add_2_nums.py

```
from data_modules.object_handler import keypad_state_manager

def add_2_nums():
    form.input_list={"inp_0": " ", "inp_1": " "}
    form.form_list = ["addition", "1st num", "inp_0", "2nd num", "inp_1"]
    form.update()
    display.clear_display()
    form_refresh.refresh()

    while True:
        inp = typer.start_typing()
        if inp == "back":
            app.set_app_name("installed_apps")
            app.set_group_name("root")
            break

        elif inp == "ok":
            try:
                num1=form.inp_list()["inp_0"]
                num2=form.inp_list()["inp_1"]
                nums_sum=" = "+str(float(num1)+float(num2))
                form.form_list.append(nums_sum)
                form.update()
                form.update_buffer("nav_u")
                form_refresh.refresh()
            except:
                form.form_list.append(" = error")
                form.update()
                form.update_buffer("nav_u")
                form_refresh.refresh()

        elif inp == "alpha" or inp == "beta":
            keypad_state_manager(x=inp)
            form.update_buffer("")

        elif inp not in ["alpha", "beta", "ok"]:
            form.update_buffer(inp)
        form_refresh.refresh(state=nav.current_state())
```

```
from data_modules.object_handler import keypad_state_manager
```

Importing keypad_state_manager, its responsible for changing the state of the keypad into alpha and beta mode

```
def add_2_nums():
    form.input_list={"inp_0": " ", "inp_1": " "}
    form.form_list = ["addition", "1st num", "inp_0", "2nd num", "inp_1"]
    form.update()
    display.clear_display()
    form_refresh.refresh()
```

Defining the app main function name add_2_nums

Initializing form buffer,

Form.input_list - this is a dictionary which is used to store data provided by the user in the input field in a form
The keys of the dictionary need to be in the format of “inp_0”, “inp_1”, “inp_2”, …
The values need to be a space or some other character/string

Form.form_list - this is a list where you define the labels and the input fields which are going to be shown on the display in the exact same order as you mention here.

If any item in **form.form_list** is “inp_0”, “inp_1”, … then it is considered as an input field and an input field appears on the display.

If the item is not an input field, then it is considered as a label, which is shown only as a text item on the screen.

form.update() - after you change the input_list and form_list you need to update the form which resets every other parameter like your cursor position in the screen, etc.

display.clear_display() - it clears the display

form_refresh.refresh() - it projects all the data in the form to the display and your app becomes visible.

```
    while True:
        inp = typer.start_typing()
```

while True: - The app loop, it will keep the app running forever until you exit from the app
typer.star_typing() - start listening to keyboard events and returns the key pressed

```
        if inp == "back":
            app.set_app_name("installed_apps")
            app.set_group_name("root")
            break
```

If inp == “back”: - the event handler started, this conditional statement will execute if the user presses back button in the keypad

app.set_app_name(“installed_apps”) - setting the name of the app which needs to be open next

app.set_group_name(“root”) - setting the group name in which the app is present which needs to be open next

Break - you exit from this app and the app which you set in the above 2 lines will run next

```
        elif inp == "ok":
            try:
                num1=form.inp_list()["inp_0"]
                num2=form.inp_list()["inp_1"]
                nums_sum=" = "+str(float(num1)+float(num2))
                form.form_list.append(nums_sum)
                form.update()
                form.update_buffer("nav_u")
                form_refresh.refresh()
            except:
                form.form_list.append(" = error")
                form.update()
                form.update_buffer("nav_u")
                form_refresh.refresh()
```

This action is performed when the user hits ok in the keypad
form.inp_list() - this function gives out the entire dictionary of the inputs which are being given be the user in the input field.

form.form_list.append(nums_sum) - inserting nums_sum item in the end of the form_list
form.update_buffer(“nav_u”) - this function is used to give commands to the form and “nav_u” means giving command of pressing up button in the keypad to the form app
Before giving this command the **form.update()** function was called which resets the cursor and all other parameters.
So after doing “nav_u” from the top line the user will reach the last line of the form app where we just appended the result.
Then **form_refresh.refresh()** which shown the all the stuff on the display

If some error comes here then the **except** block will be executed

```
        elif inp == "alpha" or inp == "beta":
            keypad_state_manager(x=inp)
            form.update_buffer("")

        elif inp not in ["alpha", "beta", "ok"]:
            form.update_buffer(inp)
        form_refresh.refresh(state=nav.current_state())
```

This code needs to be present at the end of every form application which helps in changing the keypad state and doing all the internal navigation in the form app.