code pour solutionner une problématique inattendue dans l’instant.