REPUBLIQUE TUNISIENNE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ TUNIS EL MANAR

FACULTE DES SCIENCES DE TUNIS
DEPARTEMENT DES SCIENCES DE L'INFORMATIQUE

RAPPORT DE PROJET
DE TECHNOLOGIES DU WEB ET MULTIMÉDIA

TITRE
SIDI BOU (Restaurant)

Réalisé par :

Kalai Amenallah

Mouaddeb GHAZI

Encadré par : Lassoued MOHAMMED

Organisme d'accueil: FST – DSI

Année Universitaire 2025-2026

Rapport de Projet Angular 20

RestoOnline - Plateforme de Gestion de Restaurant en Ligne

Projet de Fin de Module Angular
FST - Département Informatique

Table des Matières

---

1. Introduction

1.1 Contexte et Objectifs

RestoOnline est une application web de gestion de restaurant développée avec Angular 20. Elle permet de gérer efficacement les produits, les fournisseurs, les utilisateurs et d'afficher des statistiques en temps réel. L'application propose une interface moderne et intuitive, avec deux niveaux d'accès : administrateur et utilisateur standard.

Les objectifs principaux du projet sont de fournir une solution complète pour la gestion quotidienne d'un restaurant, incluant la gestion des stocks, des fournisseurs, des utilisateurs, et l'affichage de statistiques pertinentes pour la prise de décision.

1.2 Technologies Utilisées

- **Framework** : Angular 20.3.6

- **Langage** : TypeScript 5.7.2
- **UI Library** : PrimeNG avec PrimeIcons
- **Styling** : Tailwind CSS
- **Build Tool** : Angular CLI

---

2. Installation du Projet

2.1 Prérequis

- Node.js version 18.x ou supérieure
- npm version 9.x ou supérieure
- Angular CLI version 20.x

2.2 Installation

Cloner le dépôt

```
git clone
https://github.com/amenallah53/resto-online.git
```

```
cd resto-online
```

Installer les dépendances

```
npm install
```

Lancer le serveur de développement

```
ng serve
```

L'application sera accessible à l'adresse `http://localhost:4200/`

---

3. Architecture et Arborescence

3.1 Structure du Projet

```
resto-online/
├── src/
│   ├── app/
│   │   ├── core/                # Services et guards essentiels
│   │   │   ├── guards/
│   │   │   │   └── auth.guard.ts
│   │   │   └── services/
│   │   │       └── auth.service.ts
│   │   │
│   │   ├── shared/              # Ressources partagées
│   │   │   ├── models/          # Interfaces TypeScript
│   │   │   │   ├── user.ts
│   │   │   │   ├── credentials.ts
│   │   │   │   ├── product.ts
│   │   │   │   └── supplier.ts
│   │   │   └── data/            # Données mock
│   │   │       ├── users.ts
│   │   │       ├── products.ts
│   │   │       └── suppliers.ts
│   │   │
│   │   ├── features/            # Modules fonctionnels
│   │   │   ├── auth/
```

```
│   │   │   ├── login/
│   │   │   ├── sign-up/
│   │   │   └── login-sign-up-page/
│   │   │
│   │   ├── admin/
│   │   │   ├── dashboard/
│   │   │   ├── products/
│   │   │   │   ├── product-list/
│   │   │   │   ├── product-form/
│   │   │   │   └── product-details/
│   │   │   ├── suppliers/
│   │   │   │   ├── supplier-list/
│   │   │   │   ├── supplier-form/
│   │   │   │   └── supplier-details/
│   │   │   └── layout/
│   │   │
│   │   └── user/
│   │       ├── profile/
│   │       └── dashboard/
│   │
│   ├── app.component.ts
│   ├── app.routes.ts
│   └── app.config.ts
│
├── assets/
```

```
│   └── styles.css
│
├── angular.json
├── package.json
├── tsconfig.json
└── tailwind.config.js
```

3.2 Organisation Modulaire

Le projet est divisé en trois grandes parties :

- **Core** : Contient les services essentiels (authentification) et les guards pour protéger les routes
- **Shared** : Regroupe les modèles de données et les données mock utilisées dans toute l'application
- **Features** : Modules fonctionnels indépendants pour l'authentification, l'administration et l'interface utilisateur

---

4. Description Détaillée des Fonctionnalités

4.1 Module d'Authentification

4.1.1 Page Login-Sign-Up-Page

C'est le point d'entrée principal de l'application. Cette page intelligente affiche trois vues différentes selon l'état de connexion de l'utilisateur.

Vue 1 - Utilisateur déjà connecté : Lorsqu'un utilisateur authentifié accède à cette page, il voit :

- Un message de bienvenue "You're Already Logged In"
- Ses informations : email et rôle (Admin ou User)

- Un bouton "Go to Dashboard" qui le redirige vers le tableau de bord approprié selon son rôle (panel admin pour les administrateurs, dashboard utilisateur pour les users)
- Un bouton "Logout" pour se déconnecter

Vue 2 - Formulaire de Login : Pour les utilisateurs non connectés souhaitant se connecter :

- Champs email et password avec validation complète
- Validation du format email et longueur minimale du mot de passe
- Messages d'erreur clairs affichés sous chaque champ en cas de problème
- Système de sécurité : après 3 tentatives échouées, le formulaire est masqué et un message "Compte bloqué temporairement" s'affiche pendant 30 secondes
- Un lien "Don't have an account? Sign up" pour basculer vers l'inscription
- Notifications toast pour informer du succès ou de l'échec de la connexion

Vue 3 - Formulaire de Sign-Up : Pour les nouveaux utilisateurs :

- Formulaire complet avec 5 champs : username, email, firstName, lastName, password
- Validation stricte sur tous les champs (format email, longueur minimale username et password)
- Vérification en temps réel de l'unicité de l'email et du username
- Si l'email ou le username existe déjà, un message d'erreur s'affiche immédiatement
- Lors de la soumission réussie, l'utilisateur est créé et ajouté au système avec un ID unique auto-généré
- Redirection automatique vers la page de login après 1.5 secondes avec un message de succès
- Un lien "Already have an account? Login" pour revenir au login

Communication entre composants : Le composant parent utilise un signal pour gérer l'affichage, et les composants enfants émettent des événements via Output pour demander le changement de vue. Cette architecture permet une gestion fluide de la navigation sans rechargement de page.

4.1.2 Gestion des Rôles et Permissions

Le service d'authentification gère deux types d'utilisateurs :

- **Admin** : Accès complet au panel d'administration (gestion produits, fournisseurs, statistiques)
- **User** : Accès limité au dashboard utilisateur et à son profil

Le guard vérifie à chaque navigation si l'utilisateur est connecté et s'il a les permissions nécessaires pour accéder à la route demandée. Si ce n'est pas le cas, il est automatiquement redirigé vers la page de login.

4.2 Module Administration - Dashboard

Le dashboard administrateur affiche une vue d'ensemble des statistiques du restaurant :

Cartes statistiques principales :

- **Total Produits** : Nombre total de produits dans le stock
- **Produits en Rupture** : Compte automatiquement les produits dont le stock est à zéro, avec un badge rouge pour attirer l'attention
- **Produits Expirés** : Filtre automatiquement les produits dont la date d'expiration est dépassée, permettant une gestion proactive
- **Total Fournisseurs** : Nombre de fournisseurs actifs dans le système
- **Commandes en Attente** : Nombre de commandes nécessitant un traitement
- **Revenus du Mois** : Calcul basé sur les ventes enregistrées

Chaque carte affiche une icône PrimeIcons pertinente, une couleur distinctive et une valeur numérique. Les statistiques sont calculées dynamiquement à partir des données en utilisant des computed signals, garantissant une mise à jour automatique.

Navigation rapide : Depuis le dashboard, l'administrateur peut accéder rapidement à toutes les sections via le menu de navigation : Produits, Fournisseurs, Profil.

4.3 Module Produits - CRUD Complet

4.3.1 Liste des Produits (Product-List)

La page de liste des produits offre une vue complète et interactive :

Affichage des données :

- Tableau responsive utilisant PrimeNG Table
- Affichage des colonnes : Image, Nom, Catégorie, Prix, Stock, Date d'expiration, Actions
- Images des produits affichées sous forme de miniatures
- Badge de couleur pour le statut du stock (vert si disponible, rouge si en rupture)
- Affichage de la date d'expiration avec alerte visuelle si le produit est expiré

Fonctionnalités de recherche et filtrage :

- Barre de recherche en temps réel qui filtre sur le nom et la catégorie
- Filtre par catégorie via dropdown (Boissons, Entrées, Plats, Desserts, etc.)
- Filtre par statut de stock (Tous, En stock, Rupture de stock)
- Tri possible sur chaque colonne (nom, prix, date)

Pagination :

- Affichage de 10 produits par page
- Navigation avec boutons précédent/suivant
- Indicateur du nombre total de produits

Actions disponibles :

- Bouton "Voir" (icône œil) : Redirige vers la page de détails du produit
- Bouton "Modifier" (icône crayon) : Ouvre le formulaire d'édition
- Bouton "Supprimer" (icône poubelle) : Supprime le produit après confirmation
- Bouton "Nouveau Produit" en haut de page pour créer un nouveau produit

4.3.2 Formulaire Produit (Product-Form)

Le formulaire de produit gère à la fois la création et la modification :

Mode Création : Accessible via le bouton "Nouveau Produit", affiche un formulaire vierge avec titre "Créer un nouveau produit".

Mode Édition : Accessible via le bouton "Modifier" d'un produit, pré-remplit le formulaire avec les données existantes et affiche le titre "Modifier le produit".

Champs du formulaire :

- **Nom du produit** : Texte, requis, minimum 3 caractères
- **Description** : Textarea, requise, pour décrire le produit
- **Prix** : Nombre, requis, minimum 0, affiché avec 2 décimales
- **Catégorie** : Dropdown avec options prédéfinies (Boissons, Entrées, Plats, Desserts, Autres)
- **Stock** : Nombre, requis, minimum 0, représente la quantité disponible
- **Date d'expiration** : Calendar PrimeNG, optionnelle, pour les produits périssables
- **URL Image** : Texte, optionnelle, lien vers l'image du produit
- **Fournisseur** : Dropdown listant tous les fournisseurs disponibles

Validation et messages d'erreur : Chaque champ affiche un message d'erreur spécifique sous le champ si la validation échoue (champ requis, valeur minimum, format invalide). Les messages apparaissent en rouge avec une icône d'alerte.

Actions du formulaire :

- Bouton "Enregistrer" : Sauvegarde le produit et redirige vers la liste
- Bouton "Annuler" : Retour à la liste sans sauvegarder
- Toast de confirmation après sauvegarde réussie

4.3.3 Détails du Produit (Product-Details)

Page dédiée accessible via l'URL `/admin/products/:id` :

Affichage des informations :

- Grande image du produit en haut
- Nom du produit en titre principal
- Description complète
- Informations organisées en sections :
    - Prix (en dinars, formaté)
    - Catégorie avec badge coloré
    - Stock disponible avec badge de statut

    - Date d'expiration (si applicable) avec alerte si expiré
    - Fournisseur avec lien vers la page du fournisseur

Gestion des erreurs : Si l'ID du produit n'existe pas dans les données, la page affiche un message "Produit introuvable" avec un bouton de retour à la liste.

Actions disponibles :

- Bouton "Modifier" : Redirige vers le formulaire d'édition
- Bouton "Supprimer" : Supprime après confirmation
- Bouton "Retour" : Retour à la liste des produits

4.4 Module Fournisseurs

4.4.1 Liste des Fournisseurs (Supplier-List)

Organisation similaire à la liste des produits :

Affichage :

- Cartes ou tableau avec : Nom de l'entreprise, Email, Téléphone, Adresse, Nombre de produits fournis
- Badge indiquant le nombre de produits associés
- Icônes PrimeIcons pour email et téléphone (cliquables pour contact direct)

Fonctionnalités :

- Recherche par nom d'entreprise ou email
- Pagination (10 fournisseurs par page)
- Actions : Voir détails, Modifier, Supprimer
- Bouton "Nouveau Fournisseur"

4.4.2 Formulaire Fournisseur (Supplier-Form)

Champs requis :

- Nom de l'entreprise (minimum 3 caractères)
- Email de contact (format email validé)
- Téléphone (format numérique)
- Adresse complète

Champs optionnels :

- Liste des types de produits fournis (multi-select)
- Notes additionnelles

Validation complète avec messages d'erreur contextuels. Sauvegarde et redirection vers la liste après succès.

4.4.3 Détails Fournisseur (Supplier-Details)

Informations affichées :

- Carte d'identité du fournisseur avec toutes ses coordonnées
- Section "Produits fournis" : Liste des produits associés à ce fournisseur avec mini-cartes cliquables
- Statistiques : Nombre total de produits, valeur totale du stock fourni
- Boutons d'action : Modifier, Supprimer, Contacter, Retour

4.5 Module Utilisateur - Profil

La page de profil permet à chaque utilisateur de consulter et modifier ses informations :

Affichage du profil :

- Photo de profil (si disponible) ou avatar par défaut
- Nom complet de l'utilisateur
- Adresse email
- Nom d'utilisateur (username)
- Date de création du compte
- Rôle (Admin ou User)

Modification des informations :

- Formulaire pré-rempli avec les données actuelles
- Modification possible de : firstName, lastName, photo de profil
- Validation des changements avant sauvegarde
- Message de confirmation après mise à jour

Changement de mot de passe : Section séparée avec :

- Champ mot de passe actuel
- Nouveau mot de passe (minimum 6 caractères)
- Confirmation du nouveau mot de passe
- Validation que les deux nouveaux mots de passe correspondent

4.6 Layout et Navigation

4.6.1 Admin Layout

Le layout administrateur entoure toutes les pages du panel admin avec :

Barre de navigation supérieure :

- Logo RestoOnline cliquable (retour au dashboard)
- Menu horizontal avec liens : Dashboard, Produits, Fournisseurs
- Section utilisateur à droite : Photo de profil, nom, menu déroulant avec "Mon Profil" et "Déconnexion"

Sidebar latérale (optionnelle) :

- Navigation verticale avec icônes
- Indicateurs visuels de la page active
- Collapse/expand pour plus d'espace

Zone de contenu principale :

- Affiche le composant de la route active via router-outlet
- Padding et styling cohérents

4.6.2 Interactions et Communication

Entre composants parent-enfant :

- Utilisation de `@Input()` pour passer les données aux composants enfants (exemple : product-card reçoit l'objet product)
- Utilisation de `@Output()` avec EventEmitter pour remonter les événements (exemple : login-page émet switchToSignUp vers login-sign-up-page)

Entre composants via services :

- AuthService partagé pour l'état de connexion accessible dans tous les composants
- ProductService et SupplierService pour la gestion des données CRUD

Navigation programmatique :

- Utilisation du Router pour naviguer après actions (création, modification, suppression)
- Redirection automatique selon le rôle après login

---

5. Choix Techniques

5.1 Angular 20 et Fonctionnalités Modernes

Le projet utilise les dernières fonctionnalités d'Angular 20, notamment :

Composants Standalone : Tous les composants sont standalone, éliminant le besoin de NgModules et simplifiant l'architecture.

Directives de Contrôle de Flux :

- `@if` pour l'affichage conditionnel (remplace *ngIf)
- `@for` pour les boucles (remplace *ngFor)
- `@switch` pour les conditions multiples (remplace *ngSwitch)

Signals : Utilisés pour la gestion d'état réactive moderne, notamment dans le AuthService pour `isLoggedIn` et `currentUser`.

Fonction inject() : Injection de dépendances moderne utilisée dans les composants et guards au lieu de l'injection par constructor.

5.2 Routing et Guards

Le routing utilise :

- Routes paramétrées pour les détails (`:id`)
- Guards fonctionnels avec `CanActivateFn` pour protéger les routes admin

- Lazy loading potentiel pour optimiser les performances

5.3 Formulaires

Template-Driven Forms : Utilisés pour le login et sign-up avec FormsModule, NgForm et ngModel pour une approche simple et rapide.

Reactive Forms : Utilisés pour les formulaires complexes (produits, fournisseurs) avec ReactiveFormsModule et FormBuilder pour un contrôle programmatique avancé.

5.4 Interface Utilisateur

PrimeNG : Bibliothèque UI riche fournissant des composants prêts à l'emploi : Table, Button, InputText, Dropdown, Calendar, Toast, Dialog. Garantit une cohérence visuelle et des fonctionnalités avancées.

PrimeIcons : Jeu d'icônes complet intégré pour tous les boutons et indicateurs visuels.

Tailwind CSS : Framework CSS utility-first pour un styling rapide et cohérent. Configuration personnalisée dans `tailwind.config.js` pour les couleurs et thèmes du projet.

5.5 TypeScript et Typage

Utilisation stricte de TypeScript avec des interfaces bien définies pour tous les modèles de données : User, Product, Supplier, Credentials. Garantit la sécurité du typage et facilite la maintenance.

---

6. Conclusion

6.1 Objectifs Atteints

Le projet RestoOnline répond à tous les critères du cahier des charges :

- Authentification complète avec gestion des rôles et système de sécurité (3 tentatives)

CRUD complet pour produits et fournisseurs avec liste, création, modification, suppression et détails
- Dashboard avec statistiques dynamiques calculées en temps réel
- Routing avancé avec paramètres et guards de protection
- Formulaires Template-Driven et Reactive selon les besoins
- Interface moderne et responsive avec PrimeNG et Tailwind
Communication entre composants via Input/Output et services
- Utilisation des directives Angular 20 (@if, @for, @switch)
- Architecture modulaire propre et maintenable

6.2 Points Forts

- Interface utilisateur moderne et intuitive
- Navigation fluide entre les différentes sections
- Gestion complète des produits et fournisseurs
- Système d'authentification robuste avec sécurité
- Code organisé et structuré selon les bonnes pratiques Angular
- Utilisation des technologies les plus récentes du framework

6.3 Limites Actuelles

- Utilisation de données mock (pas de backend réel)
- Pas de persistance des données après rechargement
- Tests unitaires non implémentés

Le projet constitue une base solide pour une application de gestion de restaurant et démontre une maîtrise complète d'Angular 20 et de ses fonctionnalités modernes.

Projet réalisé par : Amenallah & Ghazi Meddeb
Date : Décembre 2024
Framework : Angular 20.3.6
GitHub : https://github.com/amenallah53/resto-online