Requirements

REQ. 1. User Interface

For this calculator, you will be in charge of the user interface and user experience. You can decide how to adapt the interface to the rest of the requirements.

The interface should work on all iPhone devices and iPads. You should consider how orientation and device size affects the UI and decide if changes on the UX/UI must be done.

REQ. 2. Basic operations

The calculator must have the following operations, which does **not** require any online call.

Symbol	Operation	Example
+	addition	`2 + 2 = 4`
-	subtraction	`5 - 7 = -2`
*	multiplication	`6 * 2 = 12`
/	division	`20 / 5 = 4`
sin	sin (degrees)	`sin(90) = 1`
cos	cos (degrees)	`cos(360) = 1`

REQ. 3. Online operations

The calculator must have the following operation, which **requires** online calls to a backend of your choice:

Symbol	Operation	Example
₿	Given a bitcoin value, obtain the current dollar value	`₿(1) = 56191,21`

REQ. 4. Feature toggling

We would like to enable or disable features for this calculator. For instance, if an operation such as addition [+] is not needed, we should be able to disable it.
The code should be well organized to support this feature. As well, the UI should automatically distribute its components based on feature toggling.

REQ. 5. Error feedback

The user should be able to receive friendly feedback for errors (for example, if an online operator fails due to connectivity).

You can take the UX/UI decision about how you would like to give this feedback.

REQ. 6. Concatenate operations

The user should be able to concatenate operations. You can add a clear button for this purpose. For instance, you can do:

- Type 2
- Type +
- Type 3
- Result = 5
- Type *
- Type 4
- Result = 20

Non-Functional Requirements

NREQ. 1. Color Schemes

The app should support two kinds of color themes. If we wish to change the colors of the app and its components, we should be able to do it by switching a single feature.

NREQ. 2. Modularization

Distribute your app in modules that could be easily separated in the future on libraries.

NREQ. 3. Error Handling

Decide a strategy for Error Handling in your application to report problems. Those should be easy to understand and track. Remember that when an app is released, developers need to know what is going on and how problems affect the final users.

NREQ. 4. README

Add a README file to explain the important information about a project. Which should serve as a helpful resource for developers who may be new to the project, and can provide important context for future development work.