**CYB 333 Week 1 Environment Set up**

Jonathan Ricafrente

National University, School of Engineering and Computing, BS Cybersecurity Program

CYB 333 Secure Windows Administration

Sean Ayersman

Jun 21, 2025

**1. GitHub Account Setup**

* Create GitHub account
  + <https://docs.github.com/en/get-started/onboarding/getting-started-with-your-github-account>
* Apply for Student Developer Pack
  + <https://github.com/education/students>
* Configure two-factor authentication (Optional, but encouraged)
  + <https://docs.github.com/en/authentication/securing-your-account-with-two-factor-authentication-2fa/configuring-two-factor-authentication>
* Create initial repository for course work.

**2. Download and Install Visual Studio (VS) Code**

Note: If you already have an IDE installed, you can use that environment. You will need to research how to set up Github Copilot in that environment.

* Download Visual Studio Code for your operating system
  + <https://code.visualstudio.com/>
* Install the following extensions in VS CODE
  + Python
    - <https://marketplace.visualstudio.com/items?itemName=ms-python.python>
  + GitHub Copilot
    - <https://marketplace.visualstudio.com/items?itemName=GitHub.copilot>
  + Jupyter
    - <https://marketplace.visualstudio.com/items?itemName=ms-toolsai.jupyter>

**3. Set up Anaconda**

* <https://code.visualstudio.com/docs/python/environments>

**4. Environment Integration**

* [Connect VS Code to Anaconda](https://docs.anaconda.com/working-with-conda/ide-tutorials/vscode/)
* [Link GitHub account](https://code.visualstudio.com/docs/sourcecontrol/github)
* [Configure Jupyter integration](https://code.visualstudio.com/docs/datascience/jupyter-notebooks) (This is optional, we will use in week 3)
* [Test Copilot functionality](https://code.visualstudio.com/docs/copilot/overview)

**5. Documentation & Reflection**

Write a 3-5 paragraph reflection addressing your experience setting up your environment. Possible reflection items can include:

* Setup process and challenges
* Security considerations
* AI tool implications
* Professional workflow benefits
* Future automation possibilities

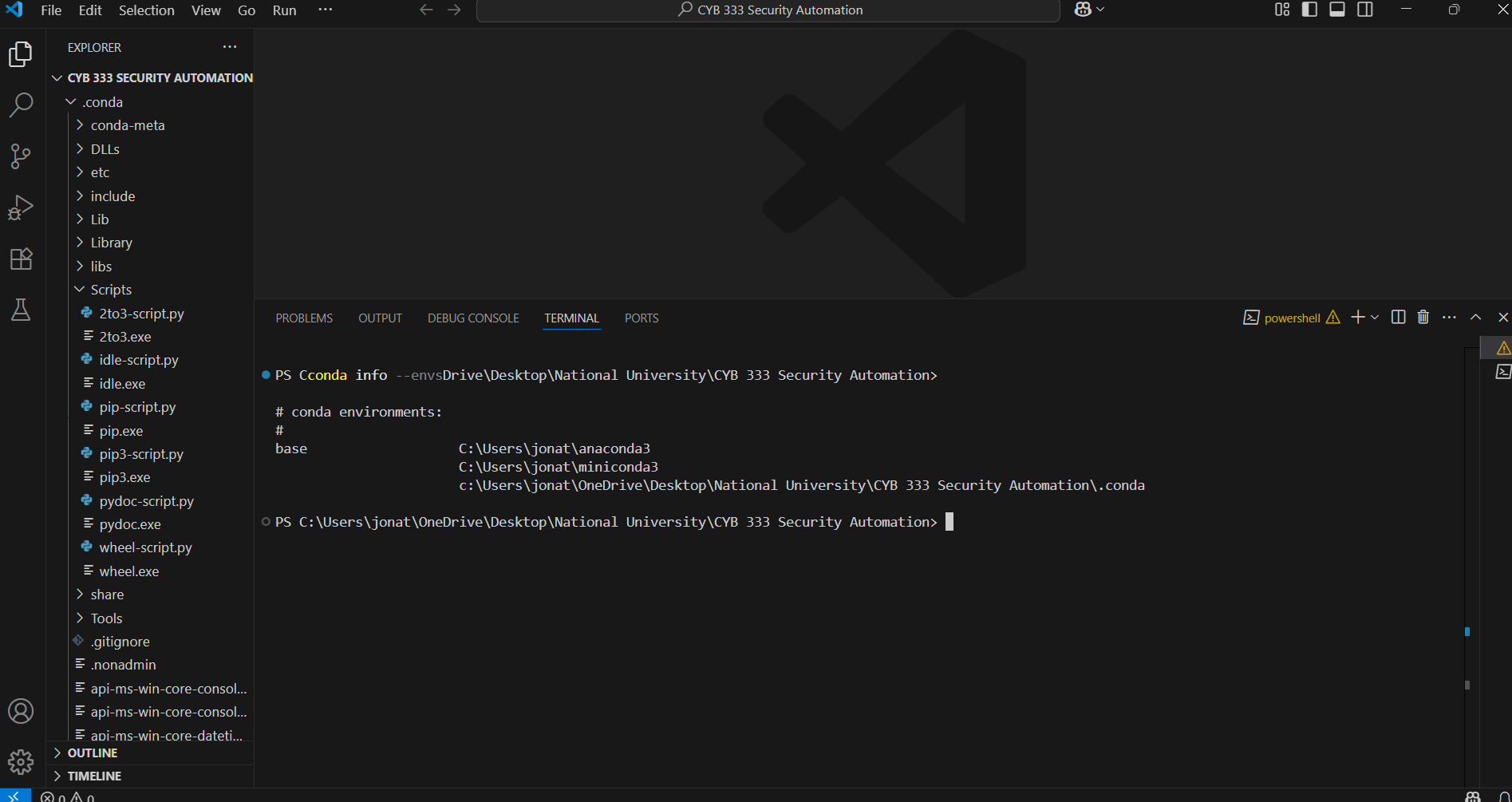
This is a personal reflection to reinforce your learning. Do not use AI to write this. You can use tools like Grammarly to QA your writing. If you set up different environment, describe that environment here.

**Deliverables**

**Submit a word document with the following items:**

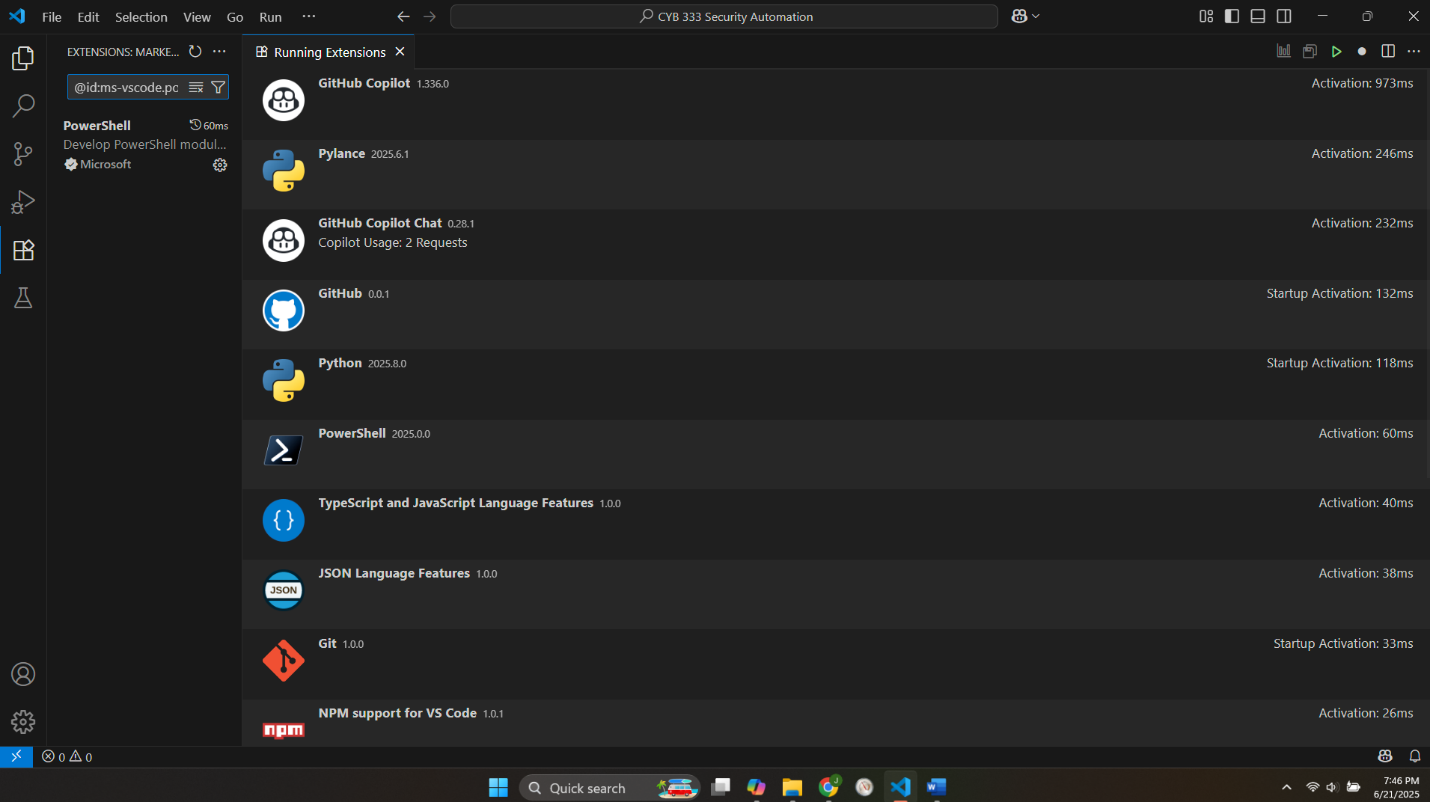
Environment Screenshots

Anaconda environment list





VS Code extensions



Working Jupyter notebook

GitHub profile



Reflection

Name the document "lastname+w1-env.docx".

# Mortgage Comparison Notebook

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

# Define mortgage parameters

mortgage\_1 = {'name': 'Option A', 'principal': 300000, 'annual\_rate': 0.04, 'years': 30}

mortgage\_2 = {'name': 'Option B', 'principal': 300000, 'annual\_rate': 0.035, 'years': 15}

def calculate\_amortization(principal, annual\_rate, years):

monthly\_rate = annual\_rate / 12

payments = years \* 12

monthly\_payment = principal \* (monthly\_rate \* (1 + monthly\_rate) \*\* payments) / ((1 + monthly\_rate) \*\* payments - 1)

return monthly\_payment, payments

def mortgage\_summary(mortgage):

monthly\_payment, payments = calculate\_amortization(

mortgage['principal'], mortgage['annual\_rate'], mortgage['years']

)

total\_cost = monthly\_payment \* payments

total\_interest = total\_cost - mortgage['principal']

return {

'Name': mortgage['name'],

'Monthly Payment': round(monthly\_payment, 2),

'Total Cost': round(total\_cost, 2),

'Total Interest': round(total\_interest, 2),

'Payments': payments

}

# Compare mortgages

summary\_1 = mortgage\_summary(mortgage\_1)

summary\_2 = mortgage\_summary(mortgage\_2)

df = pd.DataFrame([summary\_1, summary\_2])

print(df)

# Plot cumulative payments

def cumulative\_payments(mortgage):

monthly\_payment, payments = calculate\_amortization(

mortgage['principal'], mortgage['annual\_rate'], mortgage['years']

)

return np.cumsum([monthly\_payment] \* payments)

plt.figure(figsize=(10, 5))

plt.plot(cumulative\_payments(mortgage\_1), label=mortgage\_1['name'])

plt.plot(cumulative\_payments(mortgage\_2), label=mortgage\_2['name'])

plt.title('Cumulative Mortgage Payments Over Time')

plt.xlabel('Month')

plt.ylabel('Cumulative Payment ($)')

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()