**CAHIER DES CHARGES**

**Nom de l’entreprise : MadeInAkisch**

**Nom de projet : Application web de gestion de Cantine universitaire**

**Personne à contacter dans l’entreprise : AKPO K. Akisch**

**Adresse : Zanguera-Lomé-TOGO**

**Email : akpoakisch@emiaza.com**

**SOMMAIRE**

1. Présentation de l’entreprise
2. Présentation du projet
3. Arborescence de l’application
4. **Fonctionnalités de l’application**
5. **Technologies utilisées**
6. **Prestations attendues**

**I. Présentation de l’entreprise**

**Nom de l’entreprise**: Madeinakisch

**Description** :  
Madeinakisch est une entreprise spécialisée dans le développement web et mobile ainsi que dans la digitalisation des services. Son objectif est de concevoir des solutions innovantes permettant d’optimiser la gestion et l’accessibilité des services à travers des plateformes numériques performantes.

Madeinakisch s’engage à accompagner les organisations dans leur transition numérique en proposant des applications adaptées aux besoins spécifiques de chaque domaine d’activité.

**II. Présentation du projet**

**2.1. Contexte**

Dans les universités, la gestion de la cantine est souvent un défi, notamment en raison des longues files d’attente et des difficultés d’organisation des repas. Les étudiants ne disposent pas toujours d’informations en temps réel sur le menu, ce qui peut entraîner des pertes de temps et une mauvaise répartition des plats.

Avec l’essor du numérique, il devient essentiel de proposer une solution digitale permettant d’optimiser la gestion des repas et de faciliter la réservation des plats. C’est dans ce contexte que le projet de **gestion de cantine universitaire** a été conçu.

**2.2. Description du projet**

Le projet consiste à développer une **application web** permettant aux étudiants d’accéder au menu de la cantine, de réserver leurs repas à l’avance, de noter les plats et de recevoir des notifications sur les mises à jour du menu. L’administration pourra, quant à elle, gérer les plats disponibles, organiser un emploi du temps des repas, valider ou refuser les réservations et envoyer des notifications aux étudiants.

**2.3. Objectif général du projet**

Développer une application de gestion de cantine universitaire qui permet d’optimiser l’organisation des repas, d’améliorer l’expérience des étudiants et de faciliter le travail du personnel administratif.

**2.4. Objectifs spécifiques**

* **Faciliter l’accès aux repas** : Permettre aux étudiants de consulter le menu et de réserver leurs plats à l’avance.
* **Optimiser la gestion des repas** : Offrir aux administrateurs un tableau de bord pour mieux organiser les plats et les réservations.
* **Améliorer l’expérience utilisateur** : Proposer une interface fluide avec des notifications en temps réel.
* **Assurer un suivi efficace des réservations** : Permettre aux étudiants de voir le statut de leurs demandes (acceptées, refusées, expirées).
* **Gérer les avis des étudiants** : Permettre aux étudiants de laisser des avis sur les plats et aux administrateurs de les consulter.
* **Personnaliser l’expérience utilisateur** : Ajouter un espace profil étudiant pour modifier les informations personnelles.
* **Optimiser la recherche et la navigation** : Mettre en place un système de filtres et de recherche dans le menu des plats.
* **Configurer les paramètres administratifs** : Offrir à l’admin la possibilité d’ajuster certaines règles comme la durée de validité des réservations.

**III. Arborescence de l’application**

| **Rubrique** | **Sous-rubrique** | **Description** |
| --- | --- | --- |
| **Page d’accueil** | Présentation de l’application | Introduction et aperçu des fonctionnalités |
|  | Plats les plus appréciés | Affichage des plats les mieux notés |
| **Authentification** | Inscription étudiant | Création d’un compte étudiant |
|  | Connexion étudiant | Accès à l’espace étudiant |
|  | Connexion admin | Accès à l’espace administrateur |
| **Espace Étudiant** | Profil étudiant | Consultation et modification des informations personnelles |
|  | Menu de la semaine | Liste des plats disponibles par jour et par créneau (midi/soir) |
|  | Recherche et filtres | Tri des plats par type, jour, etc. |
|  | Réservation de plats | Sélection d’un plat et ajout de suppléments |
|  | Suivi des réservations | Statut des réservations (accepté, refusé, expiré) |
|  | Historique des réservations | Consultation des anciennes réservations |
|  | Notation des plats | Évaluation et commentaires sur les plats |
|  | Notifications | Alertes sur les nouveaux plats, modifications, acceptations ou refus |
|  | Déconnexion | Sortie sécurisée de l’application |
| **Espace Admin** | Gestion des plats | Ajout, modification, suppression de plats |
|  | Planification des repas | Attribution des plats aux jours et créneaux |
|  | Gestion des réservations | Validation ou refus des réservations |
|  | Gestion des avis étudiants | Consultation et modération des avis |
|  | Gestion des étudiants | Liste des étudiants inscrits |
|  | Envoi de notifications | Informations aux étudiants sur les mises à jour |
|  | Tableau de bord | Statistiques sur les réservations et utilisateurs |
|  | Paramètres | Modification des règles de gestion (ex. : durée des réservations) |

### ****IV. Fonctionnalités de l’application****

| **Module** | **Fonctionnalité** | **Description** |
| --- | --- | --- |
|  | Inscription étudiant | Création d’un compte avec nom, prénom, institut, email et mot de passe |
| **Authentification** | Connexion étudiant | Accès à l’espace étudiant via email ou nom d’utilisateur + mot de passe |
|  | Connexion admin | Accès à l’espace administrateur avec identifiants sécurisés |
|  | Déconnexion | Fermeture sécurisée de la session |
|  | Consultation du menu | Visualisation des plats disponibles avec photo, prix et type |
|  | Recherche et filtres | Recherche des plats par type (VIP, standard) et par jour |
|  | Réservation de plats | Sélection d’un plat et ajout de suppléments (boissons, accompagnements) |
| **Espace Étudiant** | Suivi des réservations | Consultation du statut des réservations (acceptée, refusée, expirée) |
|  | Historique des réservations | Accès aux réservations passées |
|  | Notation des plats | Attribution d’une note et d’un commentaire aux plats |
|  | Consultation du profil | Visualisation et modification des informations personnelles |
|  | Notifications | Réception d’alertes sur nouveaux plats, modifications, statut des réservations |
| **Espace Admin** | Ajout de plats | Création d’un plat avec photo, nom, prix, type et description |
|  | Modification de plats | Mise à jour des informations d’un plat existant |
|  | Suppression de plats | Retrait d’un plat du menu |
|  | Planification des repas | Attribution des plats aux jours de la semaine et aux créneaux (midi/soir) |
|  | Gestion des réservations | Validation ou refus des demandes de réservation |
| **Espace Admin** | Gestion des avis étudiants | Consultation et suppression/modération des avis sur les plats |
|  | Gestion des étudiants | Liste et gestion des étudiants inscrits |
|  | Envoi de notifications | Communication des mises à jour aux étudiants |
|  | Tableau de bord | Affichage des statistiques (nombre de réservations, étudiants inscrits) |
|  | Paramètres généraux | Gestion des règles de l’application (durée avant expiration d’une réservation, types de plats, etc.) |

### ****V. Technologies utilisées****

| **Domaine** | **Technologie** | **Description** |
| --- | --- | --- |
| **Frontend** | **React JS** | Bibliothèque JavaScript utilisée pour créer une interface utilisateur interactive et dynamique. Permet de concevoir des composants modulaires pour chaque page (menu, réservation, profil, etc.). |
|  | **HTML / CSS / JavaScript** | Technologies de base pour structurer, styliser et animer l'interface. |
|  | **Bootstrap** | Framework CSS pour un design responsive rapide. |
| **Backend** | **Django** | Framework Python robuste pour le développement d'applications web. Il gère l’authentification, les opérations de base de données, la sécurité et l’API. |
|  | **Django REST Framework (DRF)** | Extension de Django utilisée pour créer une API RESTful permettant la communication entre le frontend (React) et le backend. |
| **Base de données** | **PostgreSQL** | Système de gestion de base de données relationnelle puissant, utilisé pour stocker les données des étudiants, plats, réservations, etc. |
| **API** | **REST API (via DRF)** | Interface de communication entre le frontend et le backend, utilisant les protocoles HTTP (GET, POST, PUT, DELETE). |
| **Hébergement** | **Render.com / Railway.app** | Plateformes modernes recommandées pour héberger à la fois le backend Django + PostgreSQL et le frontend React. |
| **Outils divers** | **Git & GitHub** | Gestion du code source et du travail collaboratif. |
|  | **Postman** | Outil de test des requêtes API. |
|  | **VS Code** | Environnement de développement recommandé pour React et Django. |

### ****VII. Prestations attendues****

| **Catégorie** | **Prestation** | **Description** |
| --- | --- | --- |
| **Conception** | Élaboration du design de l’interface utilisateur | Réalisation d’une interface intuitive, responsive et moderne à l’aide de React JS et Bootstrap. |
| **Contenu** | Rédaction des textes statiques du site | Textes descriptifs des pages (accueil, notifications, messages, etc.) fournis et rédigés par le développeur. |
| **Développement** | Création de l’application web complète | Développement du backend avec Django & DRF, du frontend avec React JS, et d’une base de données PostgreSQL. |
|  | Mise en place des fonctionnalités principales | Modules : authentification, gestion des plats, réservations, notifications, notation, administration, etc. |
|  | Mise en œuvre d’une API REST sécurisée | L’API assurera la communication entre React (frontend) et Django (backend). |
| **Base de données** | Modélisation et mise en place de la base PostgreSQL | Conception du schéma de base de données et intégration via l’ORM Django. |
| **Sécurité** | Authentification et protection des accès | Gestion des sessions, mots de passe hashés, protection CSRF, etc. |
| **Test & Déploiement** | Tests des fonctionnalités et correction de bugs | Tests manuels via Postman et en interface pour s’assurer de la stabilité du système. |
|  | Déploiement complet de l’application | Hébergement du frontend, backend et base de données sur une plateforme comme Render ou Railway. |
| **Documentation** | Guide d'utilisation | Un manuel utilisateur expliquant le fonctionnement des différentes interfaces et fonctionnalités. |
|  | Documentation technique | Détails techniques sur l’architecture, les technologies et les points clés. |

**NOTE:** Dans mon travail, le terme 'étudiants' est souvent utilisé de manière générale pour désigner tous les membres du personnel souhaitant utiliser l'application, qu'il s'agisse des professeurs, des vigiles, des membres de l'administration ou d'autres acteurs.