

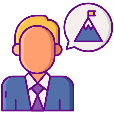
RAPPORT

TRAVEL TRACK

Réaliser par: - SBAI SALAH

* MHARZI CHAYMAE

Professeur: **Mme Roubi Sara**



IDÉE D'APPLICATION : USER STORY

***SALAH ET CHAYMAE*** avaient toujours été des voyageurs passionnés, visitant de nouvelles villes et pays à chaque occasion. Cependant, ils se retrouvaient souvent à avoir du mal à se souvenir des noms des endroits où ils étaient allés et des itinéraires qu'ils avaient pris pour y arriver. C'était frustrant pour eux de ne pas pouvoir se rappeler des détails de leur voyage lorsque quelqu'un leur en parlait plus tard.

Un jour, en discutant de leurs problèmes de voyage avec leur ami, celui-ci leur a suggéré d'utiliser une application de suivi de voyage. ***SALAH ET CHAYMAE*** étaient intrigués et ont décidé d'essayer. Ils ont téléchargé une application qui pouvait suivre leur position et les endroits qu'ils avaient visités pendant leur voyage, mais avec la possibilité de garder leurs souvenirs pour eux-mêmes sans les partager. Ils appréciaient l'idée de conserver leurs expériences de voyage privées.

L'application s'est avérée très utile pour ***SALAH ET CHAYMAE***. Ils pouvaient facilement revenir sur leur voyage et se souvenir de tous les détails, tout en gardant la possibilité de ne pas partager leurs souvenirs avec les autres. Ils pouvaient voir les itinéraires qu'ils avaient pris, les noms des endroits qu'ils avaient visités et même les photos qu'ils avaient prises, tout en choisissant ce qu'ils souhaitaient partager ou garder privé.

Cela leur a permis de se sentir plus en contrôle de leurs souvenirs de voyage et de les garder précieusement pour eux-mêmes. L'application est devenue un outil essentiel pour leurs voyages, leur permettant de profiter pleinement de leurs expériences **sans se sentir obligés de les partager avec le monde entier.**

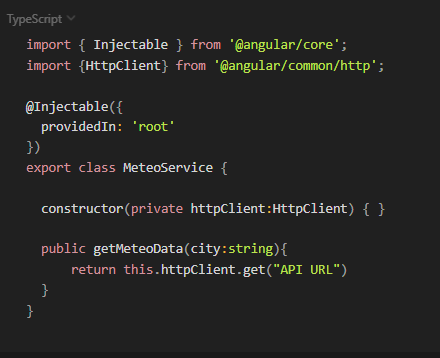
**THE APP INTERFACES: PAGES**

* SPLASH SCREEN
* AUTHENTICATION PAGE : FIREBASE
* PROFILE PAGE
* HOME PAGE WITH MENU : Side Menu (coder from scratch)
* WEATHER PAGE (API) : Voir la météo.
* GALLERY PAGE (API) : Voir les photos.
* LOCATIONS PAGE : PLACES YOU VISITED
* NEW LOCATION PAGE
* LOCATION DETAILS



1- Intégration de la météo

La page Météo de l'application de voyage utilise une intégration d'API pour récupérer les données météorologiques en temps réel. L'API *OpenWeatherMap* est utilisée pour obtenir les informations météorologiques en fonction de la destination sélectionnée par l'utilisateur. L'application effectue des requêtes HTTP vers l'API, récupère des réponses JSON et analyse les données pour afficher des détails météorologiques pertinents tels que la température, l'humidité, la vitesse du vent et les prévisions.



meteo.service.ts

Et pour rendre le code plus facilement maintenable, réutilisable et testable, j'ai placé le code responsable des appels HTTP à l'API dans un service.



2- Intégration du gallery

La page de galerie de l'application utilise l'API *Pixabay* pour afficher des images basées sur un mot-clé. Les utilisateurs peuvent effectuer des recherches, charger plus d'images et parcourir la galerie.

Au chargement de la page, les variables sont initialisées, telles que le mot-clé de recherche, le numéro de page actuel, la taille des résultats et les tableaux de données d'images. L'utilisateur peut effectuer une recherche en utilisant le mot-clé spécifié.

Lorsque la fonction "onLoadImages()" est appelée, les données d'images sont réinitialisées et une recherche est effectuée en utilisant le mot-clé et les paramètres de pagination. L'API Pixabay est interrogée avec l'URL appropriée, et les résultats sont récupérés via une requête HTTP.

La fonction "doSearch()" envoie une requête HTTP GET à l'API en incluant le mot-clé de recherche, la taille des résultats et le numéro de page actuel. Les données récupérées sont traitées, et chaque image est ajoutée au tableau de données d'images. Une fois que la requête est effectuée avec succès, la fonction traite les données renvoyées par l'API. Elle parcourt les résultats d'images (hits) et les ajoute au tableau `dataImages`. Elle calcule également le nombre total de pages de résultats en utilisant la formule `totalHits / size`, où `totalHits` représente le nombre total d'images correspondant au mot-clé de recherche et `size` est le nombre d'images par page.

La fonction "loadData(evt)" est utilisée pour charger plus de données lorsque l'utilisateur fait défiler la galerie. Si le nombre de pages actuel est inférieur au nombre total de pages disponibles, une nouvelle requête de recherche est effectuée pour la page suivante. L'événement "evt" est utilisé pour notifier la complétion du chargement des données.

Le processus se répète jusqu'à ce que toutes les pages aient été chargées. Lorsque le nombre de pages actuel atteint ou dépasse le nombre total de pages, le chargement des données est désactivé.

3- GESTION D’EMPLACEMENT: Location Page

La méthode ionViewWillEnter() est un événement du cycle de vie d'Angular qui est appelé juste avant que la page ne soit affichée. Elle appelle la méthode onGetLocations() pour récupérer les emplacements à afficher.

Nouvelle Location: La méthode `onNewLocation()` est utilisée pour naviguer vers la page de création d'un nouvel emplacement lorsque l'utilisateur souhaite ajouter un emplacement. Elle fait appel au service `Router` pour effectuer cette navigation.

Détails d’une Location: D'autre part, la méthode `onDetailLocation(p: Place)` est utilisée pour afficher les détails d'un emplacement spécifique. Elle stocke l'emplacement sélectionné dans la variable `currentLocation` du service `LocationsService` et utilise le service `Router` pour naviguer vers la page des détails de l'emplacement.

La méthode publique `onTakePicture()` est appelée lorsque l'utilisateur souhaite prendre une photo de l'emplacement. Elle affiche une boîte de dialogue (AlertController) pour permettre à l'utilisateur de choisir entre la caméra et la bibliothèque de photos comme source de l'image. En fonction du choix de l'utilisateur, la méthode `getPicture()` est appelée avec les options appropriées pour capturer l'image. La méthode privée getPicture utilise le service Camera pour capturer une image en utilisant les paramètres spécifiés. L'image capturée est convertie en format base64 et ajoutée à l'emplacement actuel à l'aide du service LocationsService.

|  |  |
| --- | --- |
| SPLASH SCREEN | SQUELETTE |
|  |  |
| LOGIN PAGE | HOME PAGE |
|  |  |
| METEO PAGE : BEFORE | METEO PAGE : AFTER |
|  |  |
|  |  |
| GALLERY | LOCATIONS |
|  |  |
| NEW LOCATION : BEFORE | NEW LOCATION : AFTER |
|  |  |
| LOCATION DETAILS | CAMERA |
|  |  |