TP 2, MMDO 2018

1. Finir le TP 1 si pas OK, rappel :
   1. Coder la page Home
      1. Ajouter une barre de recherche (composant fourni par ionic)
      2. Gérer la saisie utilisateur dans la barre de recherche (ngModel <https://angular.io/api/forms/NgModel> + ionInput, <https://ionicframework.com/docs/api/components/searchbar/Searchbar/>)
         1. Quand la query est vide (empty string), afficher « Aucun résultat » (<https://angular.io/api/common/NgIf>)
         2. Sinon afficher les faux résultats
   2. Naviguer vers la page Details depuis les résultats de recherche de la page Home (<https://ionicframework.com/docs/api/components/nav/NavPush/>)
   3. Coder la page Details
      1. Faire un template dynamique destiné à afficher les données du résultat sur lequel on a cliqué (title, author, date, image)
      2. Passer les données du résultat sur lequel on a cliqué (navParams <https://ionicframework.com/docs/api/components/nav/NavPush/>)
      3. Récupérer les données en question dans la page Details (<https://ionicframework.com/docs/api/navigation/NavParams/>)
2. Dans main.ts, tout au début, ajouter les lignes suivantes :  
   import 'rxjs/add/observable/of';

import 'rxjs/add/operator/pluck';

1. Dans app.module.ts, importer/déclarer HttpClientModule depuis @angular/common/http
2. Créer un fichier nommé tmdb.ts dans le répertoire app/, celui-ci doit être ignoré par Git (.gitignore) et doit exporter une constante contenant votre API key TMDB
3. Modifier l’interface Result créée au TP 1 pour refléter un vrai résultat renvoyé par l’API search de TMDB (<https://developers.themoviedb.org/3/search/search-movies>), modifier également vos faux résultats pour que tout fonctionne comme avant
4. Dans home.ts, créer une méthode nommée fetchResults, celle-ci doit retourner un Observable<Result[]> (Observable est à importer depuis rxjs/Observable) en utilisant HttpClient (<https://angular.io/api/common/http/HttpClient>) et HttpParams (<https://angular.io/api/common/http/HttpParams>) - tous deux à importer depuis @angular/common/http - pour interroger (http GET) l’API search TMDB mentionnée ci-dessus
5. Modifier la logique de recherche dans home.ts pour assigner soit Observable.of([]), soit le résultat de l’appel à la méthode fetchResults en guise de résultats en fonction de ce que contient la barre de recherche, supprimer les fausses données créées au TP 1
6. Utiliser le pipe async (<https://angular.io/api/common/AsyncPipe>) directement dans home.html sur \*ngIf et \*ngFor pour pouvoir afficher les données récupérées depuis TMDB
   1. Pour les images : <https://developers.themoviedb.org/3/getting-started/images> et <https://www.themoviedb.org/talk/53c11d4ec3a3684cf4006400>
   2. Optionnel : utiliser le pipe date (<https://angular.io/api/common/DatePipe>) dans le template pour mieux formatter les dates
7. Modifier details.html pour afficher plus d’informations parmi celles disponibles, toujours sur le même principe
8. Commit, push