Sujet TD : ISEN Smart Companion
Votre Assistant Personnel Étudiant

Contexte : Vous avez été contacté par l'administration de l'ISEN Toulon pour développer une application d'assistant personnel pour les étudiants de l'école. L'objectif de cette application est de faciliter la gestion de leur vie étudiante en les aidant à trouver un équilibre entre leurs études et leur vie associative (BDE, BDS, etc.). Grâce à l'intégration d'une intelligence artificielle, l'application permettra d'offrir des conseils personnalisés pour organiser leurs cours, gérer leur emploi du temps, participer à des événements ou optimiser leur temps libre.

Votre mission est de concevoir cette application en utilisant Jetpack Compose pour l'interface utilisateur et d'intégrer une API d'intelligence artificielle pour répondre aux besoins spécifiques des étudiants de l'ISEN.

Partie I : Découverte d'Android Studio et création d'une première interface

1. Initialisation du projet

- Créer un nouveau projet Android dans Android Studio. Utiliser Empty Compose Activity et nommer l'application ISENSmartCompanion. Pour le package name utilisé le format suivant : fr.isen.[nom].isensmartcompanion.
- Configurer le niveau d'API pour qu'il regroupe au moins 96 % des utilisateurs.
- Tester l'application sur un émulateur ou votre téléphone Android en mode développeur.
- Se familiariser avec les différents composants proposés par jetpack compose dans Android Studio (Text, Image, Button, Row, Column ….).
- Initialiser le projet git et créer un répertoire sur Github (Publique) qui sera lié à votre projet. Le lien du projet sera à rendre pour l'évaluation du TD.

2. Conception d'une interface - Page d'accueil (MainActivity)

- Dans la page MainActivity, créer son propre composant qui affichera à l'utilisateur un titre associé à un logo, un champ de saisie pour poser une question à l'assistant, un bouton pour envoyer et un texte pour afficher la réponse de l'IA (fictive pour le moment).

Aide : Consulter la documentation sur les composants de base de Jetpack Compose (Text, TextField, Row, Image) pour voir comment les organiser dans une Column.

Voici un exemple de cette première page.

3. Interaction utilisateur et gestion de l'état

- Configurer le bouton pour afficher un Toast indiquant "Question envoyée" lorsque l'utilisateur clique dessus. Faire en sorte que le texte de la réponse change en fonction de l'entrée de l'utilisateur dans le champ de texte.
- Utiliser les mots clés remember et MutableState, pour sauvegarder et afficher les modifications sur la valeur du texte précédemment renseigné par l'utilisateur.

Partie II : Introduction à la navigation et gestion de l'activité

1. Création d'une barre de navigation et démarrage d'une nouvelle activité

- Configurez une Navigation Bar pour naviguer entre plusieurs composants (composable). Celle-ci sera située en bas de votre page. Chaque composant représentera un menu auquel l'utilisateur aura accès. Les items du menu seront :

Voir pour créer des packages pour ces 4 éléments afin de ne pas tout mettre dans le main

Accueil (MainScreen), Événements (EventsScreen), Agenda (AgendaScreen), Historique (HistoryScreen)

- Dans l'écran des événements, créer un bouton (qui sera remplacé par la suite) pour accéder à une nouvelle activité. Cette activité affichera le détail de l'événement, EventDetailActivity.

Aide : Se documenter sur la Navigation avec Jetpack Compose utilisation (NavController, NavHost, NavigationBar, NavigationBarItem). Vous pouvez vous inspirer de l'article Medium suivant : https://medium.com/@jpmtech/jetpack-compose-bottom-navigation-bar-3e1e8749fb2c

2. Affichage de la liste des événements et de la page détail

- Créer l'interface pour afficher la liste des événements de l'ISEN. (Soirée BDE, Gala, journée de cohésion etc). Utiliser le composant LazyColumn et la fonction items pour votre liste.
- Créer une classe qui modélisera votre objet Event. Les champs de votre objet sont les suivants : id, title, description, date, location, category. Puis créer une liste fictive d'événement pour alimenter votre liste d'items.
- Au clic sur un événement de la liste, rediriger l'utilisateur sur l'activité EventDetailActivity créée précédemment en passant l'objet event en paramètre.
- Afficher le détail de l'événement dans cette nouvelle activité.

Partie III : Introduction aux web services et intégration d'une API d'IA

1. Récupération des événements via un web service

- Intégrer la librairie Retrofit et son convertisseur Gson, à votre projet pour récupérer la liste dynamique des événements via une requête web service.
- Créer l'ensemble des classes nécessaires pour réaliser votre requête Retrofit. L'url pour récupérer les événements est : https://isen-smart-companion-default-rtdb.europe-west1.firebasedatabase.app/events.json. La méthode à utiliser est GET et il n'y a pas de paramètre à ajouter.
- Parser le json avec la librairie Gson et afficher les résultats à la place des événements fictifs précédemment créés.

Aide : Utiliser une IA pour vous aider à implémenter votre requête Retrofit ou consulter la documentation de Retrofit : https://square.github.io/retrofit/

2. Utilisation de l'IA Gemini pour la page d'accueil

- Intégrer le SDK Google AI Client à votre projet en consultant la documentation : https://developer.android.com/ai/google-ai-client-sdk
- Utiliser le modèle Gemini 1.5 flash pour analyser le texte envoyé par l'utilisateur dans la page d'accueil.
- Récupérer la réponse de l'intelligence artificielle et l'afficher sous forme de liste de réponse. Cette liste sera enrichie à chaque nouvelle entrée utilisateur.

Partie IV : Gestion de l'historique et des notifications

1. Stockage local des interactions avec l'IA

- Implémentez une base de données locale avec la librairie Room pour sauvegarder les questions posées par l'utilisateur et les réponses générées par l'IA.
- Créer l'ensemble des classes Entity, DAO et RoomDatabase en suivant la documentation officielle de Room
- Sauvegarder chaque question/réponse échangée dans MainActivité dans votre base de données.
- Afficher l'historique dans l'onglet HistoryScreen en indiquant la date pour chaque échange.
- Donner la possibilité de supprimer un couple question/réponse ou l'ensemble de l'historique

2. Ajout de Rappels avec les Notifications

- Donner la possibilité à l'utilisateur d'être notifié d'un événement. Pour cela dans le détail d'un événement, ajouter une nouvelle icône qui indiquera si l'événement sera notifié ou non. Les données seront sauvegardées dans les préférences utilisateur.
- Implémenter l'envoie de notification Android pour chacun des événements épinglés par l'utilisateur. Envoyer la notification Android 10 secondes après que l'utilisateur s'est abonné.

Partie V : Création de l'agenda de l'étudiant et finalisation de l'application

Vous êtes plus libre sur cette dernière partie, où l'objectif est de personnaliser votre application par rapport aux autres.

Créer un nouveau menu Agenda à votre navigation qui permettra d'afficher les cours de l'étudiant, les événements auxquels il souhaite participer.

Rendu du TD

Le dépôt GitHub du projet contenant le code source de l'application ainsi que le fichier APK signé (release).

A rendre pour mercredi 17 avril 16h30

Ajouter le bouton pour créer un nouvel événement direct sur le calendrier + Mettre bulle de couleur sur les dates avec un event + Adapter la taille à chaque téléphone + Voir pourquoi la bdd par de zéro à chaque redémarrage