# Objectifs

* Navigation
* Fragments
* Exploiter une api
  + <https://pokemonsapi.herokuapp.com/>
* RecyclerView
* Librairie Picassa
* Librarie GSON
* Implémenter un formulaire
* Obtenir un *bearer token*
* Effectuer une requête authentifiée
* Utilisation de ViewModels

**NOTE**

**Un dépôt gitlab a été créé pour vous avec un projet initial. Le projet est réalisé de façon individuelle.**

<https://gitlab.com/cegepmaisonneuve/aut24/5d6/tp1>

**NOTE**

**Utilisez le patron de conception MVVM (UI dans activités/fragments, données dans le ViewModel et requêtes dans Repository).**

**Toutes les requêtes Volley sont exécutée sur le « IO thread » (viewModelScope.launch(Dispatchers.IO) { … })**

# LoginActivity

* Au départ de l’application, l’activité LoginActivity est chargée (voir le fichier AndroidManifest.xml).
* L’activité LoginActivity permet de s'authentifier à l'api et obtenir un jeton d'authentification (*bearer token*).

Une image contenant texte, capture d’écran, Police, conception

Description générée automatiquement

* L’utilisateur entre son courriel et mot de passe et clique le bouton login pour s’authentifier à la méthode [/auth/token](https://pokemonsapi.herokuapp.com/#/default/post_auth_token).
* Utilisez votre propre compte utilisateur (courriel *e1234567*@site.com et mot de passe e1234567 où 1234567 est votre numéro étudiant). Un compte a déjà été créé pour vous ou créer en un nouveau à l'aide de la méthode [/auth/register](https://pokemonsapi.herokuapp.com/#/default/post_auth_register_).
* Lorsqu’un jeton d’authentification est obtenu, stocker le jeton d'authentification (bearer token) dans la variable globale MainActivity.TOKEN et démarrez ensuite l’activité MainActivity.
* Affichez un message d’erreur (Toast) si l’authentification échoue.

# MainActivity

* Contient le code nécessaire pour charger le système de navigation de fragment d'android (navHostController).

# Fragment HomeFragment

* Affiche les pokémons obtenues par l'appel de [/pokemons](https://pokemonsapi.herokuapp.com/#/default/get_pokemons). Vous pouvez désérialiser les données en utilisant la classe Pokemon, écrite pour vous (package com.inf5d6.tp1.dataclasses).

Une image contenant jouet, Silhouette d’animal, dessin humoristique, clipart

Description générée automatiquement

* Un RecyclerView est utilisé pour afficher les images.
* La page Home affiche tous les pokémons (RecyclerView avec un layoutManager de type GridLayoutManager de 2 colonnes)
* Un clic sur une des pokémons permet la navigation vers le fragment DetailsFragment.

# Fragment DetailsFragment

* Ce fragment est chargé lorsque l'utilisateur clique un pokémon dans le fragment HomeFragment, dans le fragment FavoritesFragment ou encore sur une des évolutions dans le fragment DetailsFragment.
* Affiche les informations détaillées du pokémon obtenues par l'appel de la méthode [/pokemon?pokemonId=X](https://pokemonsapi.herokuapp.com/#/default/get_pokemon) où X est le pokemonId du pokémon. Vous pouvez désérialiser les données en utilisant la classe DetailsPokemon, écrite pour vous (package com.inf5d6.tp1.dataclasses).

Une image contenant oiseau, dessin humoristique

Description générée automatiquement

* La vue affiche le nom du pokémon, le nom de l'habitat du pokémon, le nom de l'espèce du pokémon, les pokétypes du pokémons séparés d'une virgule, et les images de ses évolutions, si présents. Il peut y avoir au maximum deux évolutions.
* Un clic sur une des évolutions affiche la page DétailsFragment pour ce pokémon.

# Fragment FavoritesFragment

* Consiste à exploiter l'api disponible à l'adresse /api/favorites pour afficher les pokémons favorites de l'utilisateur présentement connecté.
* Ceci est une api authentifiée, elle nécessite un *bearer token* pour fonctionner.
* Récupérez le *bearer token* de la variable globale MainActivity.TOKEN.

Une image contenant dessin humoristique, clipart

Description générée automatiquement

* Le format de la réponse est identique à l'api [/pokemons](https://pokemonsapi.herokuapp.com/#/default/get_pokemons), vous pouvez désérialiser la réponse de la même façon (Array<Pokemon>) et utiliser la même classe RecyclerViewAdapter comme adaptateur.
* Un clic sur une des pokémons permet la navigation vers le fragment DetailsFragment.

# Ajouter/supprimer pokémons favoris

* Dans le fragment DetailsFragment, ajoutez des boutons permettant d'ajouter ou de supprimer un pokémon de vos favoris. Un seul des boutons est visible à la fois.

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | |  |

* Pour déterminer quel bouton doit être visible, utilisez la méthode [/favorite?pokemonId=X](https://pokemonsapi.herokuapp.com/#/default/get_favorite) avec la méthode GET de l'api.
* Les méthodes permettant de consulter, ajouter ou supprimer un pokémons de la liste des pokémons favoris de l'utilisateur nécessite un jeton d'authentification *bearer token* pour fonctionner.
* Récupérez le *bearer token* de la variable globale MainActivity.TOKEN.
* Appelez [/favorite?pokemonId=X](https://pokemonsapi.herokuapp.com/#/default/get_favorite) avec la méthode **GET** pour déterminer si le pokémon dont le pokemonId est passé en paramètre fait partie des pokémons préférées de l’utilisateur.
* Un appel à [/favorite?pokemonId=X](https://pokemonsapi.herokuapp.com/#/default/post_favorite) avec la méthode **POST** permet d'ajouter le pokémon spécifiée en paramètre aux favoris de l'utilisateur.
* Un appel à [/favorite?pokemonId=X](https://pokemonsapi.herokuapp.com/#/default/delete_favorite) avec la méthode **DELETE** permet de supprimer le pokémon spécifiée en paramètre aux favoris de l'utilisateur.