

FICHE ÉTUDIANTE - PRÉPARATION À L'ÉVALUATION DU PROJET "TAJ MAHAL"

1. Objectif du projet

- ✓ Développer une application iOS affichant le menu du restaurant Taj Mahal.
- ✓ Respecter la maquette fournie et assurer une navigation fluide. ✓ Structurer le projet en utilisant l'architecture MVVM. ✓ Maintenir les bonnes pratiques de développement.

2. Architecture MVVM (Model-View-ViewModel)

- **Model** : Définit les structures de données (`Dish.swift`).
- **ViewModel** : Gère la logique métier et stocke les plats (`ViewModel.swift`).
- **View** : Affiche l'interface utilisateur et récupère les données via le `ViewModel` (`MenuView.swift`).

➡ Pourquoi MVVM ?

- Sépare clairement la logique des données et l'affichage.
- Facilite la maintenance et l'évolution du projet.
- Permet une réutilisation efficace du code.

3. Navigation et affichage des plats

- ✓ Utilisation de `NavigationStack` et `NavigationLink` pour naviguer entre les écrans.
- ✓ Affichage des plats sous forme de liste avec `List` ou `ScrollView` pour plus de flexibilité.
- ✓ Utilisation de `ForEach` pour générer dynamiquement la liste des plats.

➡ Point technique important :

`NavigationStack` est recommandé pour gérer la navigation efficacement dans iOS 16+.

4. Gestion des données et liaison avec la vue

- ✓ Stockage des plats dans un tableau dans `ViewModel.swift`.
- ✓ Mise à jour automatique de l'affichage grâce à `@Published` et `@ObservedObject`.
- ✓ Gestion du niveau d'épices avec une boucle `ForEach` pour afficher des icônes de piments.

➡ **Point clé** : `@Published` permet d'actualiser la vue dès que les données changent.

5. Gestion du projet avec Git

✓ Utilisation de GitHub pour versionner le projet.

✓ Commandes essentielles :

- `git add .` (ajouter les fichiers)
- `git commit -m "Message"` (sauvegarder un changement)
- `git push origin main` (envoyer sur GitHub)
- ✓ Création de branches pour tester de nouvelles fonctionnalités sans affecter le code principal.

➡ **Point à retenir** : Ne jamais coder directement sur `main`, utiliser des branches.

6. Respect des bonnes pratiques SwiftUI

✓ Respect des standards de design d'Apple (couleurs, typographie, marges).

✓ Séparation des vues pour favoriser la réutilisation (`MenuRowView.swift`, `DishView.swift`).

✓ Utilisation des `@State`, `@Binding`, `@ObservedObject` pour la gestion des données.

➡ **Astuce mémoire** :

- `@State` = Données locales à la vue.
- `@Binding` = Connexion entre deux vues.
- `@ObservedObject` = Vue qui observe un `ViewModel`.

7. Exemples de questions d'évaluation

❓ **Comment les données passent-elles du modèle à la vue ?**

✓ **Réponse** : Elles sont stockées dans `ViewModel.swift` et mises à jour via `@Published`. La View les observe grâce à `@ObservedObject` et les affiche.


❓ **Pourquoi avoir utilisé `List` ou `ScrollView` ?**

✓ **Réponse** : `List` est plus optimisé pour afficher de longues listes, mais `ScrollView` permet plus de personnalisation visuelle.

❓ **Comment avez-vous géré la navigation ?**


✓ **Réponse** : Avec `NavigationStack` et `NavigationLink`, permettant de passer d'un écran à l'autre en conservant un historique de navigation.


Pourquoi utiliser Git ?


 **Réponse :** Pour sauvegarder chaque modification du projet, travailler en équipe et revenir à une version antérieure en cas d'erreur.

8. Sujets recommandés à approfondir (YouTubeurs FR) :

 **Maxime Britto (Purple Giraffe)** – Développement iOS en Swift, explications MVVM.


 **SwiftUI France** – Tutoriels pratiques sur la gestion des vues et la navigation.


 **AppCademie** – Explications détaillées sur la structuration d'une application SwiftUI.


 **Objectif :** Regarder ces vidéos et revoir le code pour mieux répondre aux questions de l'évaluateur.


Récapitulatif express pour mémorisation rapide

 **MVVM** = Modèle (données), Vue (affichage), ViewModel (gestion des données).

 **Navigation** = `NavigationStack` + `NavigationLink`.

 **Affichage dynamique** = `List`, `ScrollView`, `ForEach`.

 **Mise à jour des données** = `@Published` + `@ObservedObject`.

 **Git** = `add` (ajouter) → `commit` (sauver) → `push` (envoyer).

 **YouTubeurs à suivre** = Maxime Britto, SwiftUI France, AppCademie.

 Avec cette fiche, tu peux répondre efficacement à l'évaluation et prouver ta maîtrise du projet. Bonne révision ! 