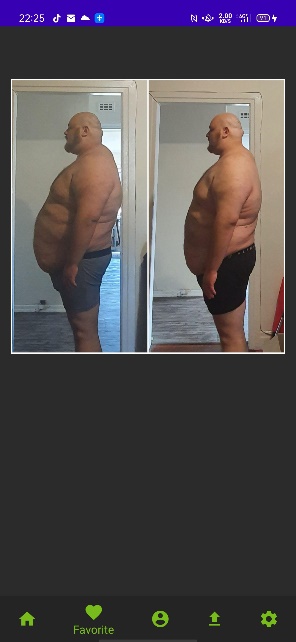
*DOCUMENTATION EPICTURE*

1. User utilisation:
2. The login page:

The login is the first page that the User will see, this page only asks to the user to click on the login button, he will be redirected to his browser and after that, he will be able to use all the functionality of our application.

1. The home page:

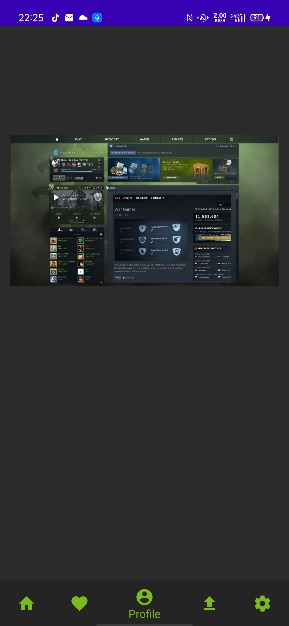
The home page simply gives you all the most trending posts made on imgur, with the possibility to click on an image to see the number of likes and comments, and the possibility to add the image in your favorites.

1. The “favorites” page:

This page lists you all your images added in favorites.

You can click on them and remove it from your favorites if you want to.

1. The profile page:

The profile page lists all the post that you made on imgur, you can check the number of likes and the number of comments directly from this page.

1. Developer documentation

Lists of classes and their usage:

Class MainAdapter:

File: adapter.kt

*/\*\*  
 \* An adapter for images.  
 \*  
 \* The class is to make the images visible on a page.  
 \*  
 \** ***@property*** *context which page on the app.  
 \** ***@property*** *imageList list of all the images got from the imgur API.  
 \*/*class MainAdapter(val imageList: List<Content>?, val context: Context): RecyclerView.Adapter<CustomViewHolder>() {

Class Constants

File: Constants.kt

*/\*\*  
 \* Constants class.  
 \*  
 \* Every constant infos used in the app.  
 \*/*public class Constants {

Class DetailImage

File: DetailImage.kt

*/\*\*  
 \* Get the images details.  
 \*  
 \* The class is to get the infos of the image, like the number of likes and comments.  
 \*/*class DetailImage : AppCompatActivity() {

Class FavFragment

File: FavFragment.kt

*/\*\*  
 \* A fragment for favorites.  
 \*  
 \* The class is to add the favorites button, and make it work in async.  
 \*/*class FavFragment : Fragment() {

Class HomeFragment

File: HomeFragment.kt

*/\*\*  
 \* An fragment for the Home page.  
 \*  
 \* The class is to create the home page and add the images, and make it work in async.  
 \*/*class HomeFragment : Fragment() {

Class HomeVc

File: HomeVc.kt

*/\*\*  
 \* Home page.  
 \*  
 \* Handle the navigation and the fragments.  
 \*/*class HomeVc : AppCompatActivity() {

Class Content

File: Content.kt

*/\*\*  
 \* Infos of an image.  
 \*  
 \* Put all the infos from an imgur image into a class.  
 \** ***@property*** *id The ID of the image.  
 \** ***@property*** *title The title of the image.  
 \** ***@property*** *description The description of the image.  
 \** ***@property*** *link The URL of the image.  
 \** ***@property*** *type gif/jpeg or png  
 \** ***@property*** *ups The upvotes of the image.  
 \** ***@property*** *downs The downvotes of the image.  
 \** ***@property*** *points The score of the image.  
 \** ***@property*** *favorite The number of time the image has been added as favorite.  
 \** ***@property*** *is\_album If the page multiple images.  
 \** ***@property*** *images list of [ImageLib].  
 \*/*public class Content(id: String, title: String?, description: String?, link: String?, type: String?, ups: Int, downs: Int,  
 points: Int, favorite: Boolean, is\_album: Boolean, images: List<ImageLib>?): Serializable {

Class ImageLib

File: ImageLib.kt

*/\*\*  
 \* Infos about imageList.  
 \*  
 \* All the infos, images per images, so it keeps track of each singular photos, even in an album.  
 \** ***@property*** *id The ID of the image.  
 \** ***@property*** *title The title of the image.  
 \** ***@property*** *type gif/jpeg or png.  
 \** ***@property*** *animated is the image a gif or not.  
 \** ***@property*** *favorite if the image in the favorties of the user or not.  
 \** ***@property*** *link The URL of the image.  
 \** ***@property*** *ups The upvotes of the image.  
 \** ***@property*** *downs The downvotes of the image.  
 \** ***@property*** *points The score of the image.  
 \*/*class ImageLib(id: String?, title: String?, type: String?, animated: Boolean,  
 favorite: Boolean, link: String?, gifv: String?,  
 ups: Int, downs: Int, points: Int?, score: Int?): Serializable {

Class ImageResponse

File: ImageResponse.kt

*/\*\*  
 \* Infos given by the image request.  
 \*  
 \* Each requests give info for an image, this class keeps track of it.  
 \*/*public class ImageResponse {

Class ImgurClient

File: ImgurClient.kt

*/\*\*  
 \* The imgur client with the list of the requests.  
 \*  
 \* The imgur client with the list of all the possible requests for every implementation.  
 \*/*public interface ImgurClient {

Class ImgurResponse

File: ImgurResponse.kt

*/\*\*  
 \* The data the imgur returns  
 \*/*class ImgurResponse {

Class MainActivity

File: MainActivity.kt

*/\*\*  
 \* Main page  
 \*  
 \* Where the request for the imgur authentification is made.  
 \*/*class MainActivity : AppCompatActivity() {

Class PhotoAdapt

File: PhotoAdapt.kt

*/\*\*  
 \* The details of the image  
 \*  
 \* When you click on an image, this is to see all the infos just under the image.  
 \*/*class PhotoAdapter(val contentData: Content?, val context: Context): RecyclerView.Adapter<CustomDetailViewHolder>() {

Class Preference

File: Preference.kt

*/\*\*  
 \* Preference class  
 \*  
 \* Keeps track of the OAuth 2 Tokens of the user.  
 \** ***@param*** *context the actual page that the user is seeing right now  
 \*/*class Preference constructor(context: Context) {

Class ProfilFragment

File: ProfilFragment.kt

*/\*\*  
 \* ProfilFragment class  
 \*  
 \* Keeps track of the user information and put it in a Fragment.  
 \*/*class ProfilFragment : Fragment() {

Class SettingsFragment

File: SettingsFragment.kt

*/\*\*  
 \* SettingsFragment class  
 \*  
 \* Fragment for the setting page.  
 \*/*class SettingsFragment : PreferenceFragmentCompat() {

Class UploadFragment

File: UploadFragment.kt

*/\*\*  
 \* UploadFragment class  
 \*  
 \* Fragment for the upload page.  
 \*/*class UploadFragment : Fragment() {

Class UploadResponse

File: UploadResponse.kt

*/\*\*  
 \* UploadResponse class  
 \*  
 \* Class to keep track of the info given by the upload response.  
 \*/*public class UploadResponse {