Member-only story

Python Clean Code: 6 Best Practices to Make Your Python Functions More Readable

Stop writing Python functions that take more than three minutes to understand

Khuyen Tran · Follow

Published in Towards Data Science
6 min read · Jan 20, 2021

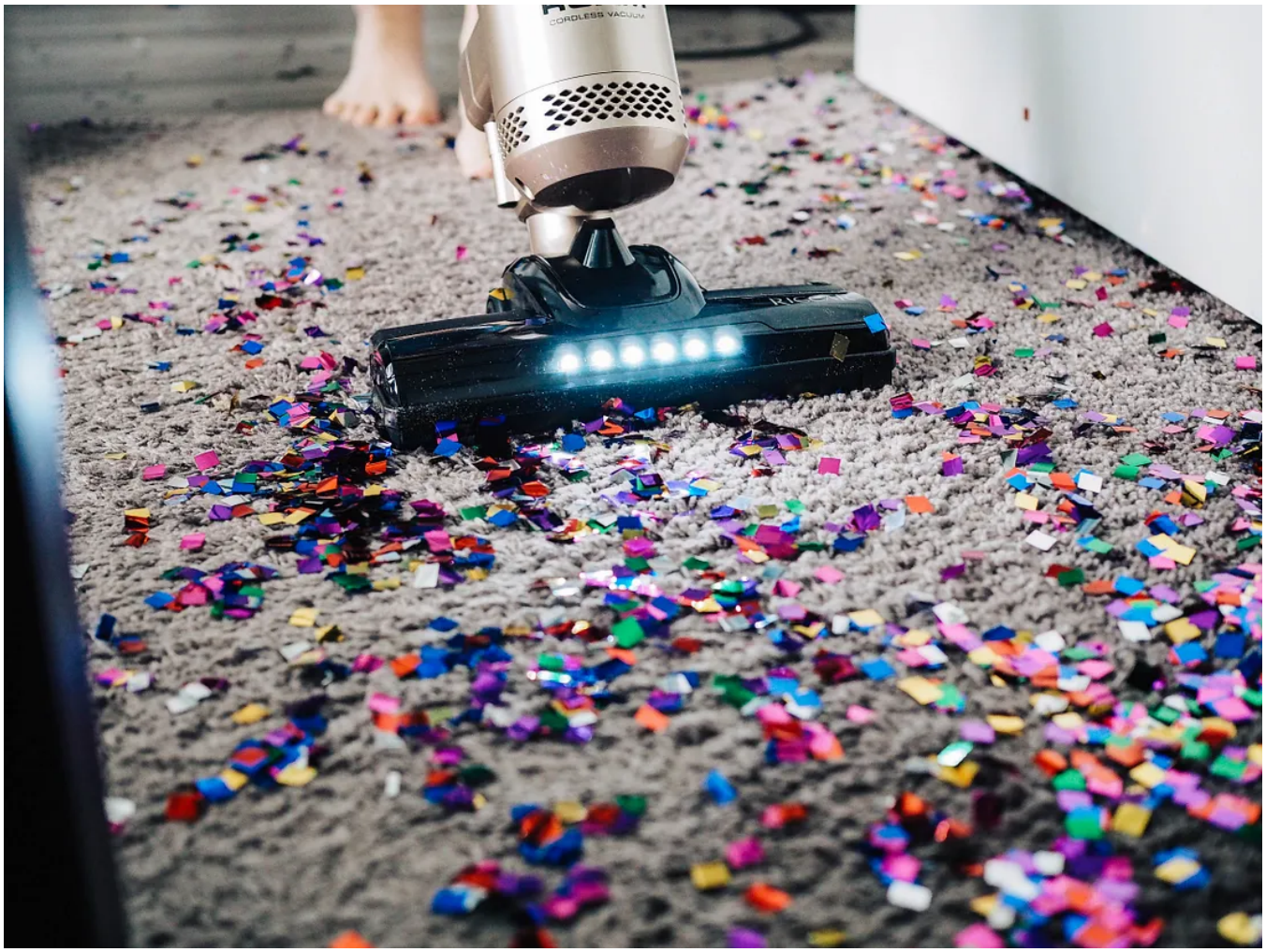


Photo by The Creative Exchange on Unsplash

> *The article has been recently updated. To access the latest version, please follow this link.*

Motivation

Have you ever looked at a function you wrote one month earlier and found it difficult to understand in 3 minutes? If that is the case, it is time to refactor your code. If it takes you more than 3 minutes to understand your code, imagine how long it would take for your teammates to understand your code.

If you want your code to be reusable, you want it to be readable. Writing clean code is especially important to data scientists who collaborate with other team members in different roles.

You want your Python function to:

- be small
- do one thing
- contain code with the same level of abstraction
- have fewer than 4 arguments
- have no duplication
- use descriptive names

These practices will make your functions more readable and easier to detect errors.

Inspired by the book *Clean Code: A Handbook of Agile Software Craftsmanship* by Robert C. Martin with code examples written in Java, I decided to write an article on how to write clean code in Python for data scientists.

In this article, I will show you how to utilize the 6 practices mentioned above to write better Python functions.

Get Started

Let's start by taking a look at the function `load_data` below.

```python
import xml.etree.ElementTree as ET
import zipfile
from os import listdir
from os.path import isfile, join

import gdown


def main():

    load_data(
        url="https://drive.google.com/uc?id=1jI1cmxqnwsmC-vbl8dNY6b4aNBtBbKy3",
        output="Twitter.zip",
        path_train="Data/train/en",
        path_test="Data/test/en",
    )


def load_data(url: str, output: str, path_train: str, path_test: str):

    # Download data from Google Drive
    output = "Twitter.zip"
    gdown.download(url, output, quiet=False)

    # Unzip data
    with zipfile.ZipFile(output, "r") as zip_ref:
        zip_ref.extractall(".")

    # Get train, test data files
    tweets_train_files = [
        file
        for file in listdir(path_train)
        if isfile(join(path_train, file)) and file != "truth.txt"
    ]
    tweets_test_files = [
        file
        for file in listdir(path_test)
        if isfile(join(path_test, file)) and file != "truth.txt"
    ]

    # Extract texts from each file
    t_train = []
```

```
    for file in tweets_train_files:
        train_doc_1 = [r.text for r in ET.parse(join(path_train, file)).getroot()[0]]
        t_train.append(" ".join(t for t in train_doc_1))

    t_test = []
    for file in tweets_test_files:
        test_doc_1 = [r.text for r in ET.parse(join(path_test, file)).getroot()[0]]
        t_test.append(" ".join(t for t in test_doc_1))

    return t_train, t_test


if __name__ == "__main__":
    main()
```

load_data_example.py hosted with ❤ by GitHub view raw

The function `load_data` tries to download data from Google Drive and extract the data. Even though there are many comments in this function, it is difficult to understand what this function does in 3 minutes. It is because:

- The function is awfully long
- The function tries to do multiple things
- The code within the function is at multiple levels of abstractions.
- The function has more than 3 arguments
- There are multiple duplications
- Function's name is not descriptive

We will refactor this code by using the 6 practices mentioned above

Small

A function should be small because it is easier to know what the function does. How small is small? There should rarely be more than 20 lines of code in one function. It can be as small as below. The indent level of a function should not be greater than one or two.

```python
import zipfile

def unzip_data(output: str):

    with zipfile.ZipFile(output, 'r') as zip_ref:
        zip_ref.extractall('.')
```

load_data_example.py hosted with ❤ by GitHub view raw

Do One Task

A function should complete only one task, not multiple tasks. The function `load_data` tries to do multiple tasks such as download the data, unzip the data, get names of files that contain train and test data, and extract texts from each file.

Thus, it should be split into multiple functions like below

```python
download_zip_data_from_google_drive(url, output_path)

unzip_data(output_path)

tweet_train, tweet_test = get_train_test_docs(path_train, path_test)
```

load_data_example.py hosted with ❤ by GitHub view raw

And each function should do only one thing:

```python
import gdown

def download_zip_data_from_google_drive(url: str, output_path: str):

    gdown.download(url, output_path, quiet=False)
```

load_data_example.py hosted with ❤ by GitHub view raw

The function `download_zip_data_from_google_drive` only downloads a zip file from Google Drive and does nothing else.

One Level of Abstraction

The code within the function `extract_texts_from_multiple_files` is at a different level of abstraction from the function.

> *The level of abstraction is the amount of complexity by which a system is viewed or programmed. The higher the level, the less detail. The lower the level, the more detail. — PCMag*

That is why:

- The function `extract_texts_from_multiple_files` is at a high-level of abstraction.
- The code `list_of_text_in_one_file =[r.text for r in ET.parse(join(path_to_file, file_name)).getroot()[0]]` is at a low-level of abstraction.

To make the code within the function to be at the same level of abstraction, we can put the low-level code into another function.

Now, the code `extract_texts_from_each_file(path_to_file, file)` is at a high-level of abstraction, which is the same level of abstraction as the function `extract_texts_from_multiple_files`.

Duplication

There is duplication in the code below. The part of code that is used to get the training data is very similar to the part of code that is used to get the test data.

We should avoid duplication because:

- It is redundant
- If we make a change to one piece of code, we need to remember to make the same change to another piece of code. If we forget to do so, we will introduce bugs into our code.

We can eliminate duplication by putting the duplicated code into a function.

Since the code to extract texts from training files and the code to extract texts from test files are similar, we put the repeated code into the function `extract_tests_from_multiple_files`. This function can extract texts from either training or test files.

Descriptive Names

> A long descriptive name is better than a short enigmatic name. A long descriptive name is better than a long descriptive comment. — Clean Code by Robert C. Martin

Users can understand what the function `extract_texts_from_multiple_files` does by looking at its name.

Don't be afraid to write long names. **It is better to write long names rather than write vague names.** If you try to shorten your code by writing something like `get_texts`, it would be difficult for others to understand exactly what this function does without looking at the source code.

If the descriptive name of a function is too long such as `download_file_from_Google_drive_and_extract_text_from_that_file`. It is a good sign that your function is doing multiple things and you should split it into smaller functions.

Have Fewer than 4 Arguments

A function should not have more than 3 arguments since it is a sign that the function is performing multiple tasks. It is also difficult to test a function with more than 3 different combinations of variables.

For example, the function `load_data` has 4 arguments: `url`, `output_path`, `path_train`, and `path_test`. So we might guess that it tries to do multiple things at once:

- Use `url` to download data
- Save it at `output_path`
- Extract the train and test files in `output_path` and save it to `path_train`, `path_test`

If a function has more than 3 arguments, consider turning it into a class.

For example, we could split `load_data` into 3 different functions:

Since the functions `download_zip_data_from_google_drive` , `unzip_data` , and `get_train_test_docs` are all trying to achieve one goal: get data, we could put them into one class called `DataGetter` .

Python Programming Data Science Code Function

Since these methods do not use any class attributes or class methods. Find more about these methods here.

Follow

Written by Khuyen Tran

414K Followers · Writer for Towards Data Science

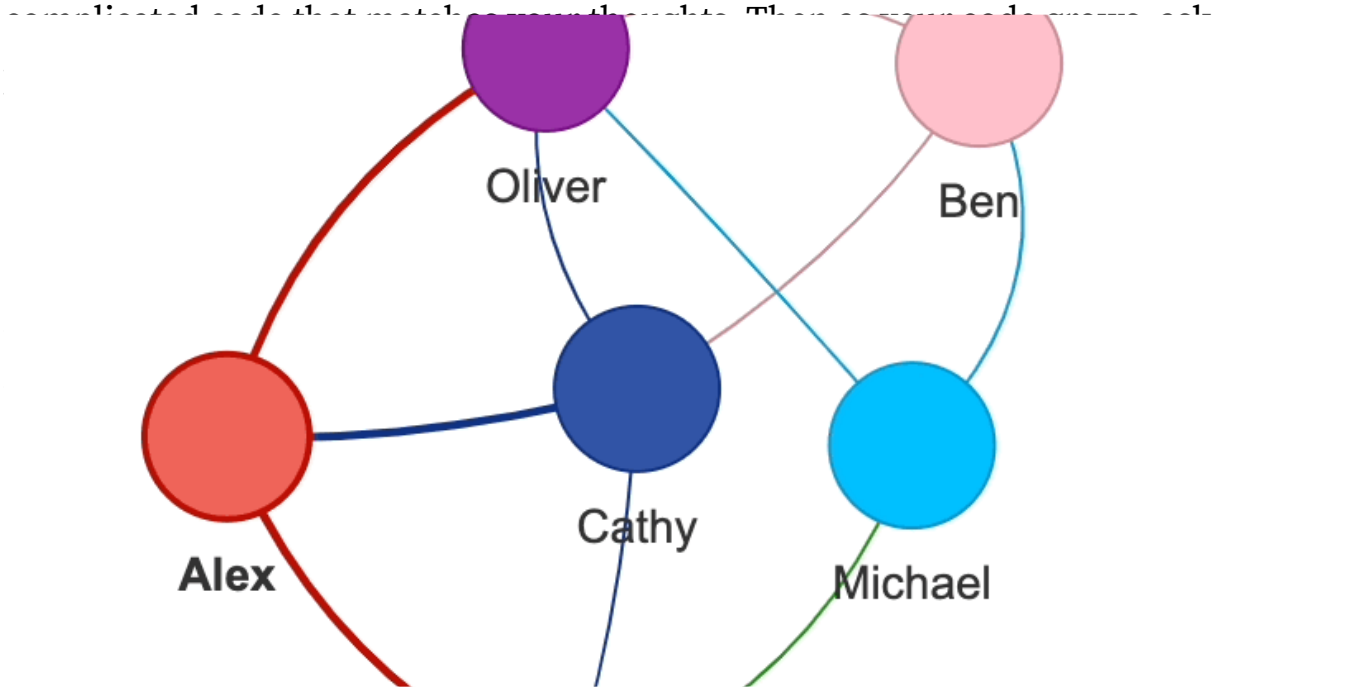
MLOps Engineer. Website: https://mathdatasimplified.com

As we can see, none of the functions above have more than 3 arguments! Even though the code that uses a class is longer compared to the code that uses a function, it is much more readable! We also know exactly what each piece of code does.

How do i write a function like this?

More from Khuyen Tran and Towards Data Science

Don't try to be perfect when starting to write code. Start with writing down



Khuyen Tran in Towards Data Science

Pyvis: Visualize Interactive Network Graphs in Python

All it takes is a few lines of code

Jun 12, 2021 830 8

I like to write about basic data science concepts and play with different algorithms and data science tools. You could connect with me on LinkedIn and Twitter.

Star this repo if you want to check out the codes for all of the articles I have written.

Follow me on Medium to stay informed with my latest data science articles like these:

Pytest for Data Scientists

A Comprehensive Guide to Pytest for your Data Science Projects

towardsdatascience.com

Stop Using Print to Debug in Python. Use Icecream Instead

Are you Using Print or Log to Debug your Code? Use Icecream instead.

towardsdatascience.com

towardsdatascience.com

Ahmed Besbes in Towards Data Science

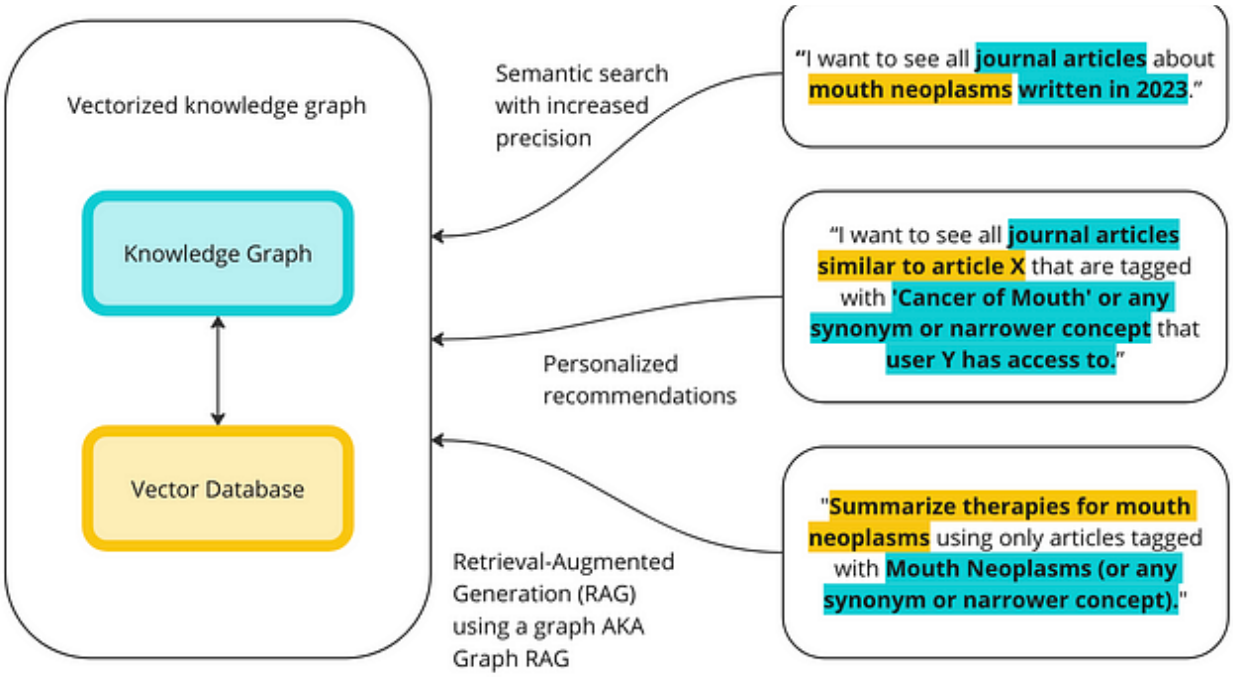
What Nobody Tells You About RAGs

A deep dive into why RAG doesn't always work as expected: an overview of the business value, the data, and the technology behind it.

Aug 23 1.6K 24

References

Martin, R. C. (2009). *Clean code: A handbook of agile software craftsmanship*. Upper Saddle River: Prentice Hall.

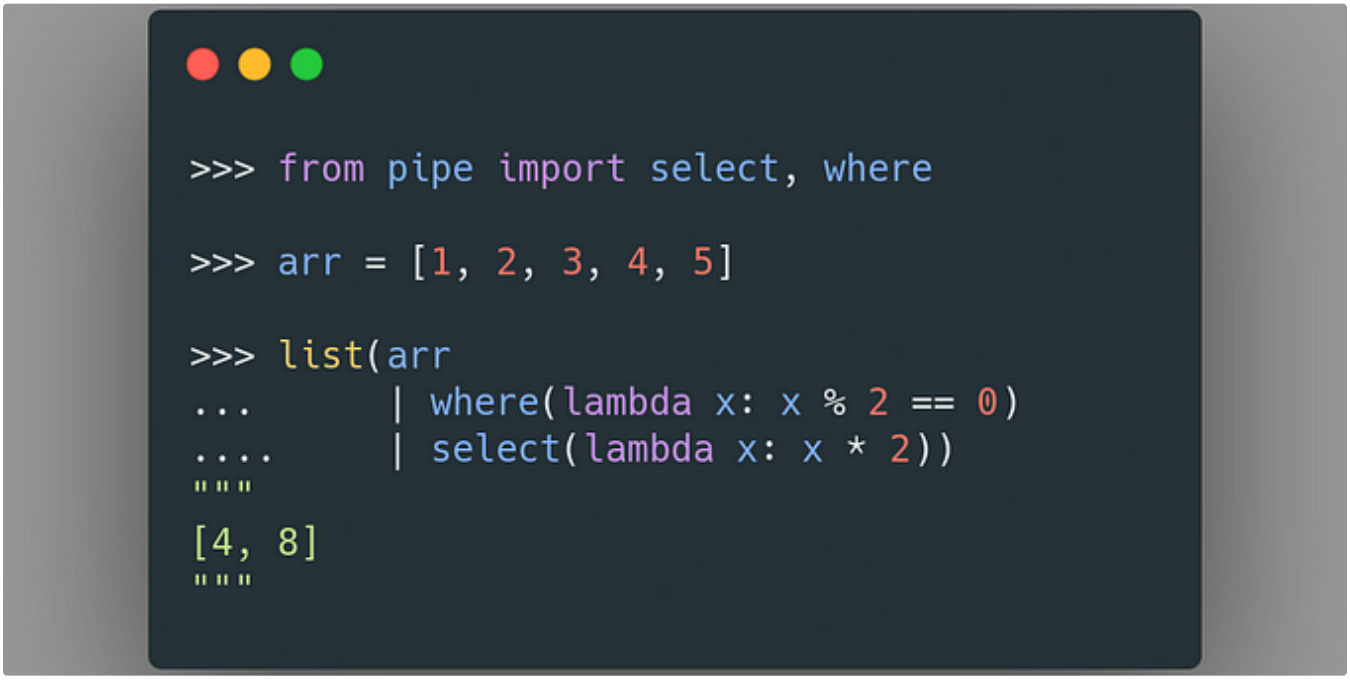


Steve Hedden in Towards Data Science

How to Implement Graph RAG Using Knowledge Graphs and Vector Databases

A Step-by-Step Tutorial on Implementing Retrieval-Augmented Generation (RAG), Semantic Search, and Recommendations

Sep 6 1.1K 13



Khuyen Tran in Towards Data Science

Write Clean Python Code Using Pipes

A Short and Clean Approach to Processing Iterables

Oct 27, 2021 3K 38

Recommended from Medium

See all from Khuyen Tran

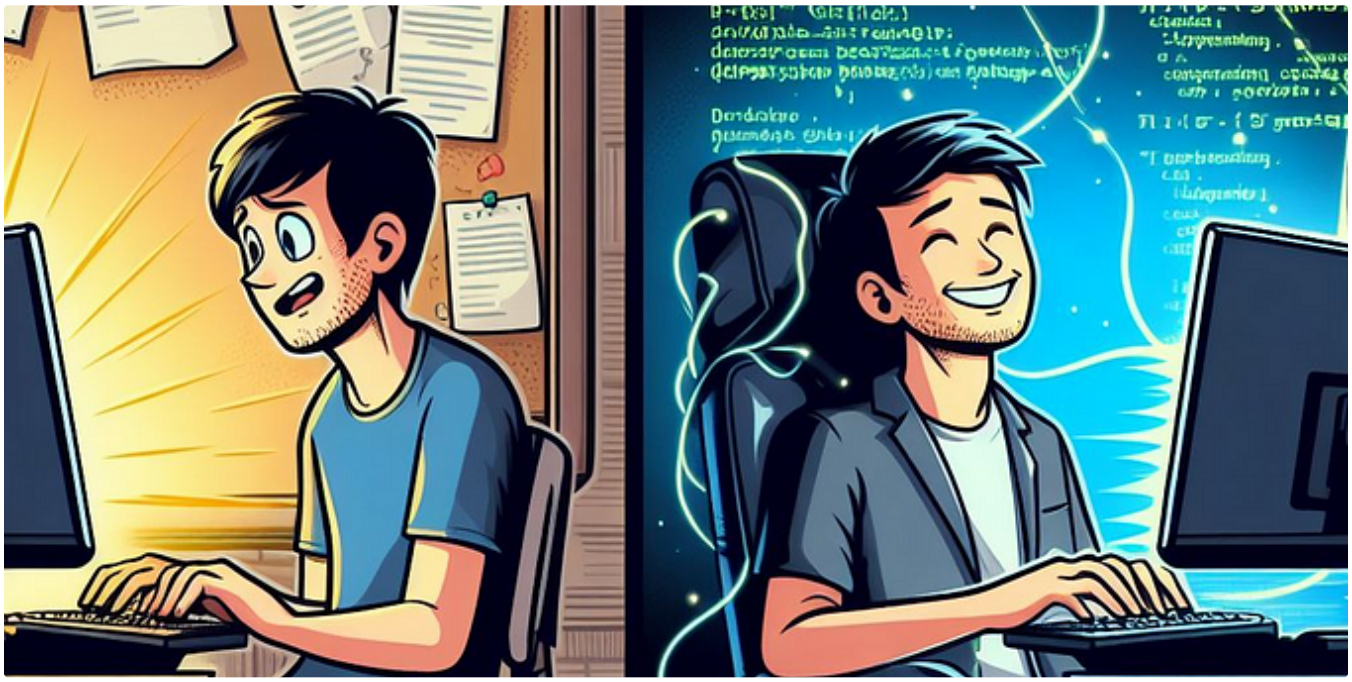
See all from Towards Data Science

Tomas Gonzalez

Microservices with FastAPI and poetry project management

In the past years, pipenv and Makefile were my favorite project management tools. But recently, I began to manage my projects with Poetry.

Jul 1 48



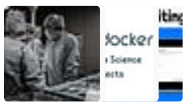
Abhay Parashar in The Pythoneers

17 Mindblowing Python Automation Scripts I Use Everyday

Scripts That Increased My Productivity and Performance

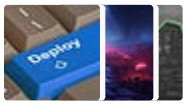
Aug 25 7.9K 73

Lists



Coding & Development

11 stories · 817 saves



Predictive Modeling w/ Python

20 stories · 1548 saves

Practical Guides to Machine Learning

10 stories · 1877 saves

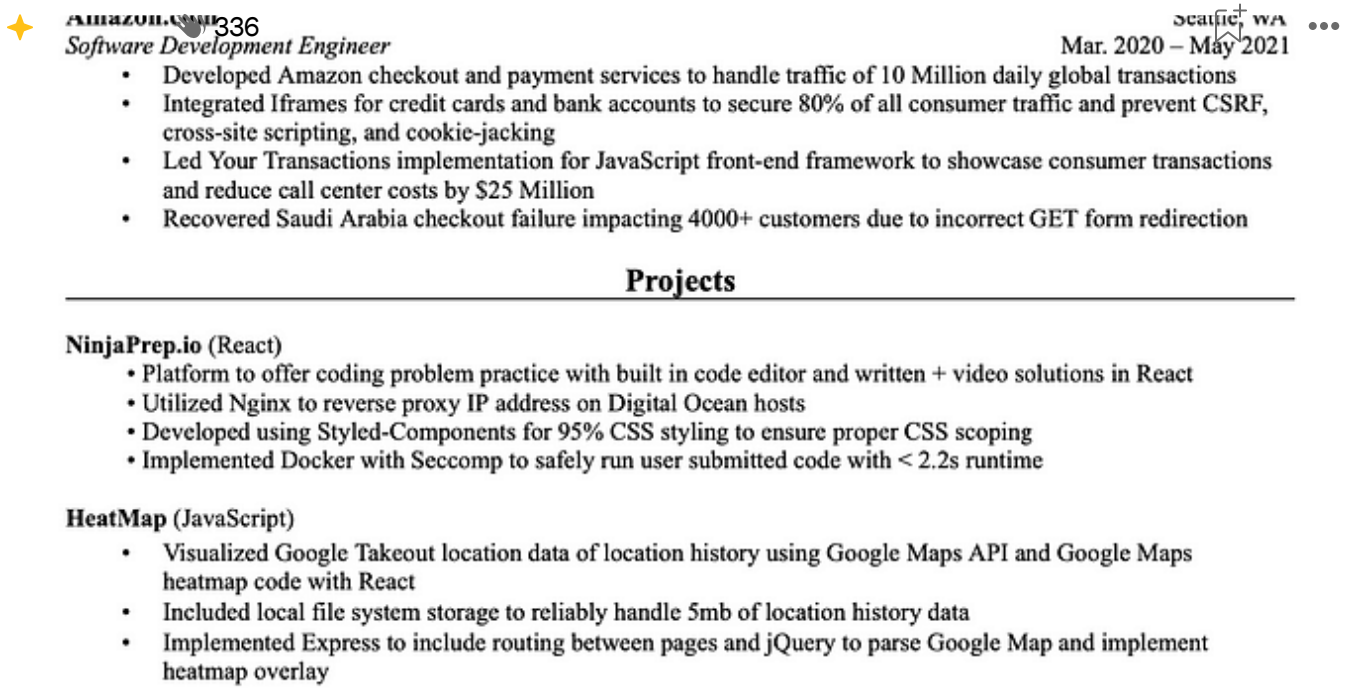
ChatGPT

21 stories · 811 saves

Marcin Kozak in Towards Data Science

Overwriting in Python: Tricky. Dangerous. Powerful

Although overwriting objects is a typical Python coding technique, it can lead to unexpected effects. You need to know how to use it.

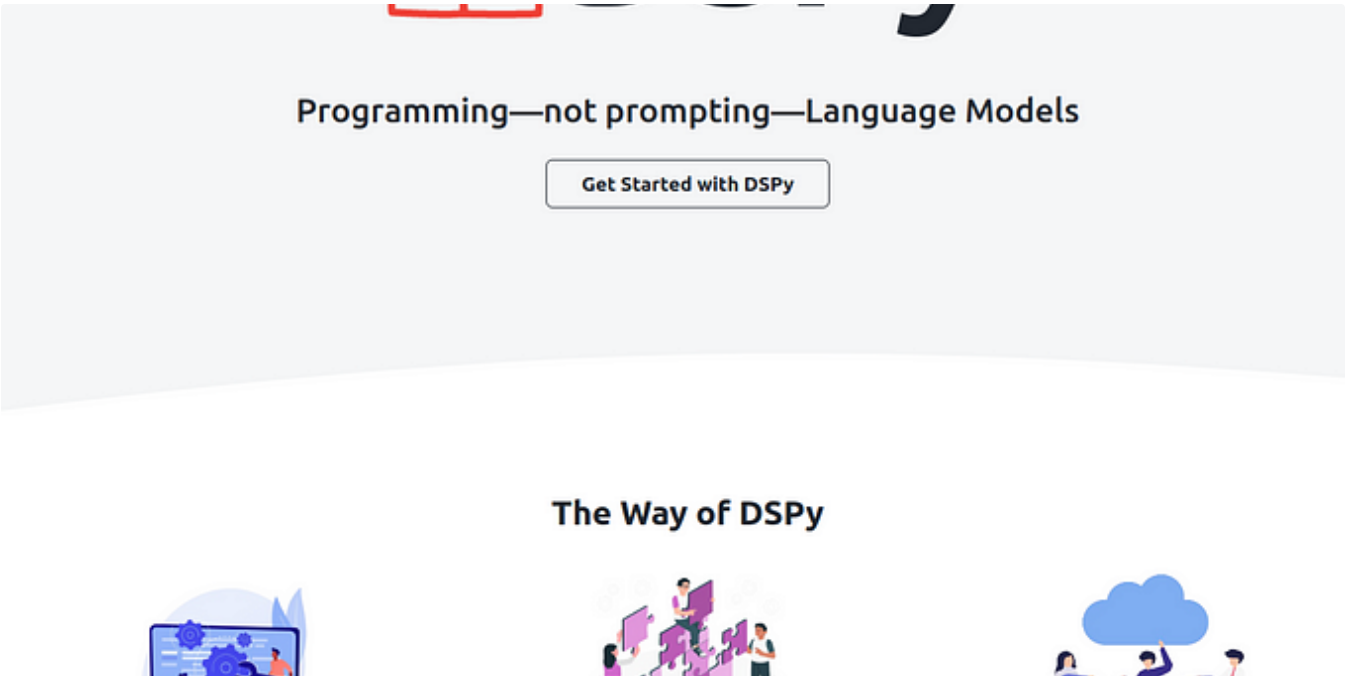


Alexander Nguyen in Level Up Coding

The resume that got a software engineer a $300,000 job at Google.

1-page. Well-formatted.

Jun 1 22K 425



Vishal Rajput in AIGuys

Prompt Engineering Is Dead: DSPy Is New Paradigm For Prompting

DSPy Paradigm: Let's program — not prompt — LLMs

May 29 4.95K 67

Gabe Araujo, M.Sc. in Python in Plain English

How I Automated My Entire Morning Routine with Python

May 16, 2023 2.5K 23

See more recommendations