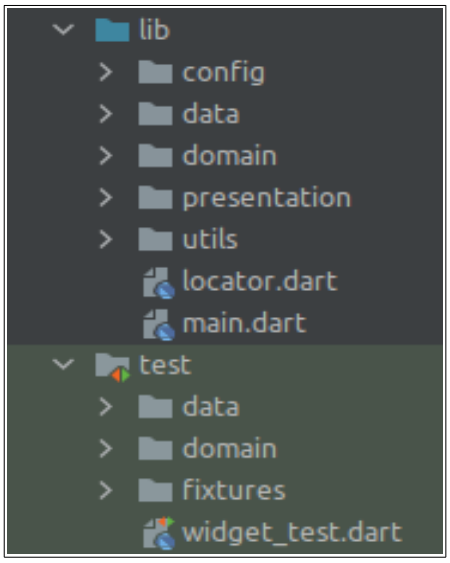
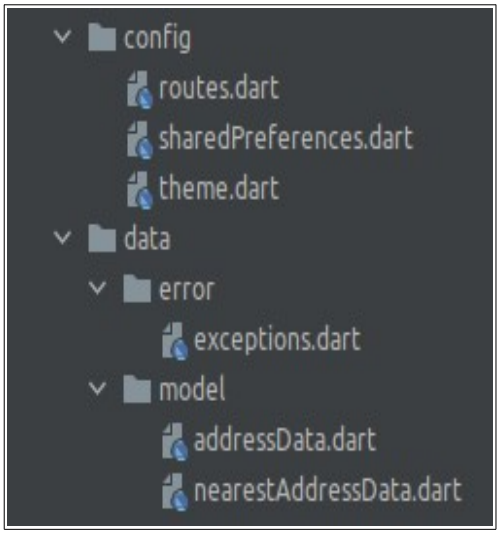
Document



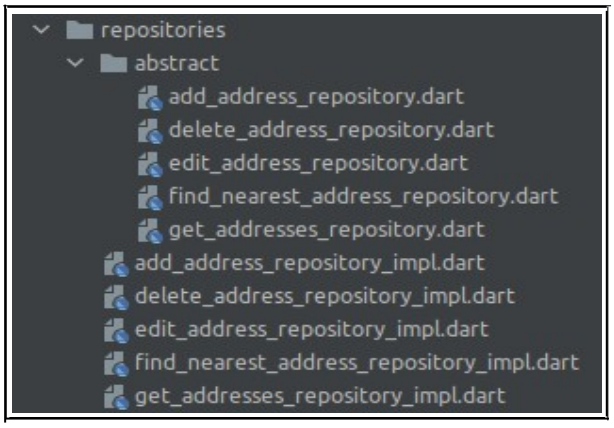
In this project I’ve separated the responsibilities and used technologies to increase de-coupling and also make the project testable and scalable.

I’ve tried to separate the presentation from business logic and the architecture of the project makes it easy to maintain the code and develop it and although it was a small sample project, its structure really ready to be extended.



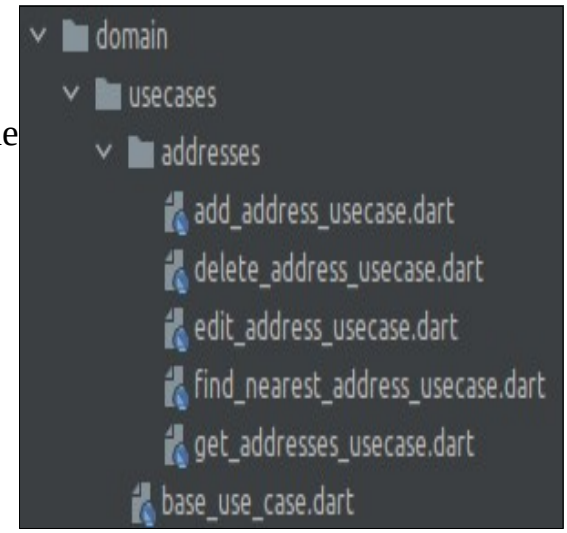
The configuration of the project includes colors and provides the routes of project parts and also implements the communication by sharedPreferences.

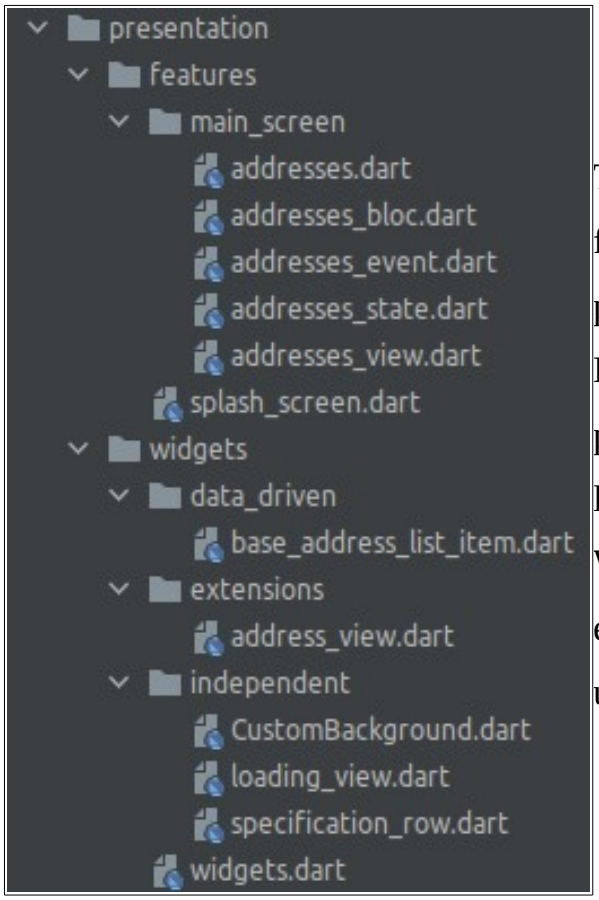
Data folder includes exception info and the model of data that is used in the business logic layer.
The data structure is defined in such a way that it can accept data in Json format.



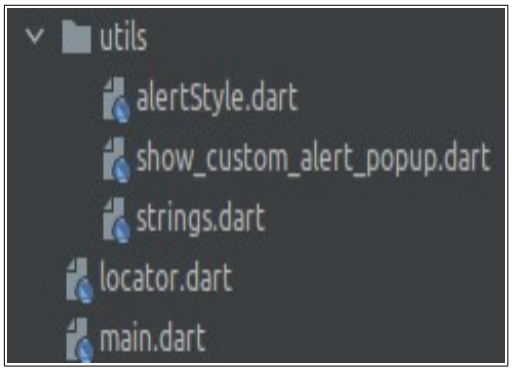
The other part of data folder is repositories that is used to abstract and implement the important part of the project to access models.

Domain Folder contains all usecases witch play the key role within the project and prepare inputs and manage the results from function calls (including database, API and local services).



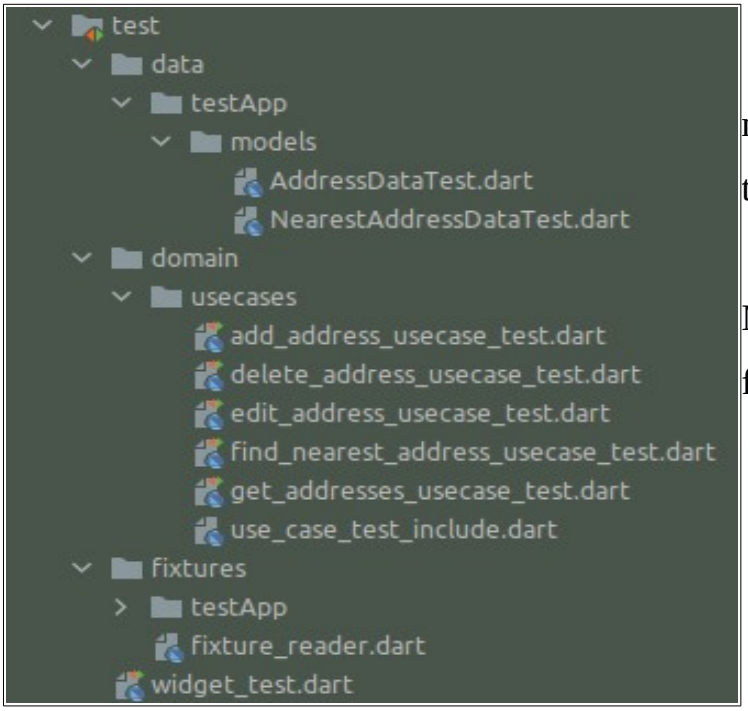


The main parts of UI are placed in Presentation folder that includes views and BloC related processes and states management.
Each screen of this project has a folder in presentation and affected regularly by the lifecycle that is generated by BloC features.
Widgets folder contains the data driven, extensions and independent widgets that are used in different parts of the project.



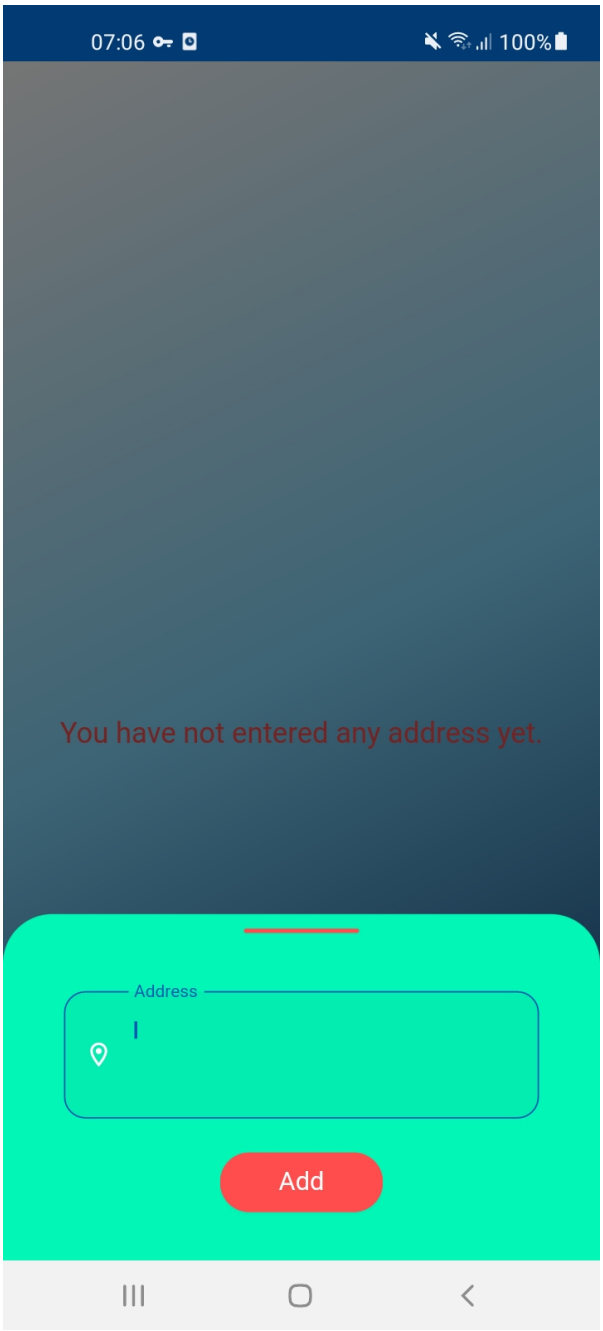
Utils generally constants which are using during scaffolds, widgets and functions through the project.

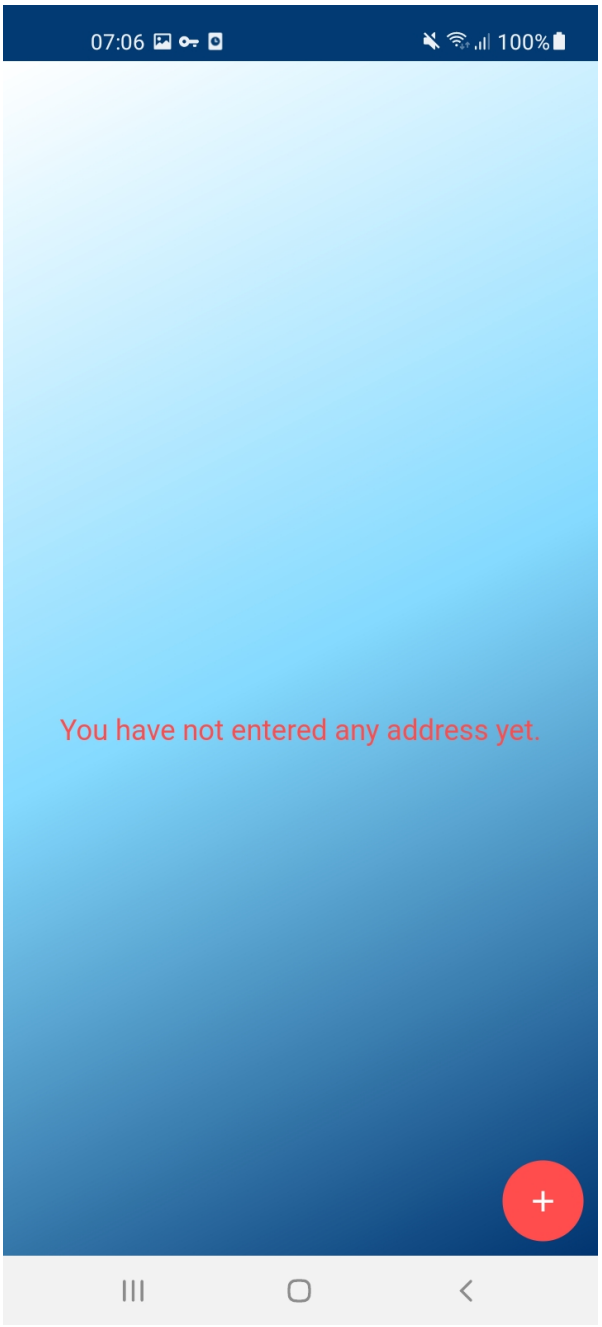
I’ve used GetIt as a service locator.
I’m using it because as our App grows, we will need to put the app's logic in classes that are separated from Widgets. Keeping our widgets from having direct dependencies makes the code better organized and easier to test and maintain.

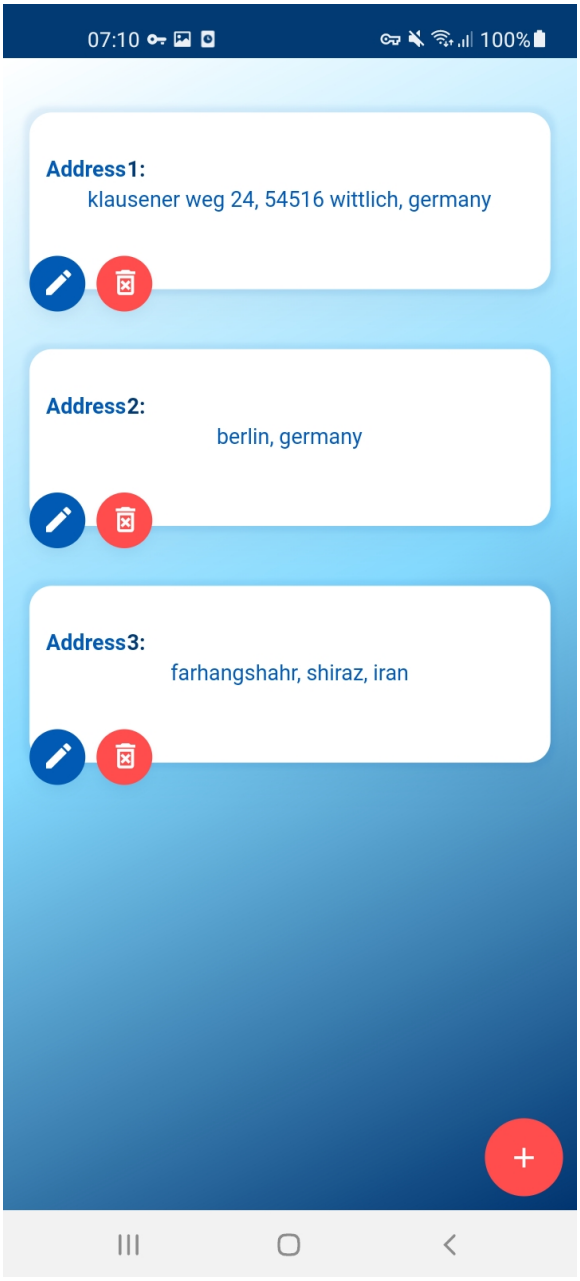


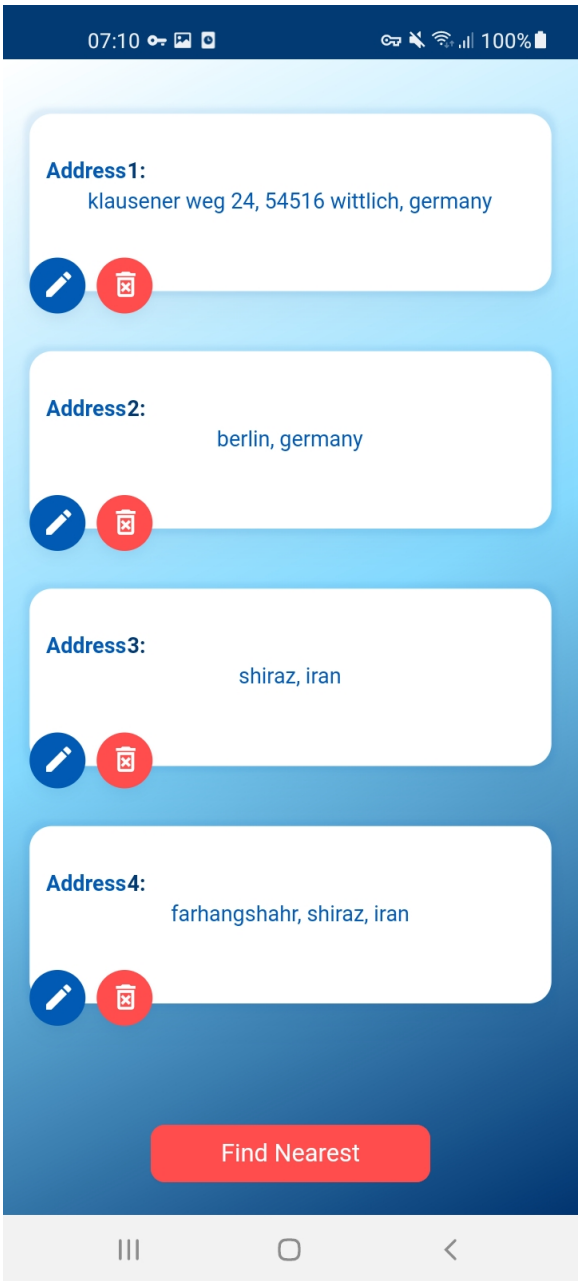
And finally I wrote Unit Tests on models and usecases which are used in the project.
To do this I employed Flutter_test, Mockito and some mocked data as the fixture.

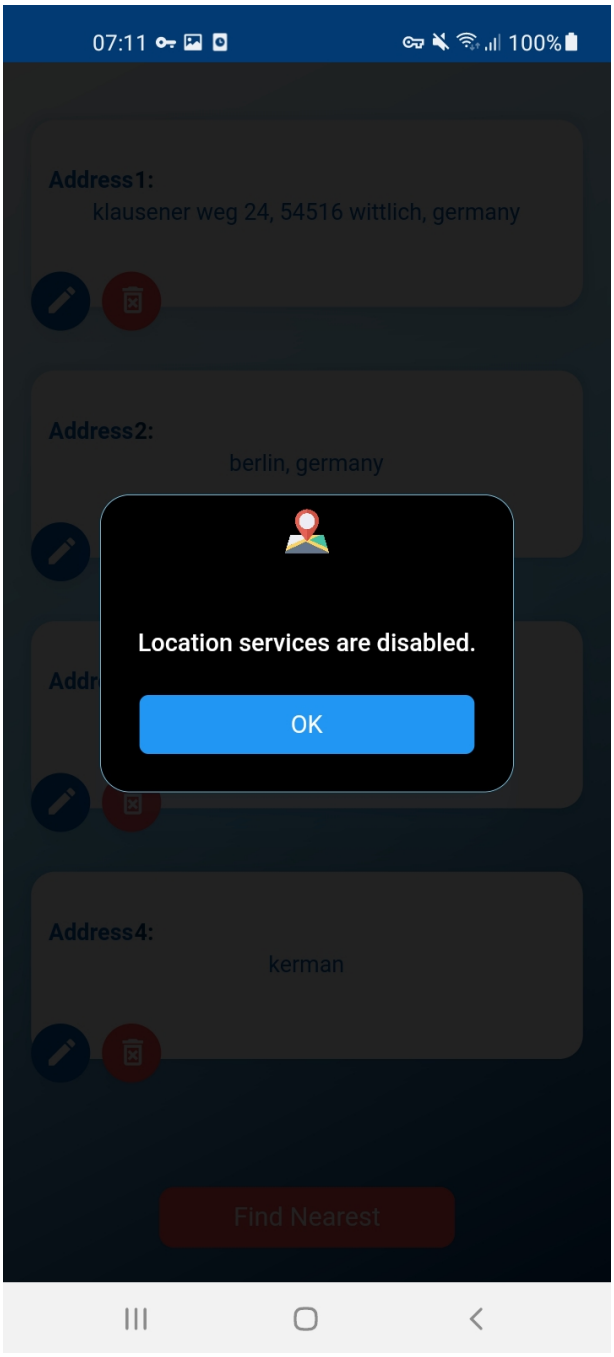
Below you can see some pages of the application:

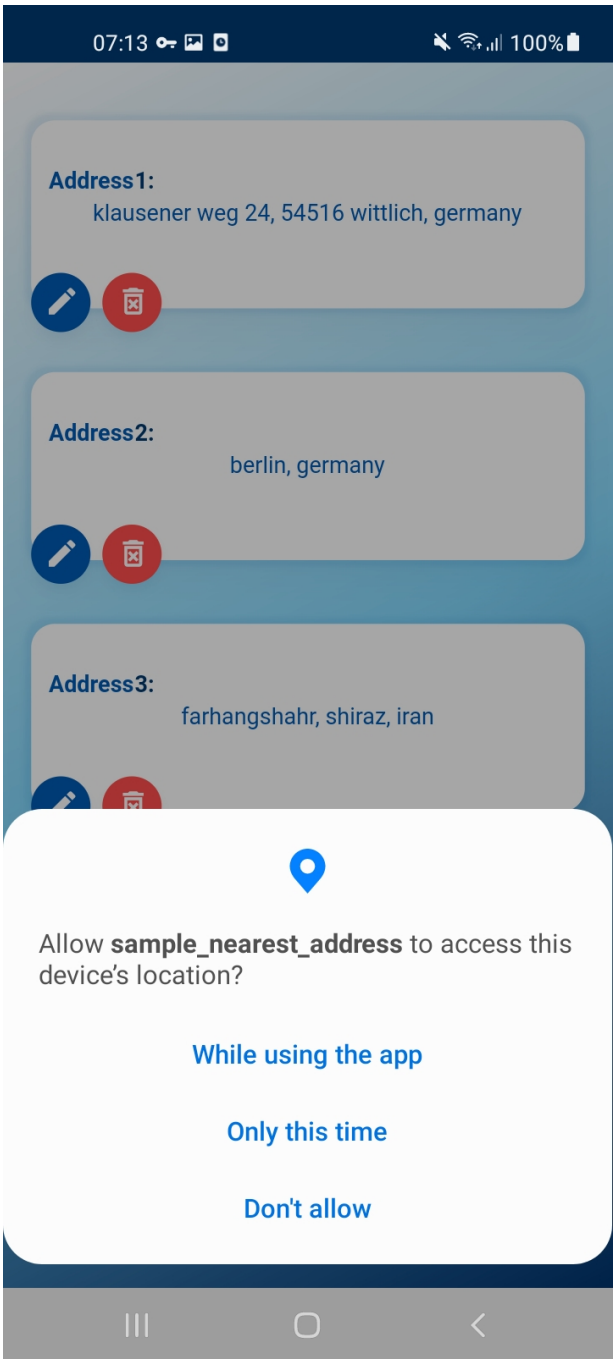


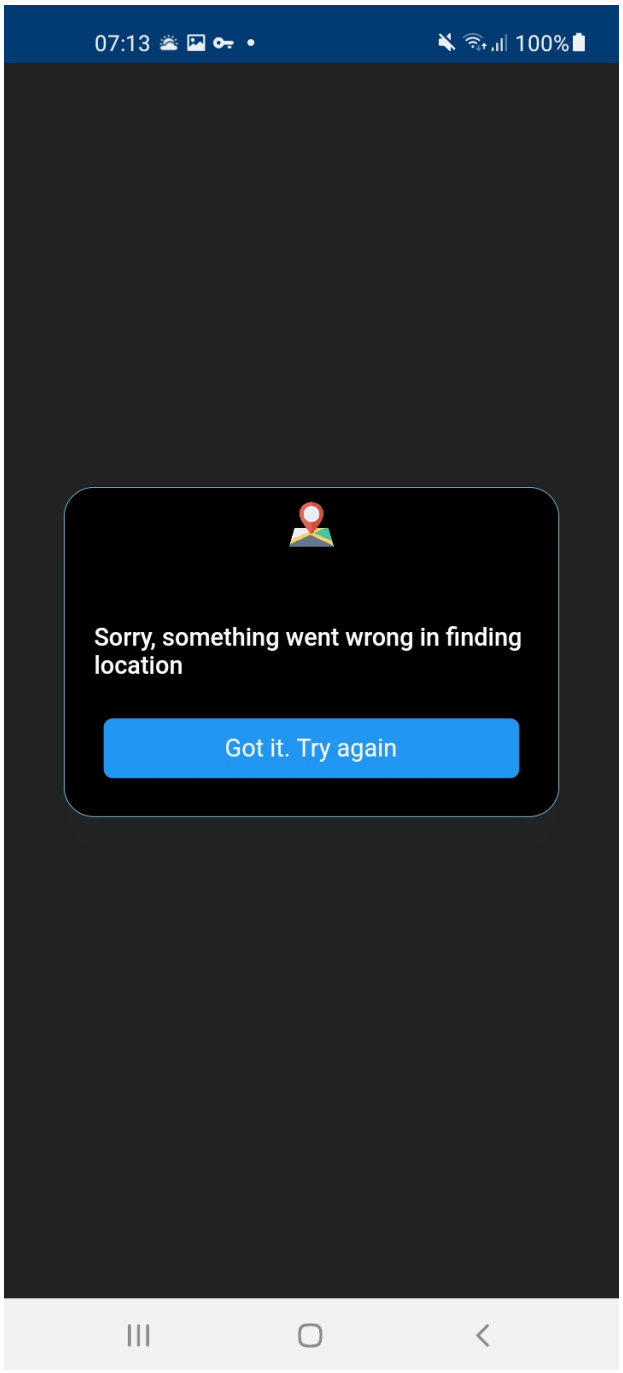


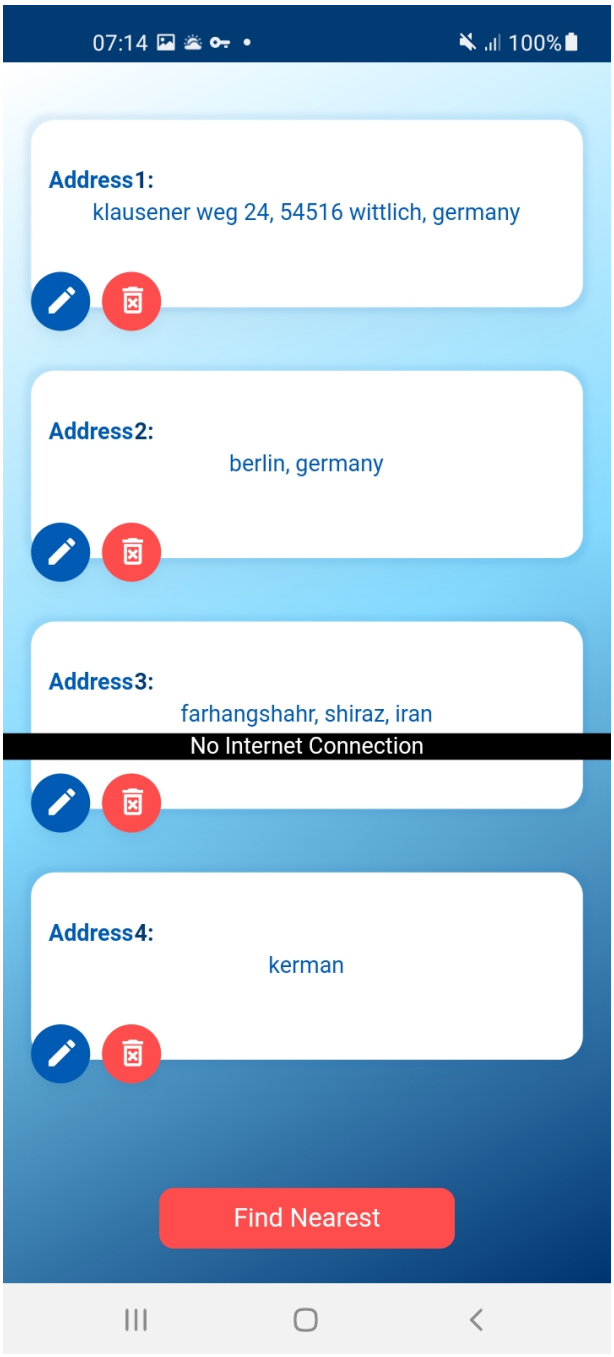

07:10
100%
Address1:
klausener weg 24, 54516 wittlich, germany
Address2:
berlin, germany
Address3:
farhangshahr, shiraz, iran


07:10
100%
Address1:
klausener weg 24, 54516 wittlich, germany
Address2:
berlin, germany
Address3:
shiraz, iran
Address4:
farhangshahr, shiraz, iran
Find Nearest


07:11
100%
Address1:
klausener weg 24, 54516 wittlich, germany
Address2:
berlin, germany
Location services are disabled.
OK
Address4:
kerman
Find Nearest


07:13
100%
Address1:
klausener weg 24, 54516 wittlich, germany
Address2:
berlin, germany
Address3:
farhangshahr, shiraz, iran
Allow sample_nearest_address to access this device's location?
While using the app
Only this time
Don't allow


07:13
100%
Sorry, something went wrong in finding location
Got it. Try again


07:14
100%
Address1:
klausener weg 24, 54516 wittlich, germany
Address2:
berlin, germany
Address3:
farhangshahr, shiraz, iran
No Internet Connection
Address4:
kerman
Find Nearest