

Add Dynamic Content to your PDF Report

Okay, we discover how to add static content, what about dynamic content? Lets find out

As I previously mention you have `python/*` folder with your `.ipynb` scripts, lets create a simple python script `python/example.ipynb`

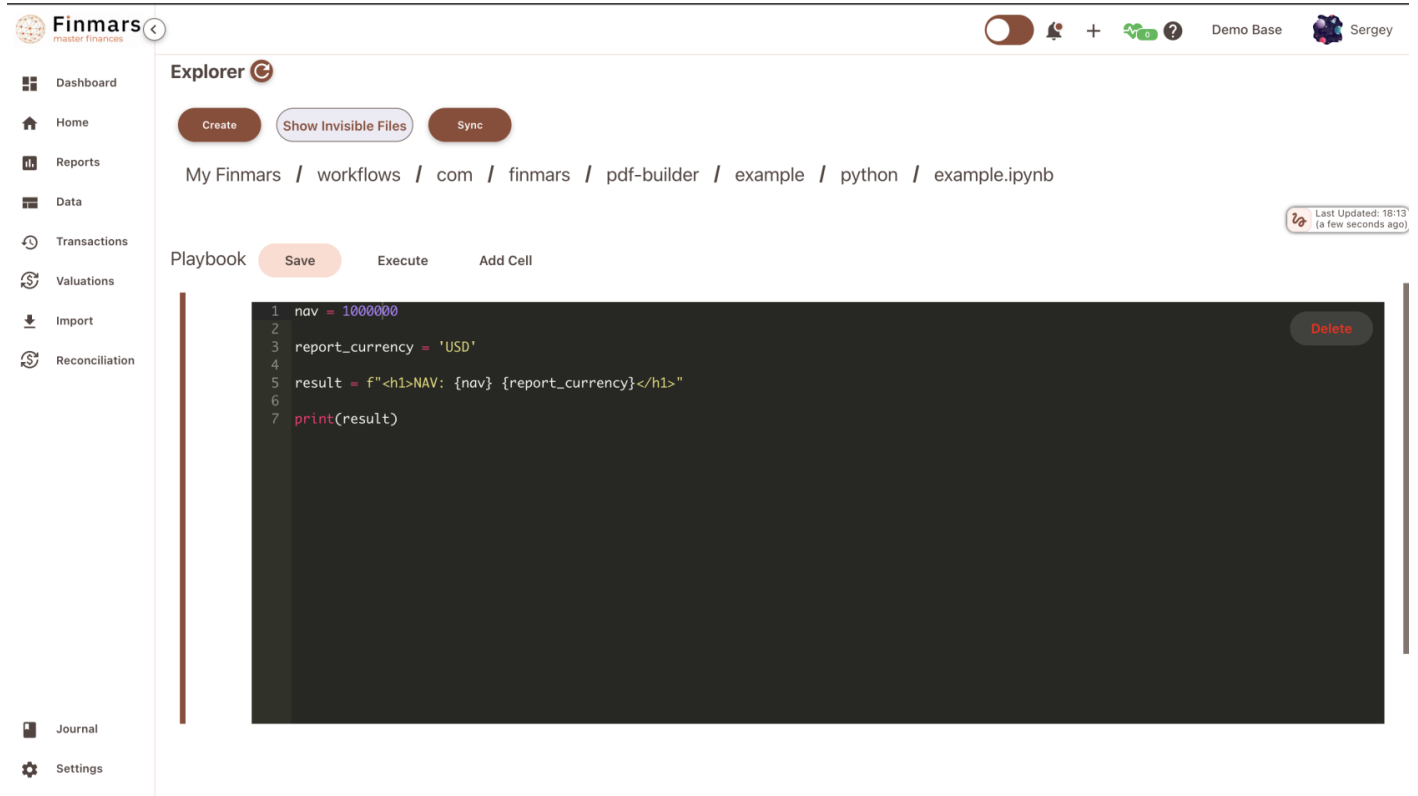
And lets output some content

```
nav = 1000000

report_currency = 'USD'

result = f"<h1>NAV: {nav} {report_currency}</h1>"

print(result)
```

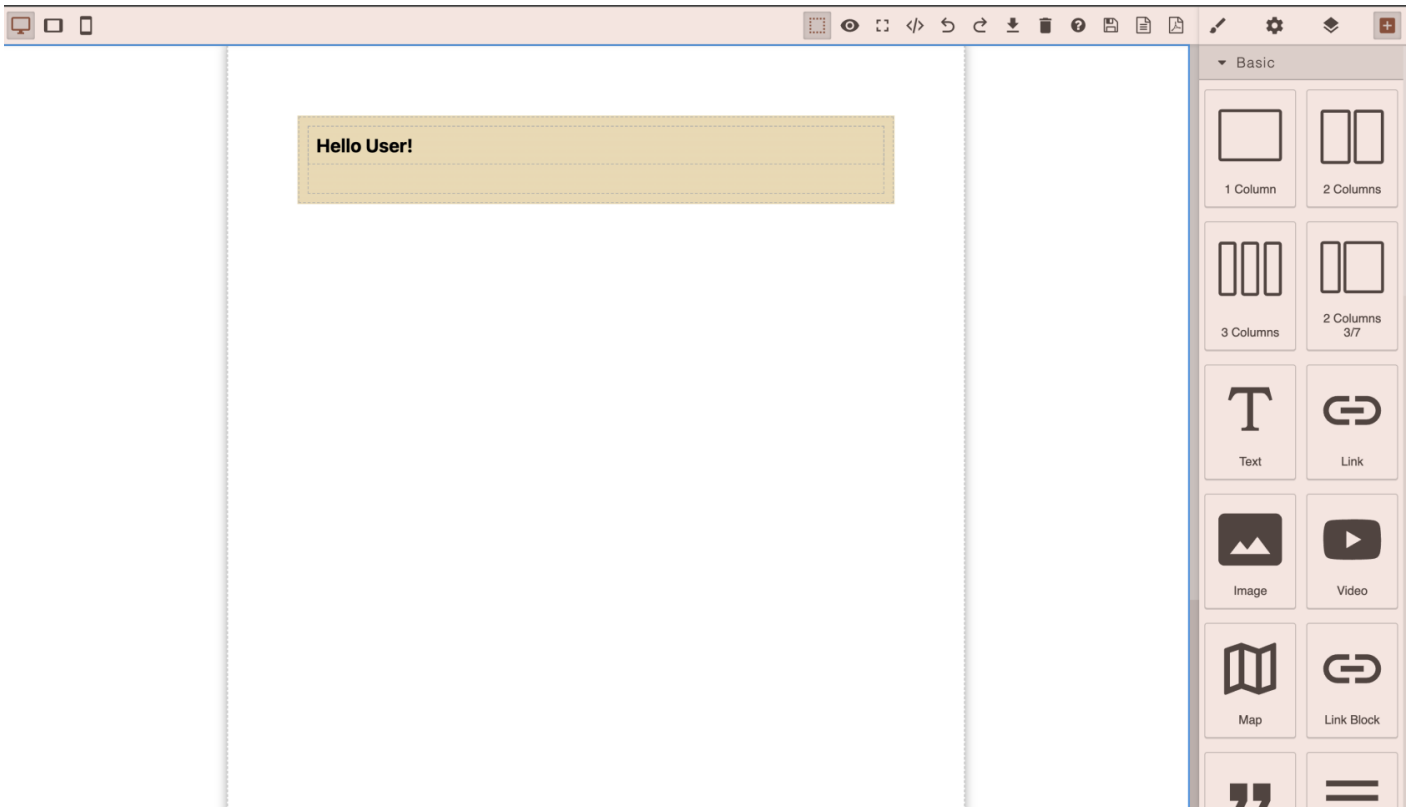


The screenshot shows the Finmars application interface. The top navigation bar includes the Finmars logo, a search icon, a toggle switch, a notification bell, a plus sign, a green checkmark, a question mark, the text "Demo Base", and a user profile icon for "Sergey". The left sidebar contains a menu with items: Dashboard, Home, Reports, Data, Transactions, Valuations, Import, Reconciliation, Journal, and Settings. The main content area is titled "Explorer" and shows a file path: "My Finmars / workflows / com / finmars / pdf-builder / example / python / example.ipynb". Below the path are buttons for "Create", "Show Invisible Files", and "Sync". The "Playbook" section has buttons for "Save", "Execute", and "Add Cell". A code editor displays the following Python code:

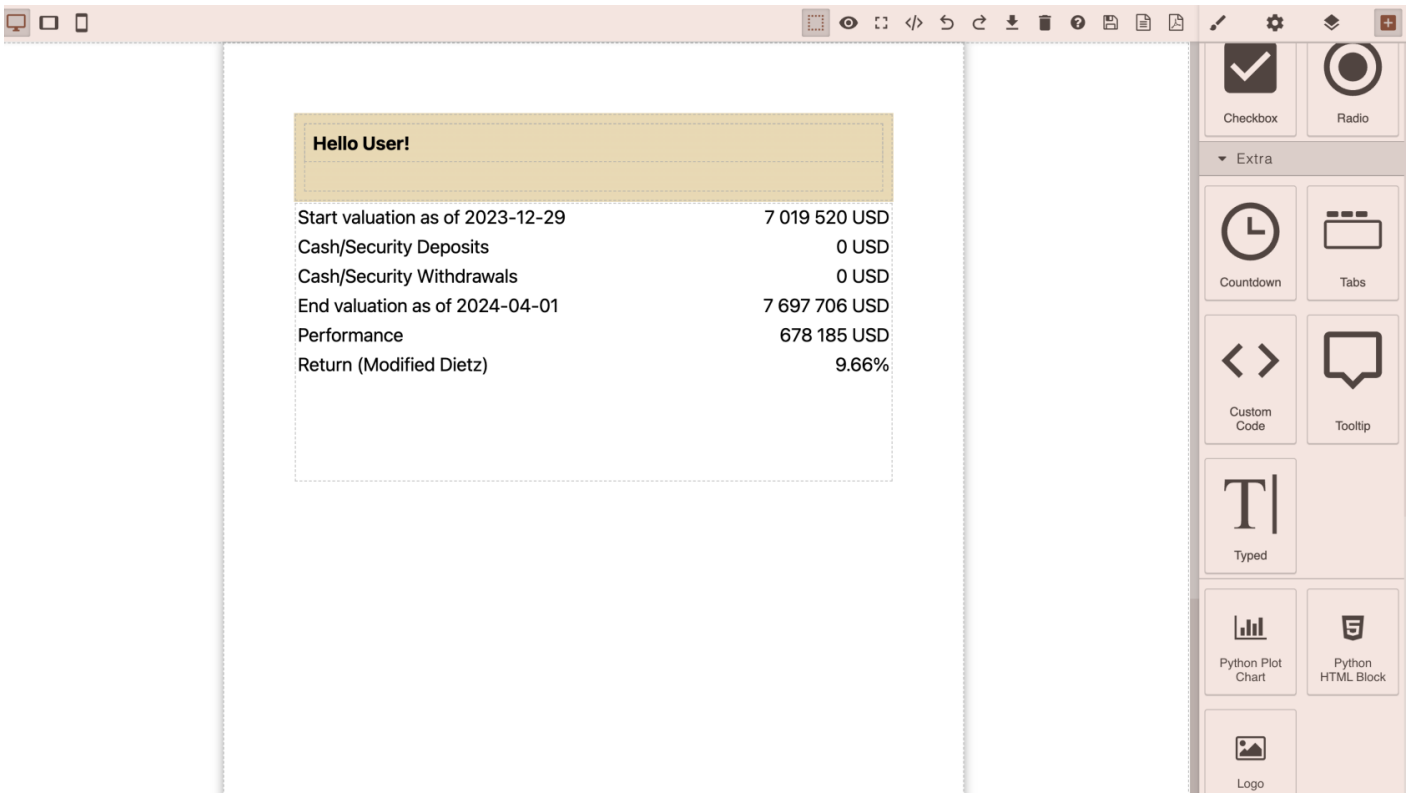
```
1 nav = 1000000
2
3 report_currency = 'USD'
4
5 result = f"<h1>NAV: {nav} {report_currency}</h1>"
6
7 print(result)
```

A "Delete" button is visible in the top right corner of the code editor. A status bar at the bottom right indicates "Last Updated: 18:13 (a few seconds ago)".

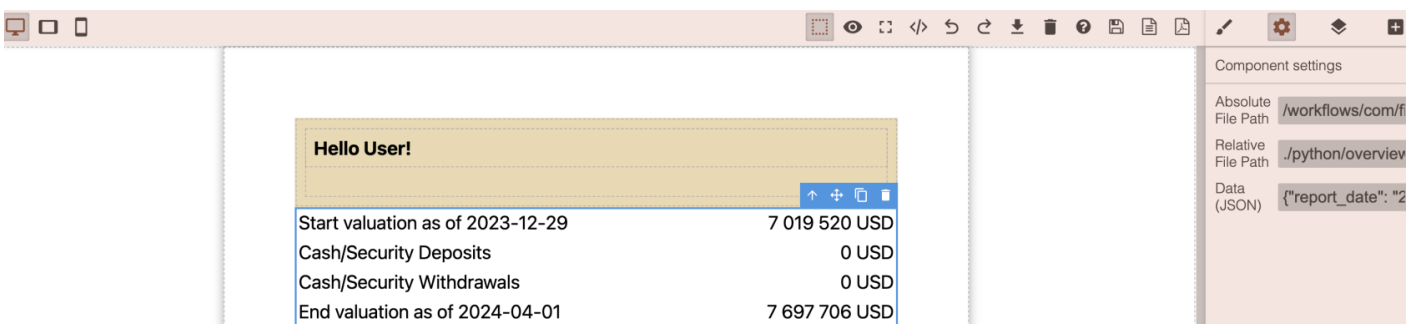
Well done, lets go back to our PDF Builder



Scroll down and find Block - **Python HTML Block**, drop it on canvas



Then click on it, and let see the **Block Settings** (On right sidebar, cog button)



So we have here 3 inputs

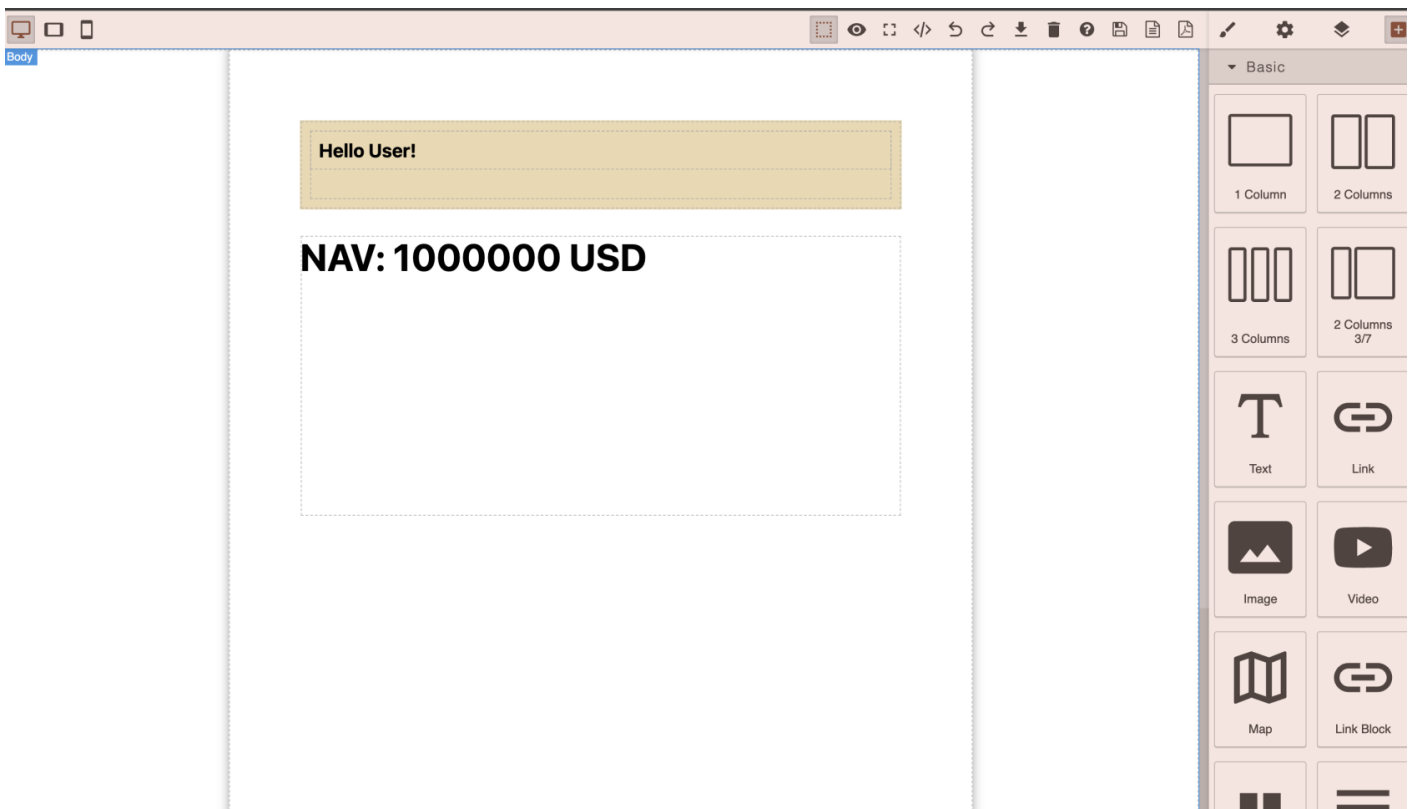
Absolute File Path - full path from root (/) of your Space Storage

Relative File Path - path from your builder location, so you can do like `./python/example.ipynb`

Data (JSON) - json field that could have some predefined settings for you `./ipynb` script

NOTA BENE: Relative File Path when set, it overrides Absolute File Path, so if you dont need relative path, just set it to null and use absolute path, otherwise, dont use Absolute File Path, when you type Relative File Path it will automatically calculated and passed to Absolute File Path

Lets put `./python/example.ipynb` and see what will happen



Great, now the result of your `python/example.ipynb` is executed and added to your PDF Template, well Done

But as you see, we just hardcoded NAV in `./ipynb` code, but we need to fetch it dynamically from Finmars REST API, lets do it!

So, we need to output some NAV, but what NAV?

Do make it all work, we need to now at least two things, **Report Date** and **Portfolio**

So, what to do? To achieve that, you can just pass it as Query Parameters, so

/example/client_report_builder.html?report_date=2024-08-10&portfolio=Bonds%20Portfolio

Simple as that, all your Blocks that you put on canvas will listen to your Query Parameters, but how to access them?

In your .ipynb script before it will be executed, Finmars fills out special variable **execution_context**

So, try it!

```
if 'execution_context' not in locals():
    execution_context = {}

portfolio = execution_context.get('portfolio', 'No Portfolio Selected')
report_date = execution_context.get('report_date', 'No Report Date is set')

print(portfolio)
print(report_date)
```

The screenshot shows the Finmars application interface. On the left is a sidebar with navigation options: Dashboard, Home, Reports, Data, Transactions, Valuations, Import, Reconciliation, Journal, and Settings. The main area is titled 'Explorer' and shows the file path: My Finmars / workflows / com / finmars / pdf-builder / example / python / example.ipynb. Below the path are buttons for 'Create', 'Show Invisible Files', and 'Sync'. The main editor area is titled 'Playbook' and contains the Python code from the previous block. A 'Delete' button is visible in the top right corner of the code editor. The top right of the interface shows a toggle switch, a notification bell, a plus sign, a green checkmark, a question mark, and the text 'Demo Base' and 'Sergey'.

And the result!

The screenshot shows a web browser window displaying the rendered output of the code. The main content area contains a yellow box with the text 'Hello User!' and 'Bonds Portfolio 2024-08-10'. The browser's developer tools are open on the right side, showing the 'Classes' panel with '- State -' and a '+' button. The 'Selected' item is 'Body #ia2s' and the 'Decorations' panel is expanded.

Great, now we can pass Parameters to our .ipynb script, to make it dynamic! Lets back to NAV

To achieve that, we must make a request to Finmars REST API, so it will calculate for us Balance Report, and we can took NAV from it.

To make it work, we need to import request_api from workflow.finmars (Yes, everything executed in Workflow Service) to achieve security Finmars REST API is isolated from Workflow User Scripts

```
import requests

from workflow.finmars import request_api

if 'execution_context' not in locals():
    execution_context = {}

report_currency = execution_context.get('Valuation Currency', 'USD')
portfolio = execution_context.get('portfolio', 'No Portfolio Selected')
report_date = execution_context.get('report_date', 'No Report Date is set')
pricing_policy = execution_context.get('Pricing Policy', 'com.finmars.standard-pricing:standard')

def get_nav():

    payload = {
        "cost_method": 1,
        "pricing_policy": pricing_policy,
        "report_date": report_date,
        "report_currency": report_currency,
        "portfolios": [portfolio],
        "frontend_request_options": {
            "columns": [],
            "filter_settings": [],
            "groups_types": [],
            "groups_values": []
        }
    }

    response = request_api(f'/api/v1/reports/backend-balance-report/items/', method='POST',
data=payload)
```

```
items = response["items"]

nav = 0

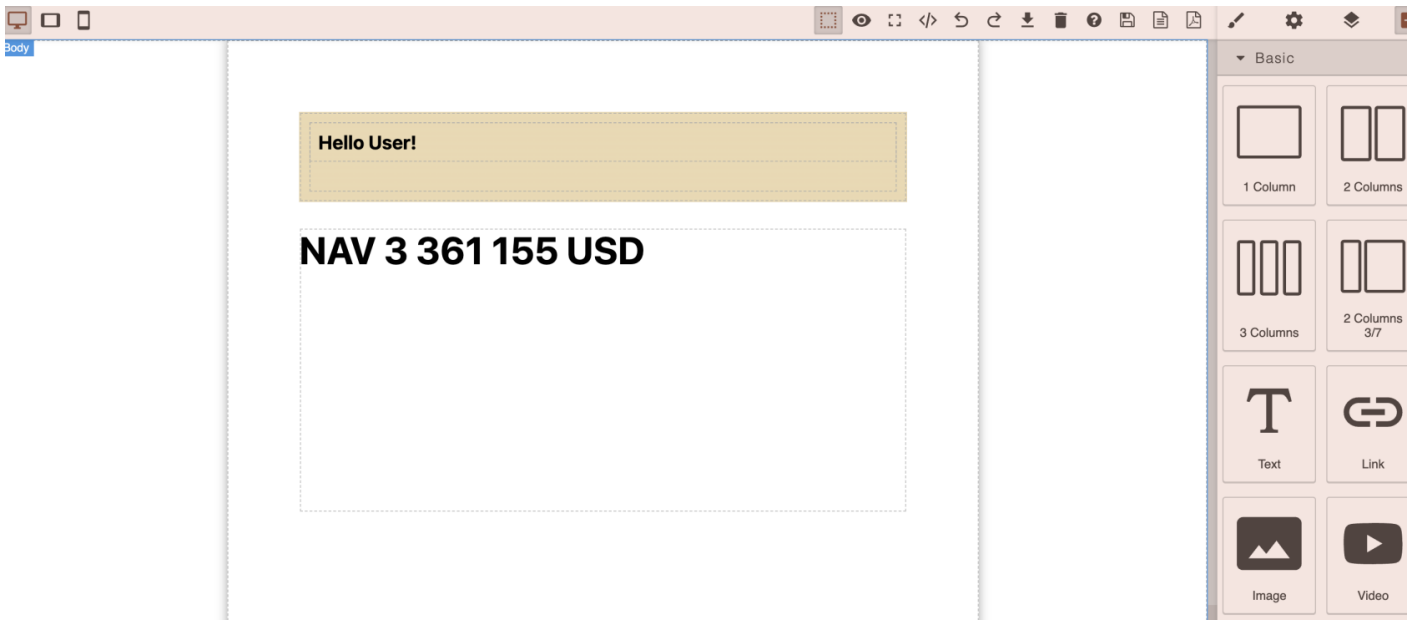
for item in items:
    if item['market_value']:
        nav = nav + item['market_value']

return f"{int(nav):,}".replace(',', ' ')

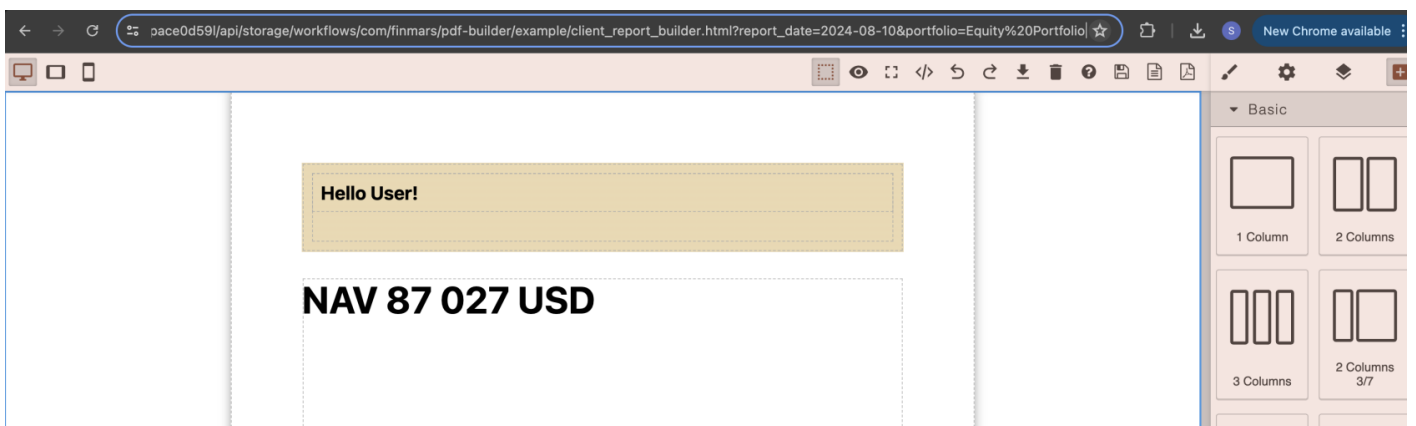
nav = get_nav()

print(f"<h1>NAV {nav} {report_currency}</h1>")
```

Okay, we made a request, we have list of positions and we just sum up all the market_values to get our NAV. And here is result!

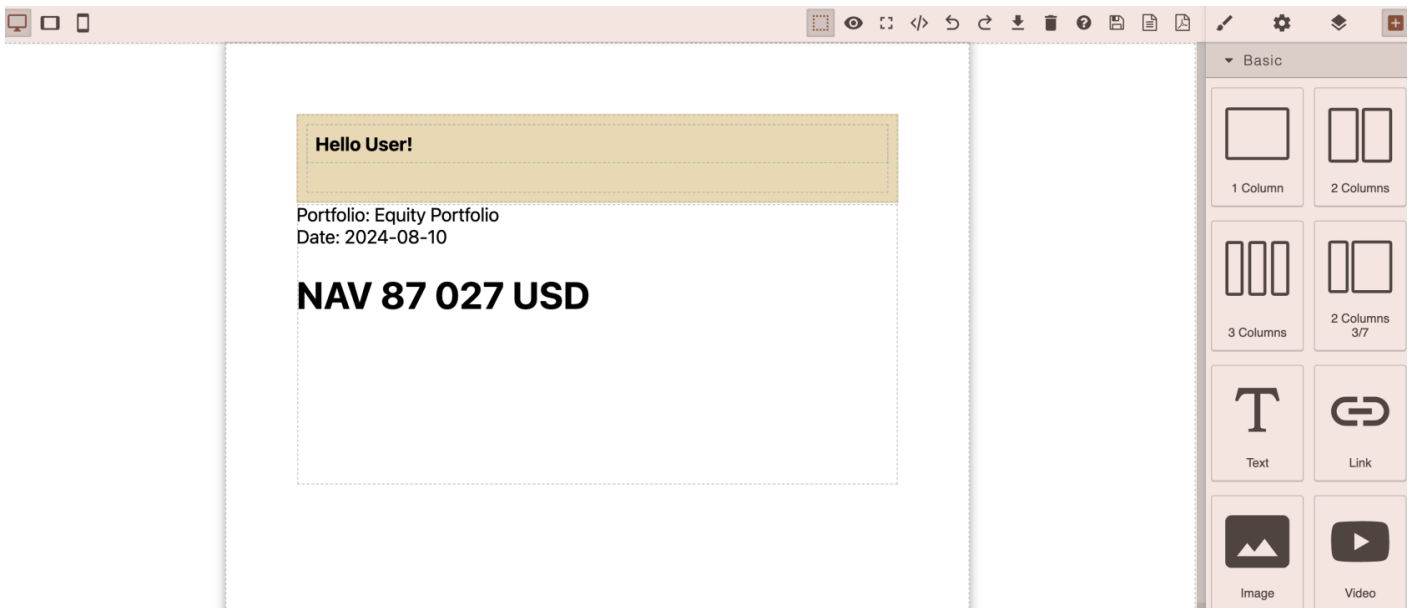


And now we can change Portfolio, for example Equity Portfolio

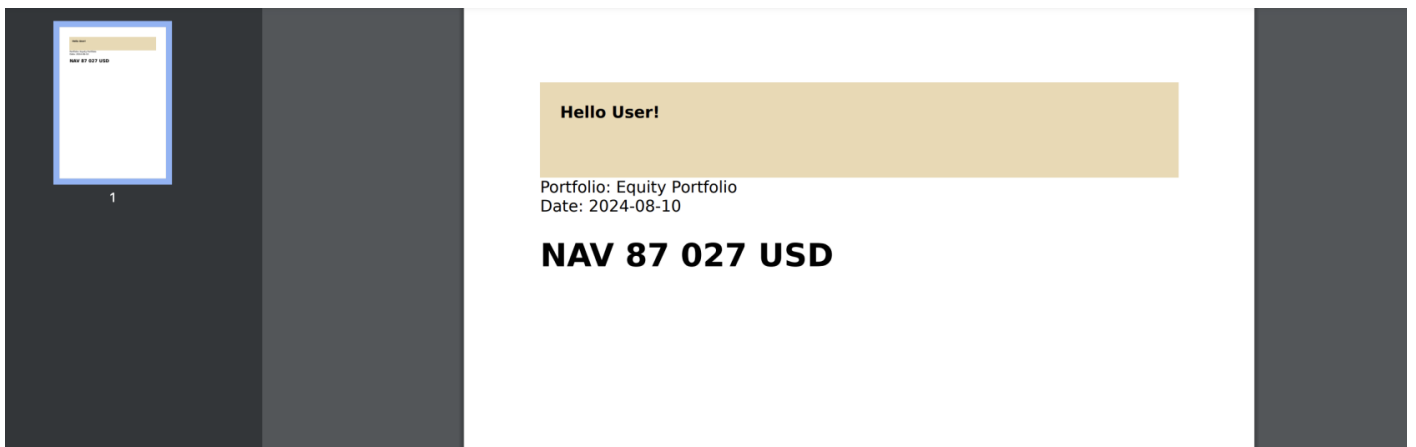


We have a dynamic PDF Template and if we export it, it will all work!

Lets Add some more text and then print



And the result!



Great work! Now lets move to next article, where we will add a Pie Chart. Go to <https://docs.finmars.com/books/pdf-builder/page/add-pie-chart-to-your-pdf-report>

P.S. But how it all work?

So each time when you work with PDF Builder, or its HTML Result, or send it to PDF Print Service each time your [access_token](#) (Keycloak) or [bearer_token](#) (Personal Access Token API) is in use. It means, generation will be done by your User and your IAM access Rights.

P.S.S In case of Workflow Schedules [finmars_bot](#) will be used as User

Revision #13

Created 2024-08-15 16:03:30 UTC by Sergei Zhitenev

Updated 2024-08-15 16:37:52 UTC by Sergei Zhitenev