**Sela angular project.**

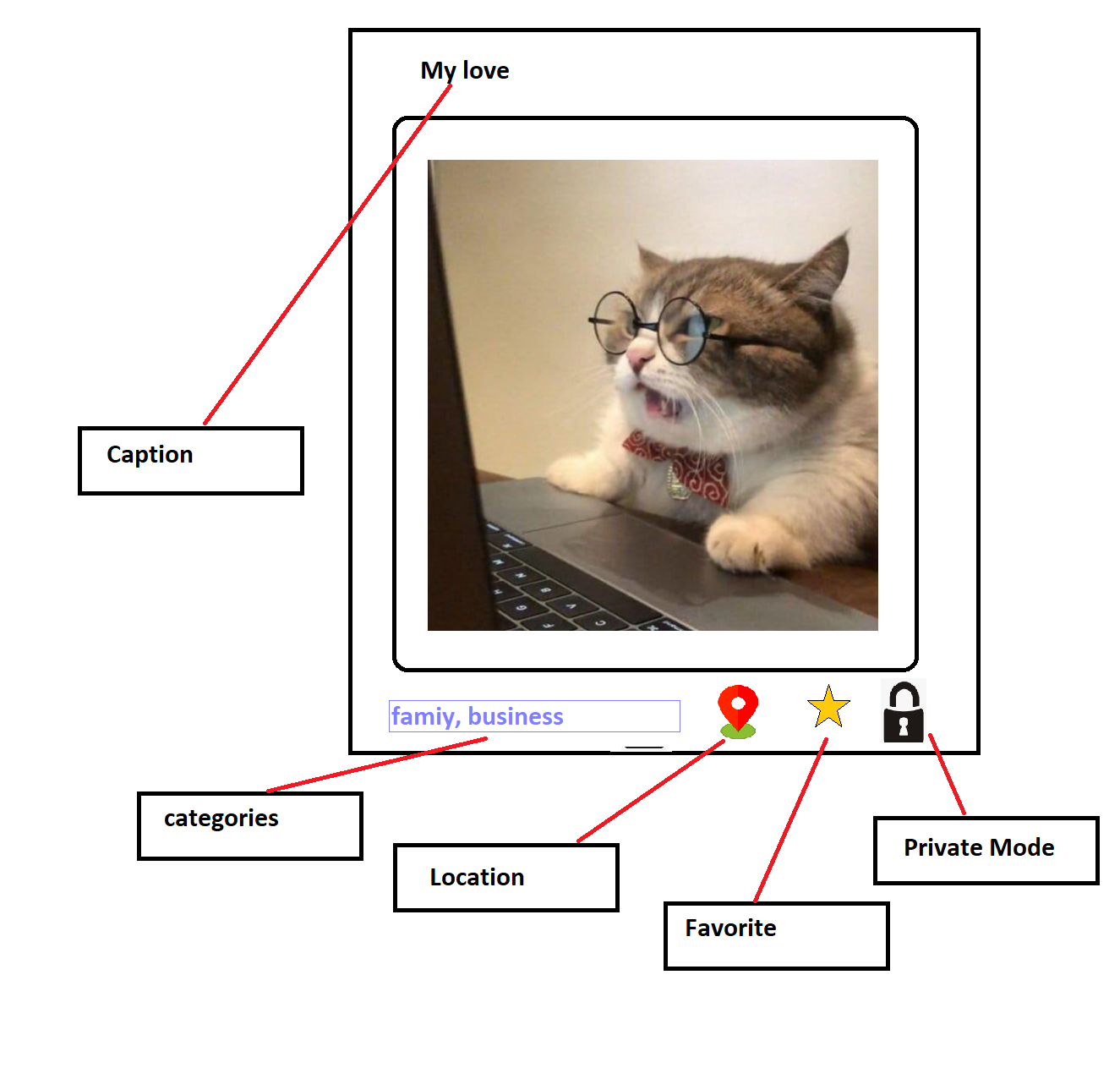


**Specification and user story**:

Sela album hub is a magnificent solution for customers who love to store photos.

No more mess with your local photos on your computer. This app has a single source of all your photos.

Any photo will include in edition,



**Caption** – headline the name of a specific image object.

**Location –** get location from google maps api and allow user to choose location

**Favorite** - ability to nark image as favorite (star)

**Categories -**  one of the major advantages of this app, save your image with one or many categories you pre defined, then you can later search items according to specific category. (drop down control)

**private** mode option..

Why is the Sela Album hub so special???

**Adding photos from everywhere !!**

**Sela album hub** support adding new images from local-device,

from online search (**Last feature to implement**)

, and even from your local device camera!

**Search**

**Sela album hub** provide smart search engine among your photos by – name / category /

**Private Mode**

**Sela album hub** provides a private mode where you can hide your private data from unwelcome guests, and show it only with password.

**Random show**

**Sela album hub** can toggle your memories in autoplay mode with totally random appearance.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Basic guidelines:**

\* User can manage personal images

\* Application will hold a data folder on dedicated location on personal computer

\* Adding new images into application collection can be done by:

1: local device camera,

2: copy from somewhere over local storage

3: search online via built in integrated search control (google style + free api)

\*user can edit any image for :

1: adding Title

2: mark as favorite (aka yellow star)

3: categorize (like family, trips, work etc..)

\* user can watch images by one of few available layout :

Grid, vertical/horizontal list, single image,

\* application provide built in random show of images (screensaver style)

\*option to mark image as private (images will be hidden from show )

\*private mode will be available by special password

\*application will support few themes (light , dark , standard)

\* search with autocomplete by categories.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Technical perspective:**

Angular is a client side framework, it is unable to access local hard drive in order to read/write files.

Nodejs however, is perfect for this purpose. (and we’re just getting started)

So – here is list of technical issues and demands-

You must complete accomplished the entire list in as POC projects (proof of concept),

Just before getting into real application action.

Your toolbox for this application are:

Angular framework, Nodejs framework, Any NPM package, Browser API. And some public free api’s around the net, like:

Images:

<https://www.pexels.com/api/>

<https://unsplash.com/developers>

maps:

<https://angular-maps.com/guides/getting-started/>

Json:

https://jsonplaceholder.typicode.com/todos/1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**List of possible challenges:**

- Acquiring files from local machine in HTML Form

- Writing files to a local machine with NodeJS.

- Running Camera from Angular, get the image (if taken) and send it to Nodejs service.

- Find public free API for images (supporting search by name) download images and send it to Nodejs service

-Location **–** get location from google maps api and allow user to choose location and save it

-Save location, caption and all other properties in close relation to physical images

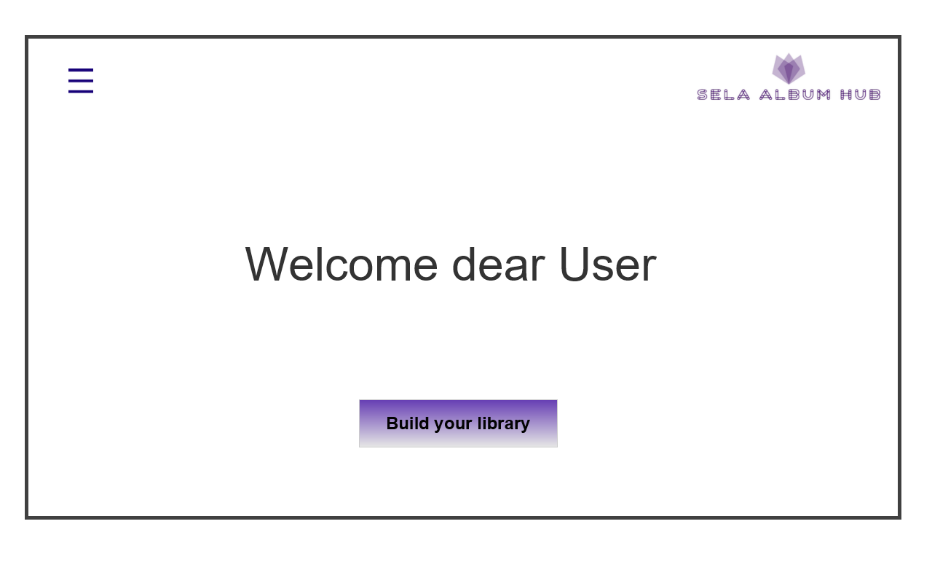
-Layout (css) make application shows images in few templates (grid\list)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Full application standards.**

1. First Running.

User open up SELA ALBUM HUB for first time will face next screen:

****

Users must click on the BUILD YOUR LIBRARY button, for the very first time.

Then we jump into **screen 02**, and select some options..

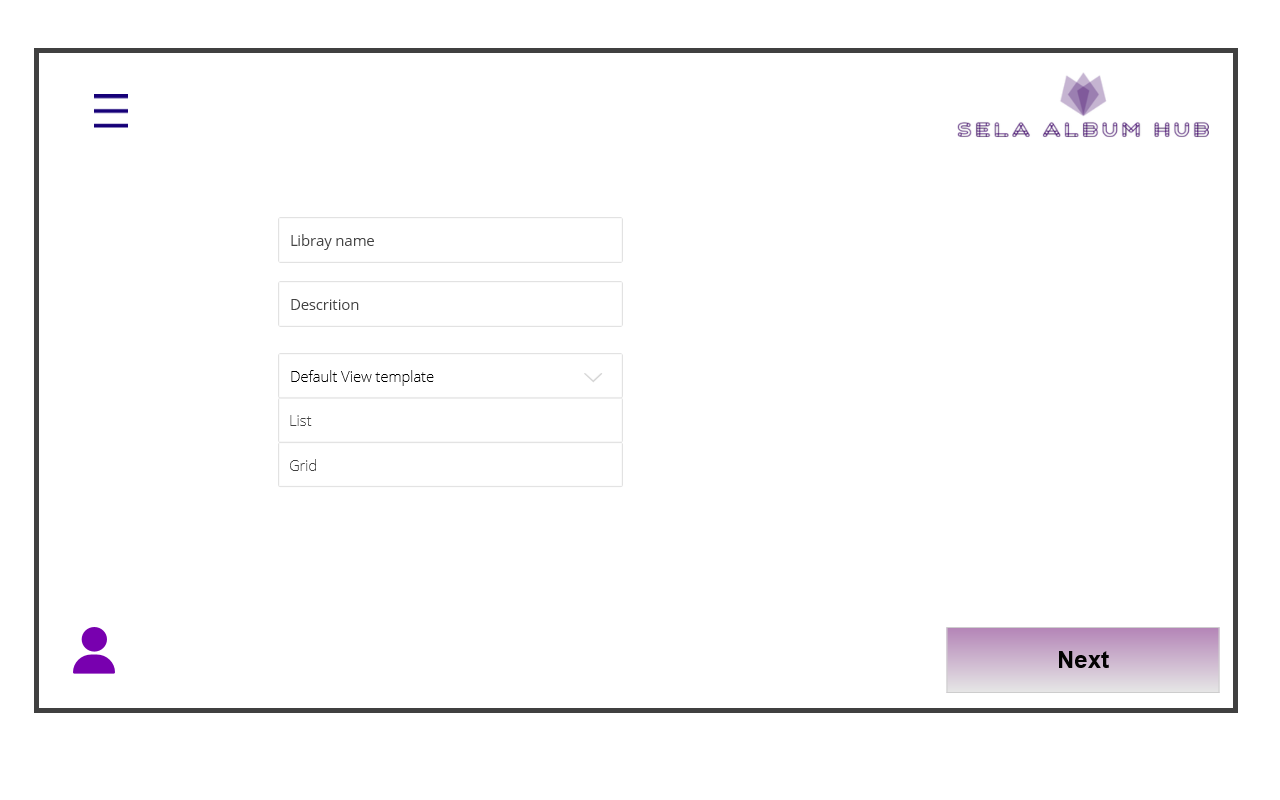


**\*\* this page, (allow camera etc…) is UI only for now (real connection to web browser is optional)**

Then by clicking on the next **button** we jump into **screen 03** and add some more details..

Library name will be the default name appears later over the collection, description is optional,

Default template will effect on how you gonna see the images by default, it’s possible to change this template later from application buttons..



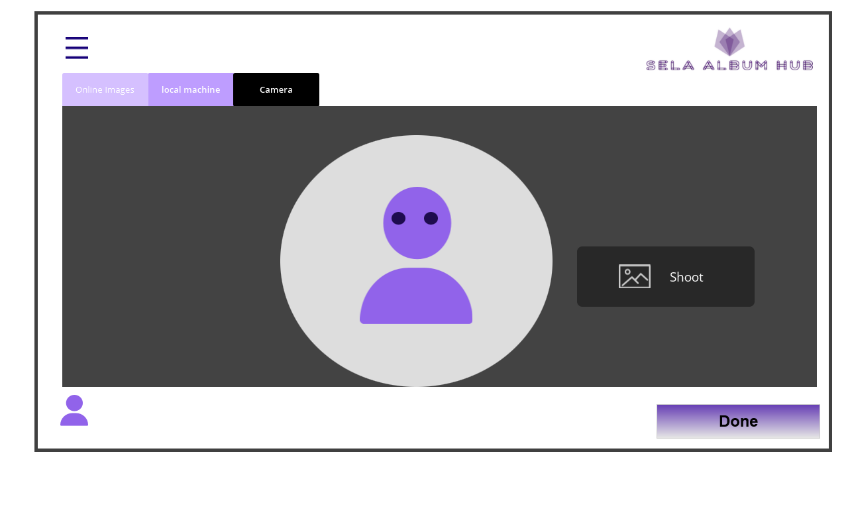
After click the **next button** in screen 03 we will jump into **screen 04** as follow,

Screen 04 is made with Tabs control, and it’s about adding images into SELA ALBUM HUB

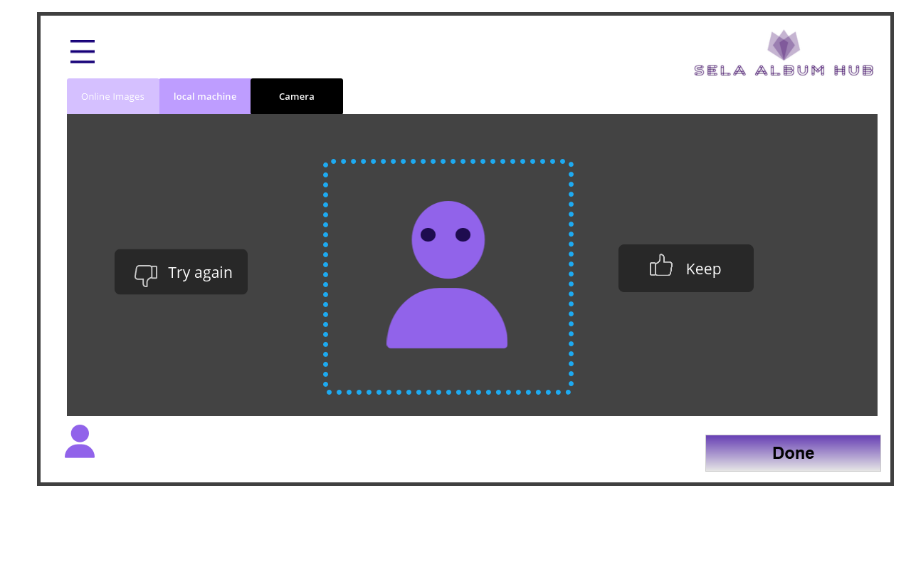
User can add those images either from online, local computer and directly from device camera

For camera specifically first take a photo than see a preview and decide to save or trash and retry

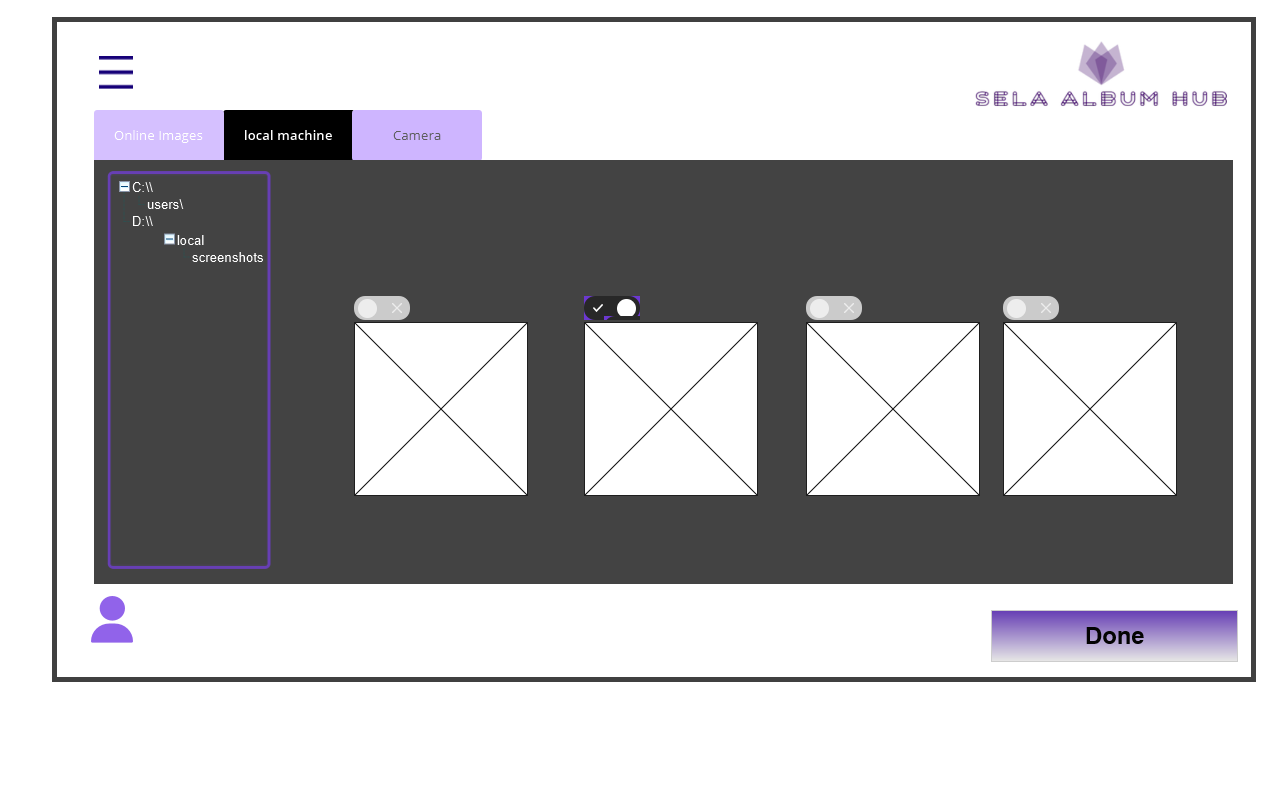
**SCREEN 04 /1**



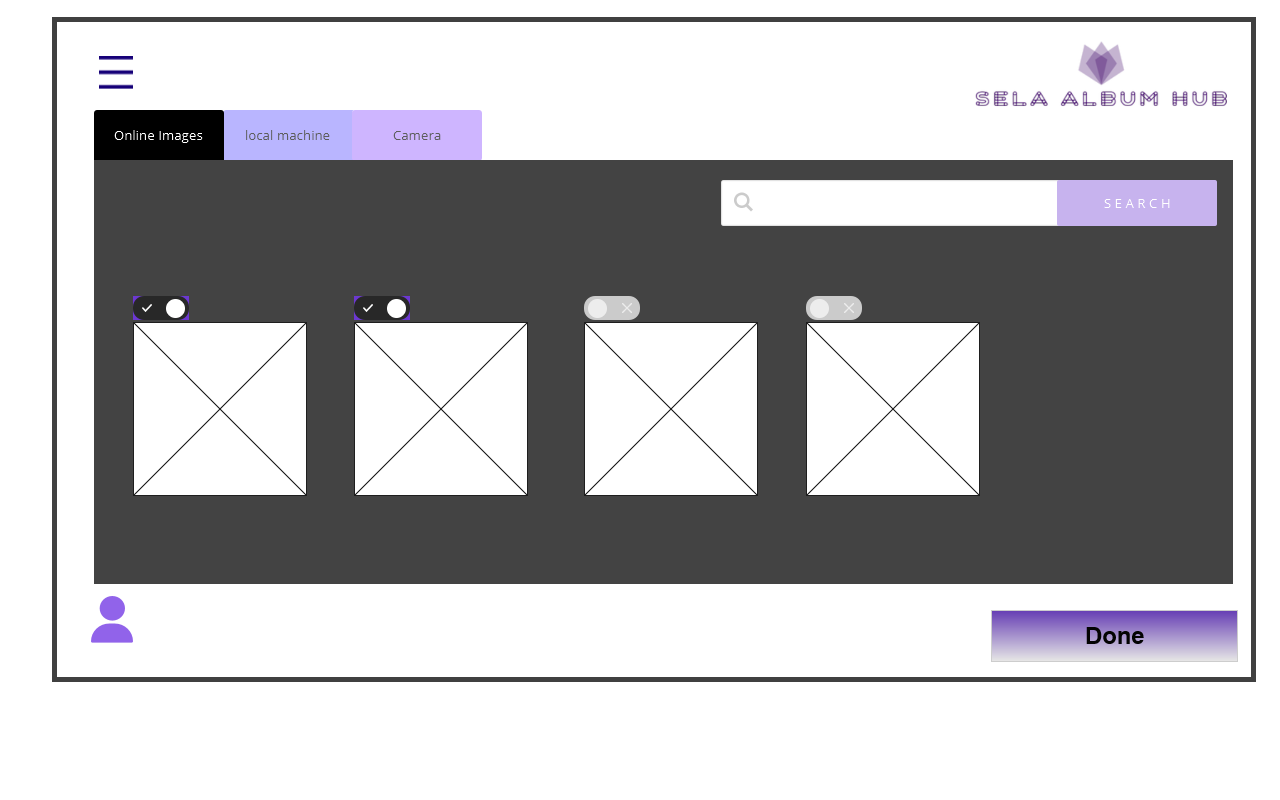
**SCREEN 04/2**

****

**SCREEN 04/3**

****

**SCREEN 04/4**

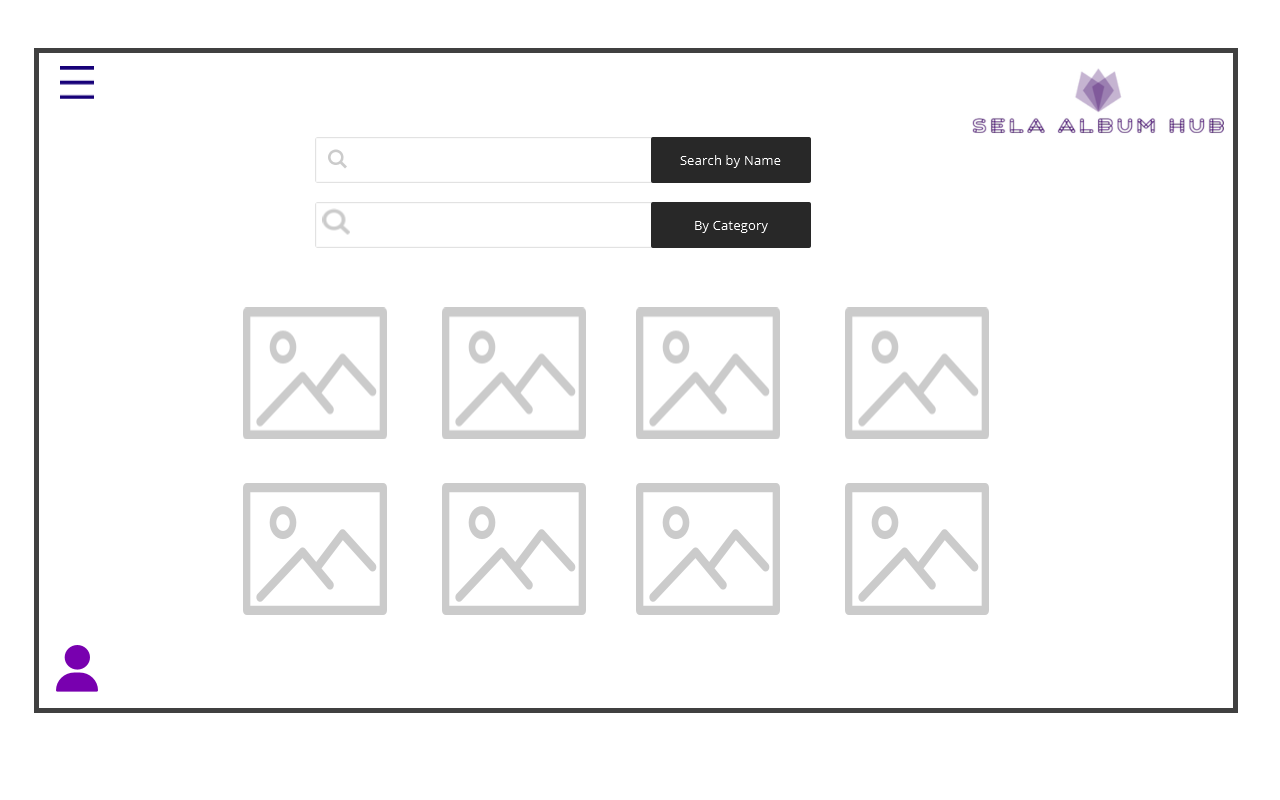


This behavior is about, “OK before we start, let's add some items into application library”

And let the user have some knowledge about application capabilities..

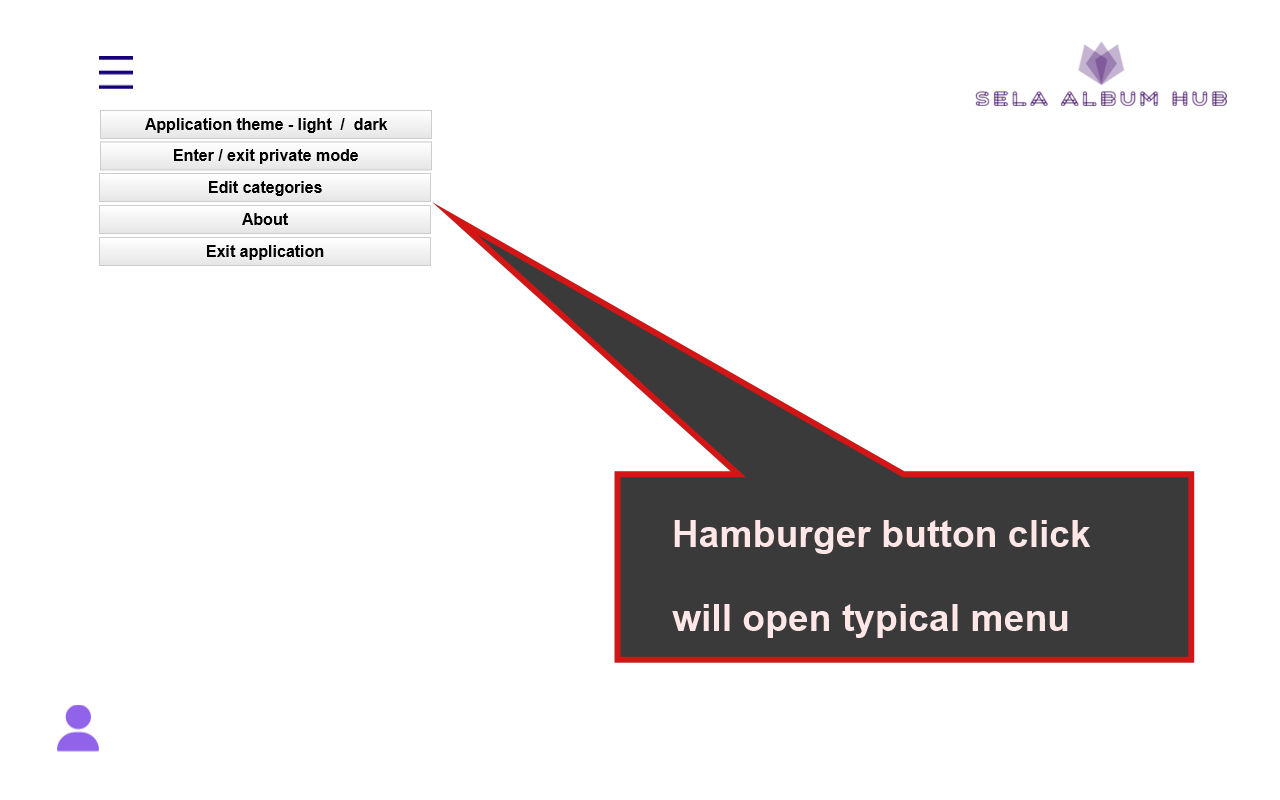
However, users can always press on the DONE **button available in SCREEN 04** and **skip the current** stage !!

DONE button will jump the user into main application interface screen:

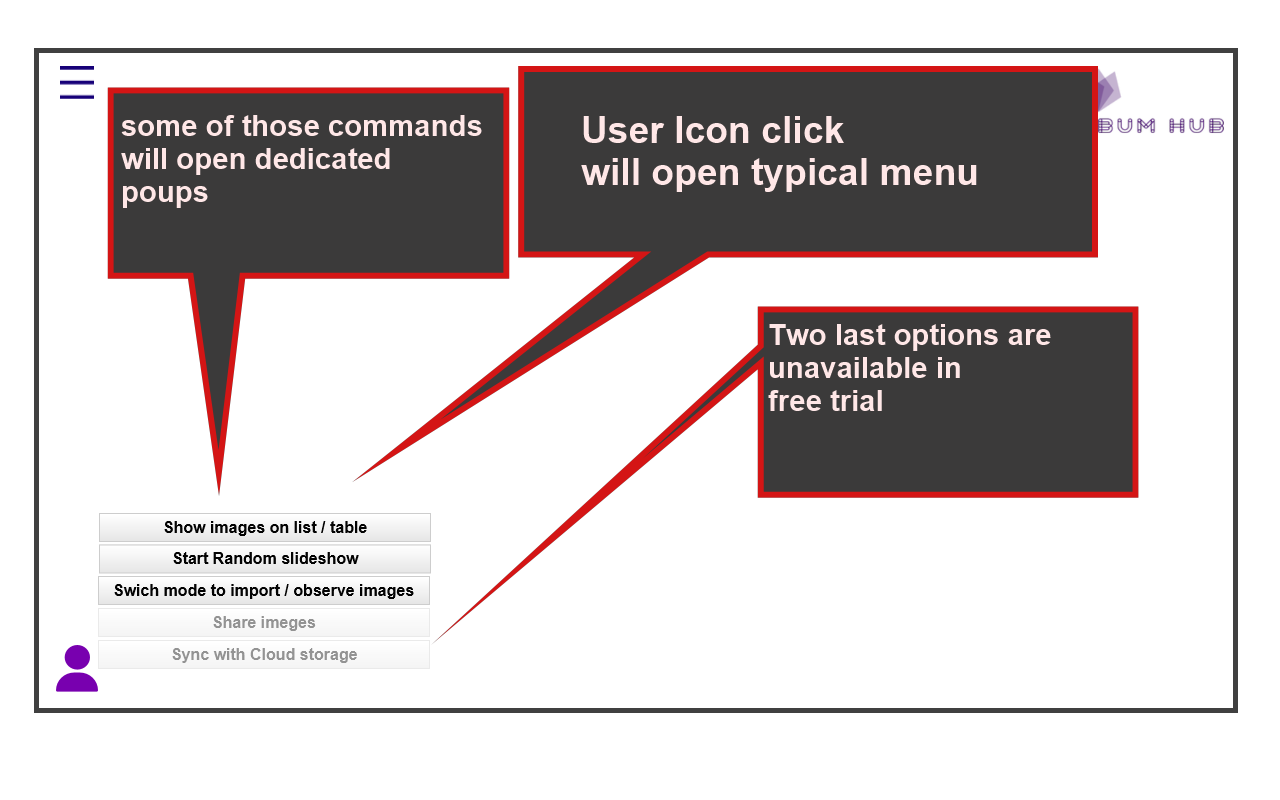


SELA ALBUM HUB also have 2 permanent menus as below:

The Top left **Hamburger Menu:**



And the bottom left User Menu:



Where users can customize the default layout of images, and start the random slideshow mode.

Another important action from this menu is import images mode as Application startup which allows users to select new images from web/local/camera.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Private mode**

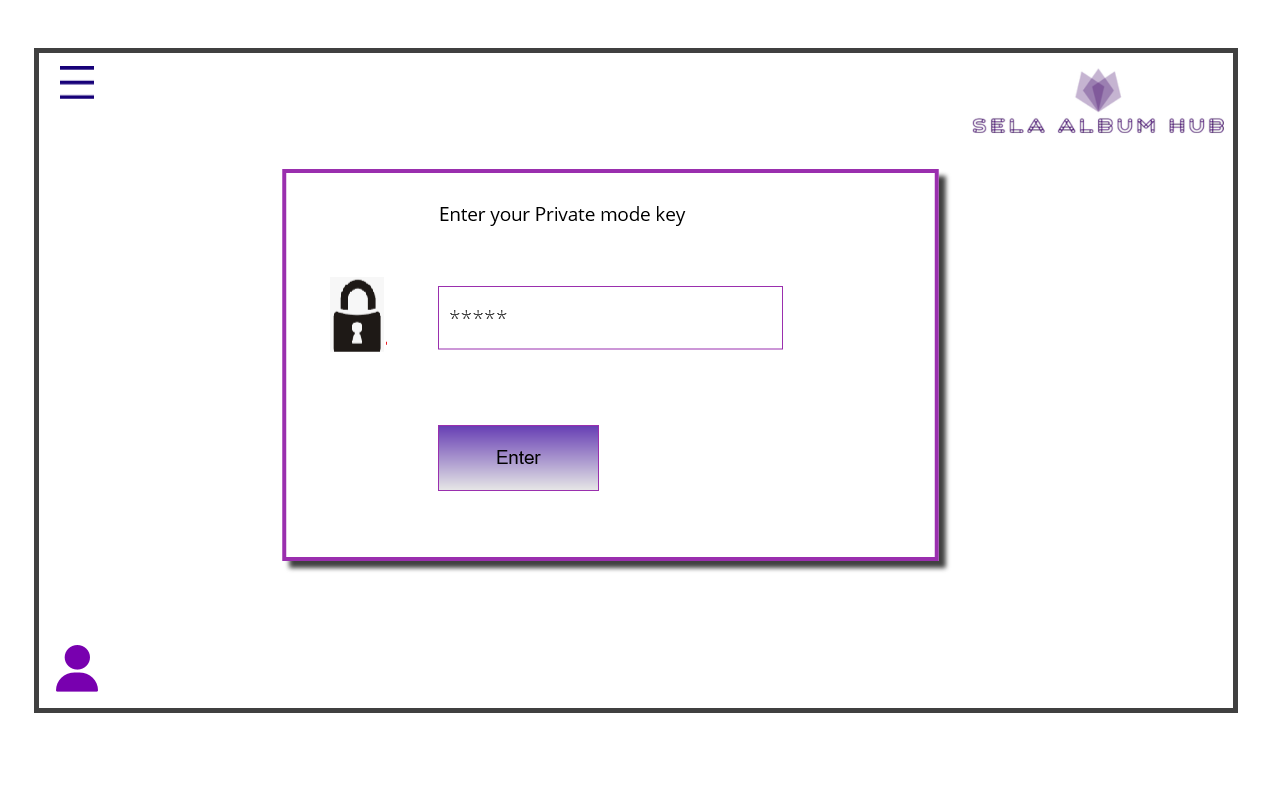
Entering the SELA ALBUM HUB private mode from the hamburger menu item by password.

**For the first time ever user will see warning**

**“no private mode configured… please set first time (and repeat for validation)”**

The user private password will kept on server ( same as all application data)

For now , no option to change password after declaration.

****

From private mode, exit is only from hamburger menu



In general any image can declare as private mode then it will disappear from general album and appear only under private mode state (and opposite) (Editing mode will work in private mode as in regular)

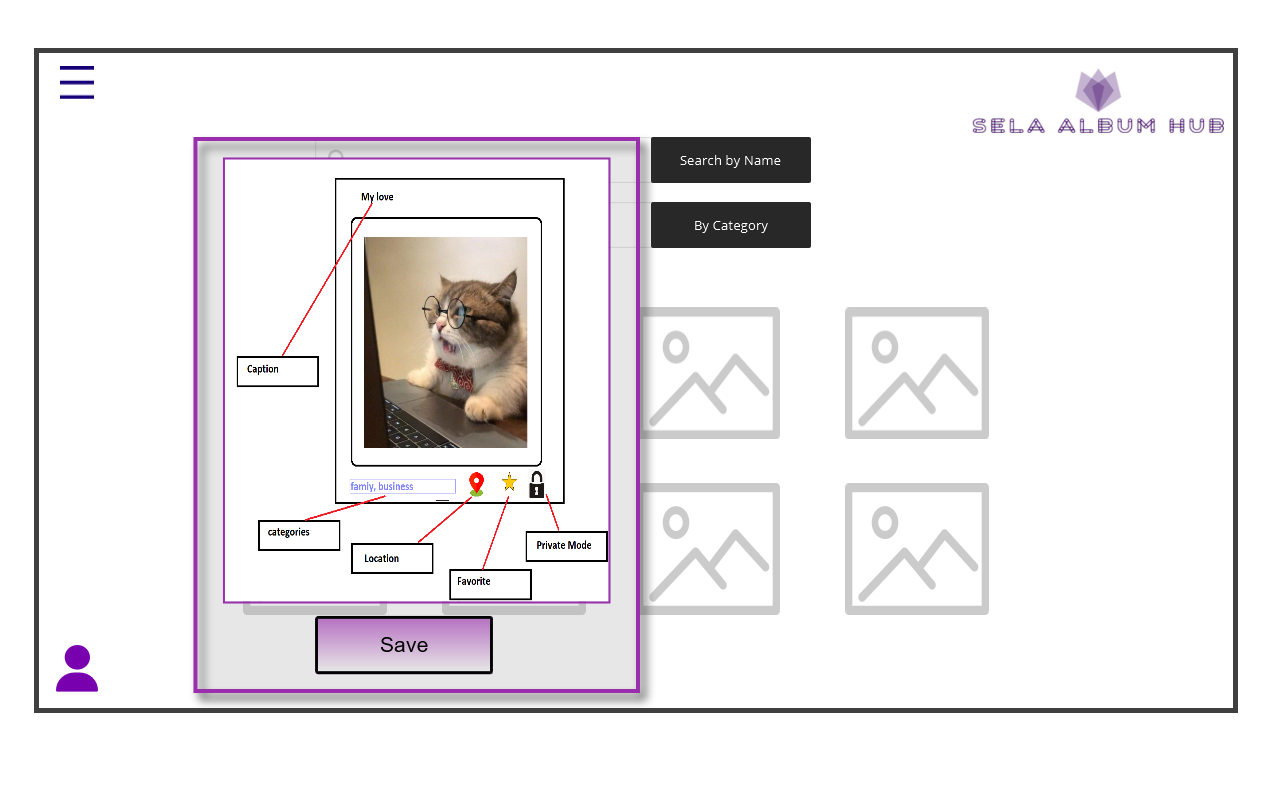
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Editing Mode**

As appeared in this document's first page, Image has few extra properties.

Any of those properties turn into clickable buttons and not just icons inside **EDITING MODE**. To enter edit mode just double click on any image from the list,

Than, the selected image will popup in editing view and we are in **editing mode**



Any of the image properties are editable, but changes will be kept and transferred to the server only by clicking on **Save Button.**

**Private mode is on-off button (lock icon will change to open or transparent icon).**

**Favorite is the on-off button (the star will remain uncolored).**

**Location will open up small map control (google maps style) and allow users to allocate some place and keep it.**

**Categories is a drop-down control with multiple selection. (categories can be edited in separate form as part of hamburger menu but no specification for** UX **in this document)**

**Caption is just a text box.**

**Additional Terms and Notes:**

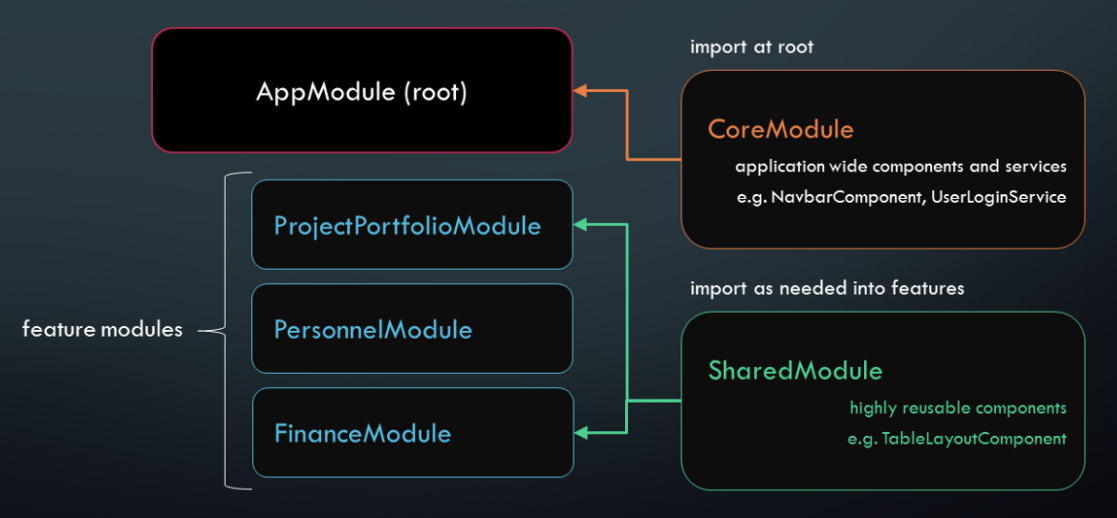
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Application architecture and coding standards:**

**Architecture:**

(only if relevant create few modules)



**State management:**

**You can use a simple service class to manage application state as much as relevant.**

**Or learn about frameworks like NGRX**

**Component architecture:**

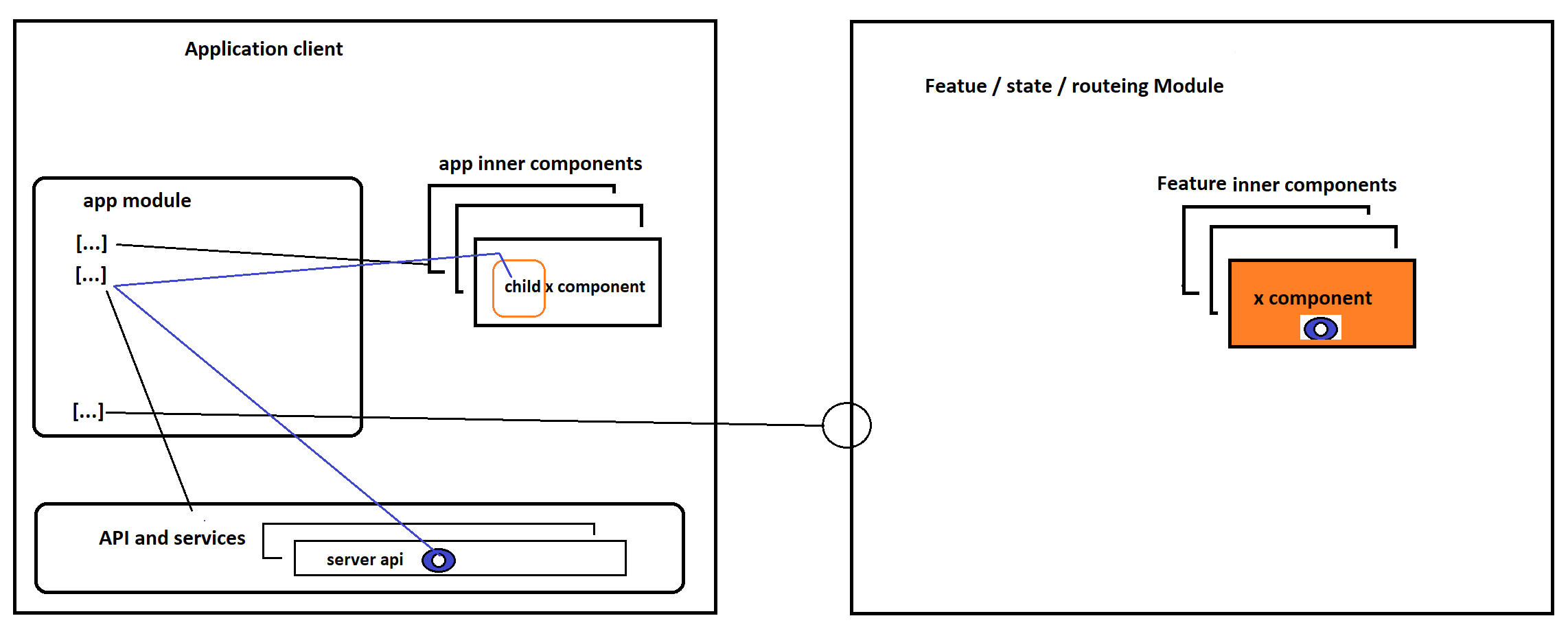
In this schema, pay attention for orang x component, declared as part of feature module,

Then the entire module is exported and imported into the app module.

And one of the app child components hosts the x component..

Then we got an Injectable purple service as part of the app component.

This service provides it’s data into the X component by angular Input data binding from parent to child.



**General guidelines:**

Clean code, small and atomic functions, prefer readable code over tricky one.

Use small components, move logic into services.

Css inline html is not legitimist at all