Oscar Cortez
@oskrgab

1. High impact, low effort.

#Python libraries can help automate time-consuming processes quickly. Tasks that take hours to complete are good candidates to automate, and you'll see immediate rewards when using Python. Here are some of my favorite libraries for engineering:

- pandas
- polars
- simpy
- scipy
- sympy

Oscar Cortez
@oskrgab

2. Start with Jupyter Notebooks.

These are a great way to see your results and the code you wrote. You could also use the Interactive Mode in #VSCode with your .py files as an alternative (this one is my favorite since it plays nicely with version control)

2. Define the differential equation

Let's define the Laplace equation for steady-state flow in SymPy. We will use the `Eq` class to define the equation.

In [2]:

```
# Define the differential equation as shown in the image
laplace_eq = Eq(p.diff(r, r) + (1/r) * p.diff(r), 0)
laplace_eq
```

Out[2]:

$$\frac{d^2}{dr^2}p(r) + \frac{\frac{d}{dr}p(r)}{r} = 0$$

Oscar Cortez
@oskrgab

3. Read various data sources.

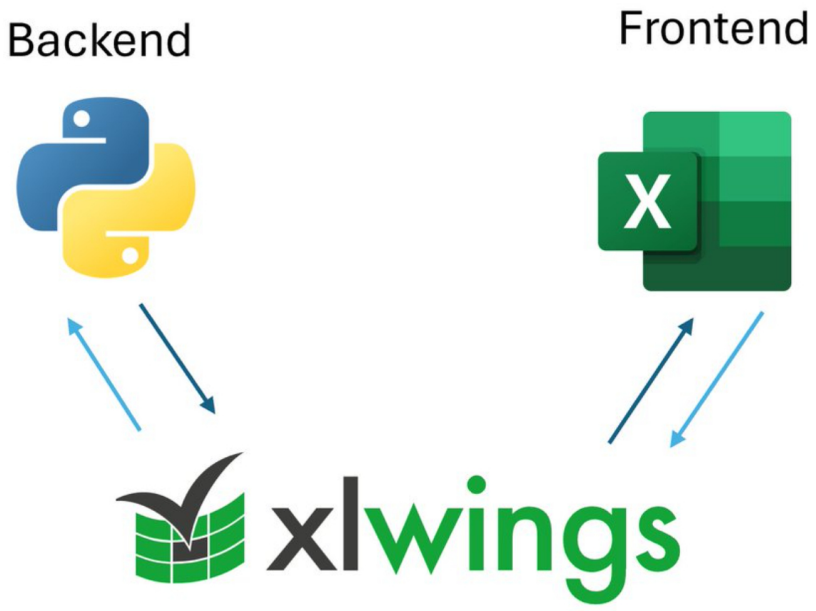
Python is widely used for data-handling tasks. Below are different data sources and the corresponding Python libraries to read them:

1. Databases - sqlite3, pyodbc, sqlalchemy
2. Text files - plain Python, pandas
3. Excel files - openpyxl, xlwings

Oscar Cortez
@oskrgab

4. Don't ditch Excel. Enhance it.

If you heavily rely on Excel, there's no need to replace it with Python completely. Instead, consider using libraries like @xlwingsorg to automate Excel processes and use it as a frontend for complex workflows.



Oscar Cortez
@oskrgab

In conclusion, automating tasks with Python scripts can significantly boost productivity, save time, and increase efficiency.

By embracing automation, you can supercharge your effectiveness and master Python to take your skills to the next level.

FOUND THIS HELPFUL? GIVE IT A LIKE AND SHARE

Oscar Cortez

@oskrgab