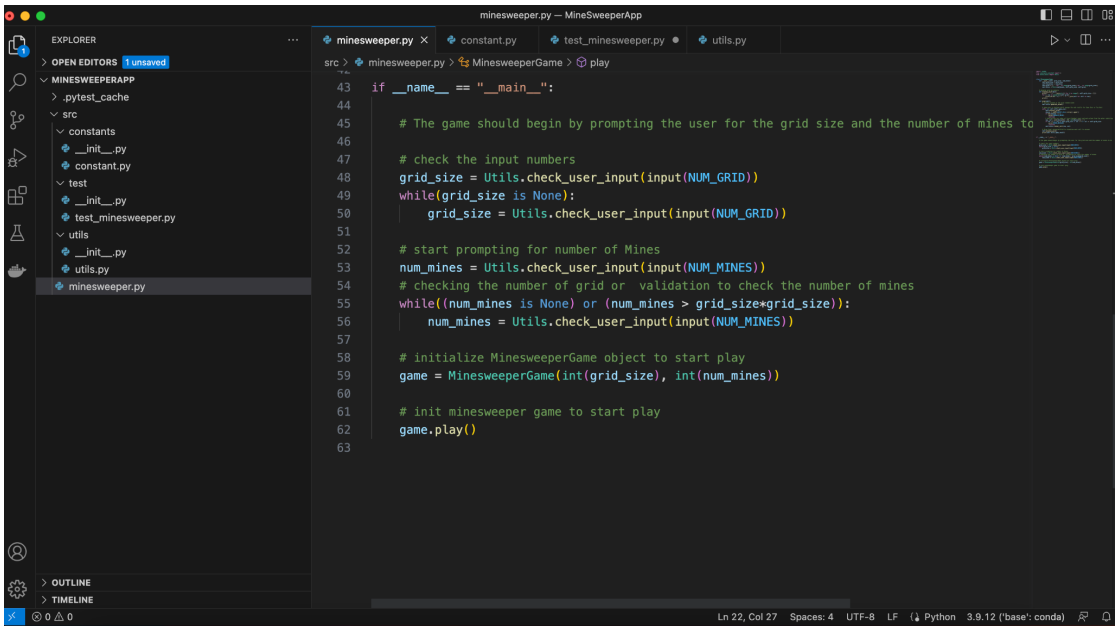
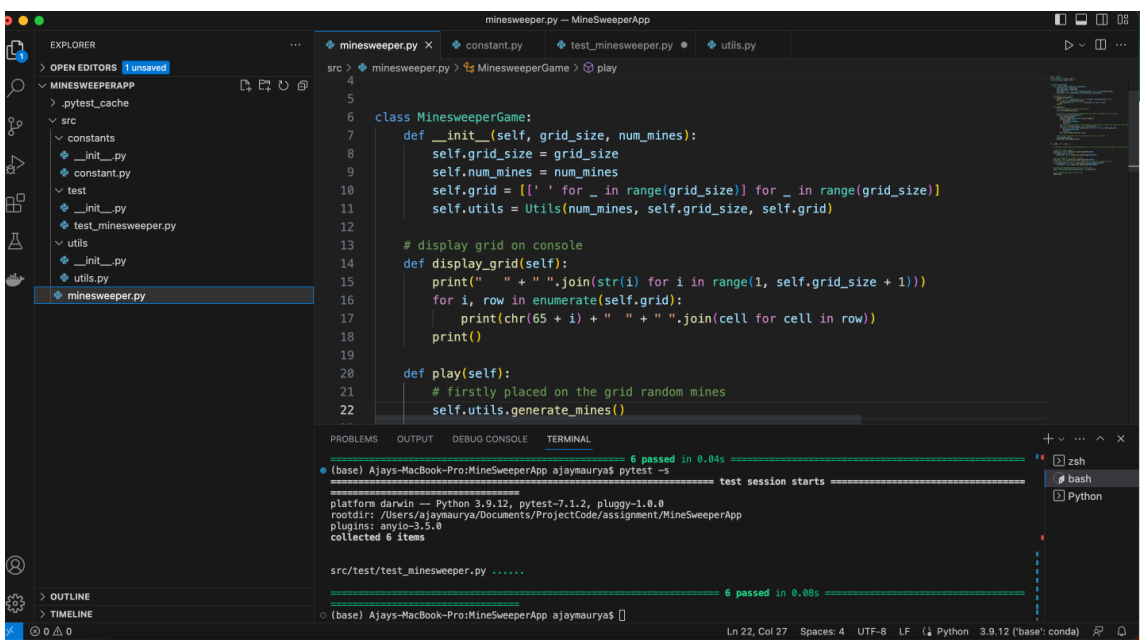
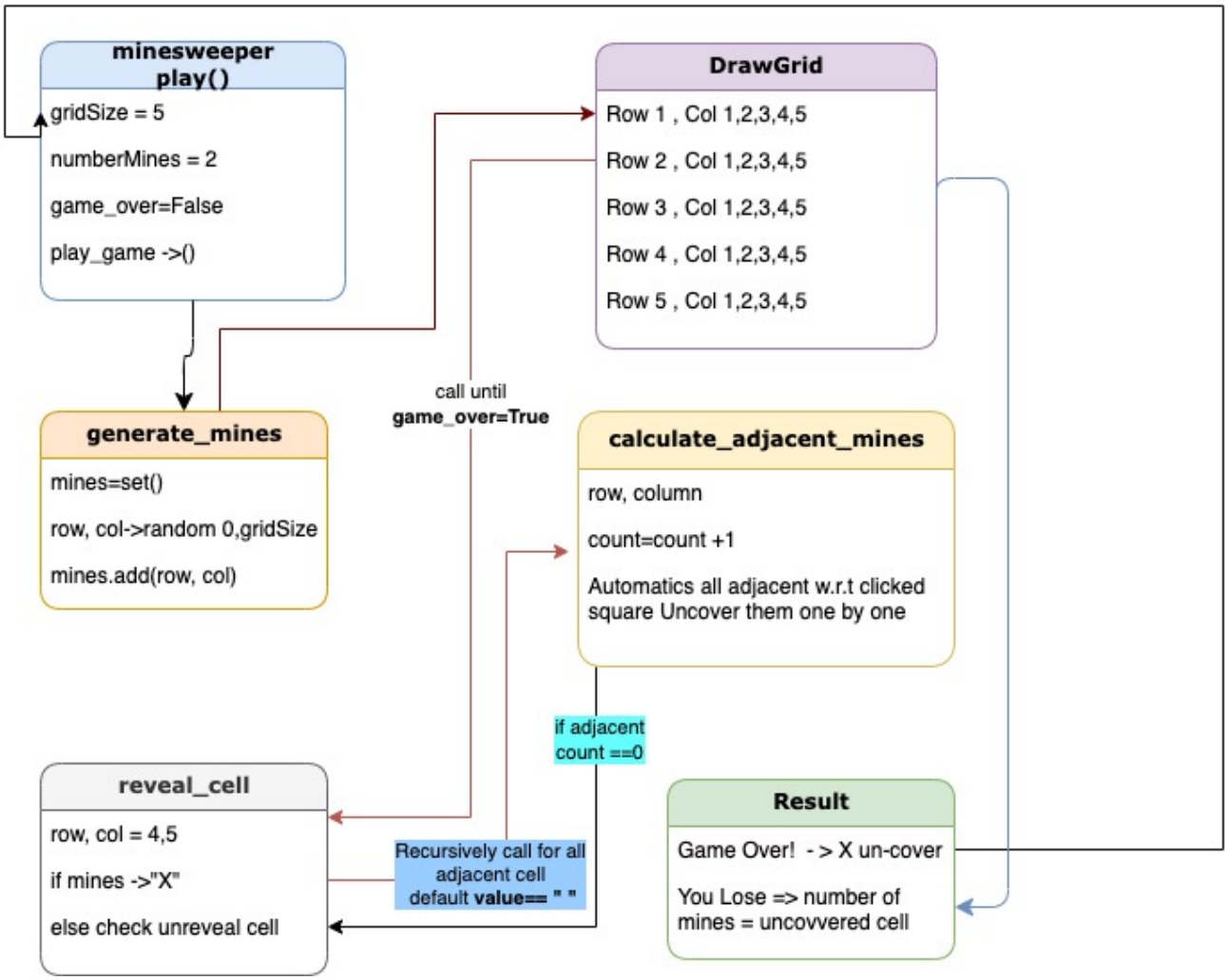
MineSweeper Game Application

Brief Explanations: Minesweeper is a game on a Square grid matrix application that I have developed from scratch in **Python Programming Language** and demonstrated a good understanding of clean code in a modular approach, fully object-oriented programing based on class and object, and the SOLID principle of the software development lifecycle. Also, Included **PyTest BDD** to solve this MinSweeper Game application.
Also, I have created a package folder to maintain the handlers and constants in the specific folder structure. Please refer to the screenshot below.





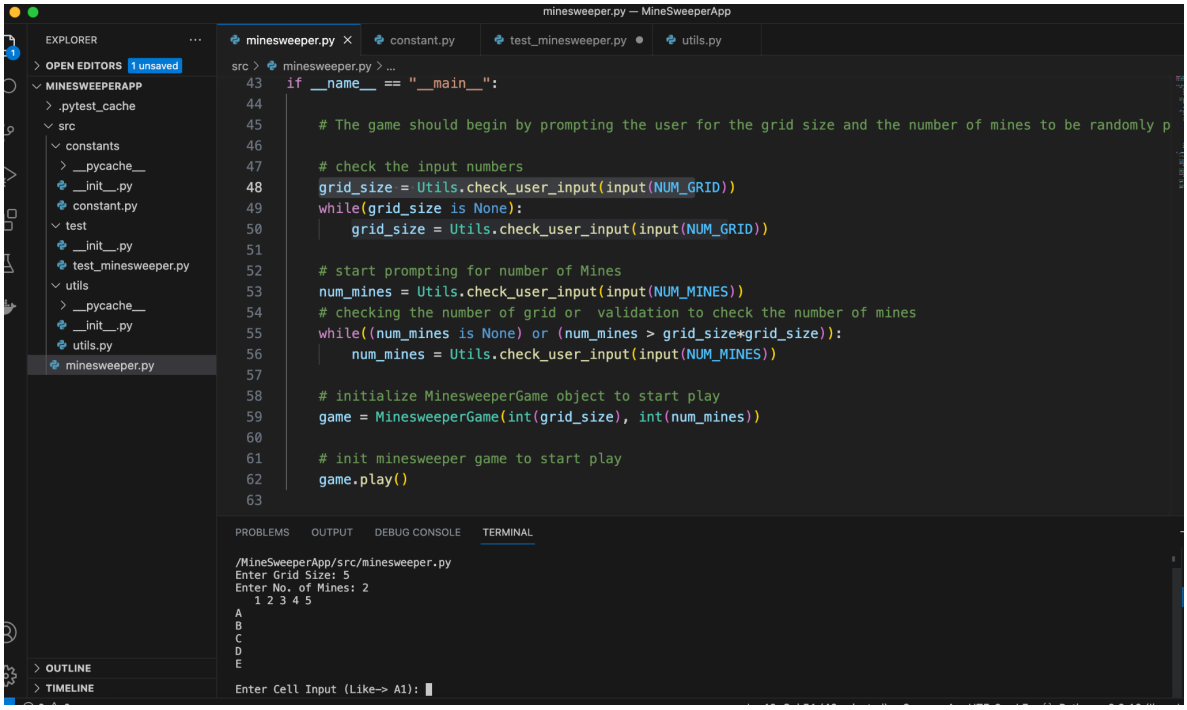
DESIGN and ASSUMPTIONS: I have created some steps to cover the implementations as follows.



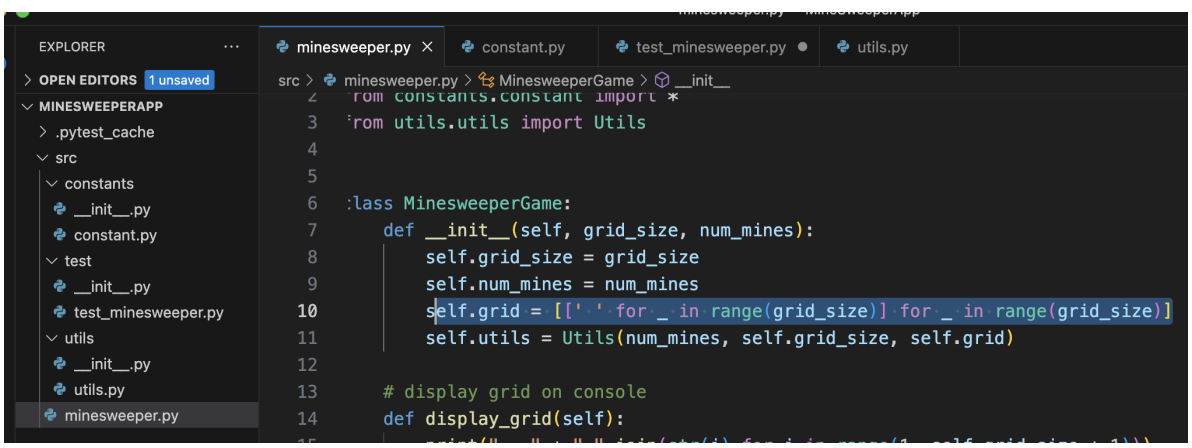
Note* Refer below to set up the project on a local machine and run the application

`

Step 1: Create **minesweeper.py** class to prompt the number of grid sizes and number of mines in terminal input.



Step 2: Create **minesweeper.py** class that holds the value of **grid_size**, the number of **mines**. In this step itself, based on the grid size I created a matrix array with the entered size variable name "**grid**". In the Utils object implemented main business logic about the reveal_cell and game over and game_result. **Based on the above entries it 5X5 grid and 2 Mines**



Utils: Initializing the Utils class by defining the required variable and game status.

`

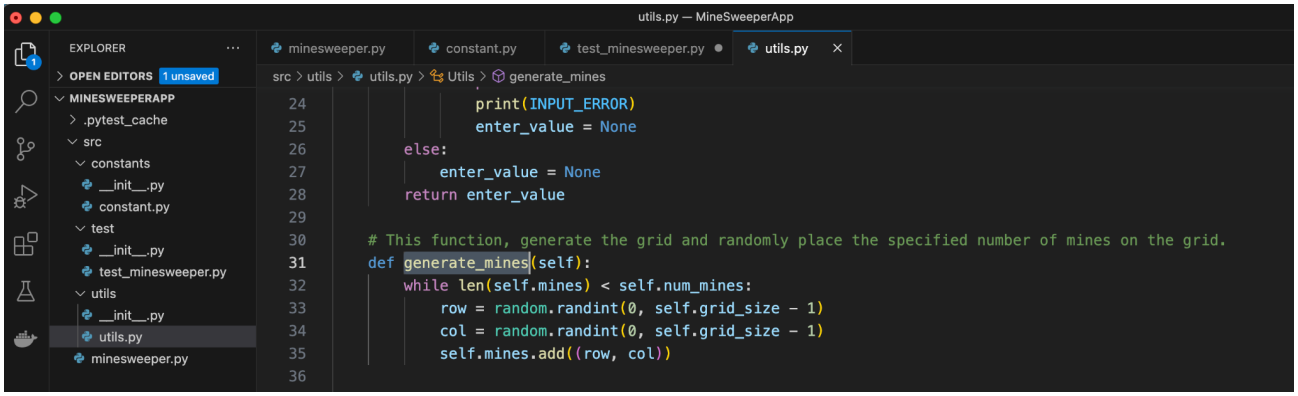
```python
import random
from constants.constant import *


class Utils:
    def __init__(self, num_mines, grid_size, grid):
        self.num_mines = num_mines
        self.mines = set()
        self.grid_size = grid_size
        # -1 because the first one always gets uncover first
        self.uncovercell = int(grid_size) * int(grid_size) - 1
        self.grid = grid
        self.game_over = False
        self.game_result = ""

    # this is static function I have created for the validating the enter number
    @staticmethod
    def check_user_input(enter_value):
        # checking the number of grid or validation to check the number of mines
        if enter_value:
            try:
```

Step 3: Then trigger the play game function which is created in this class **minesweeper.py**. It will

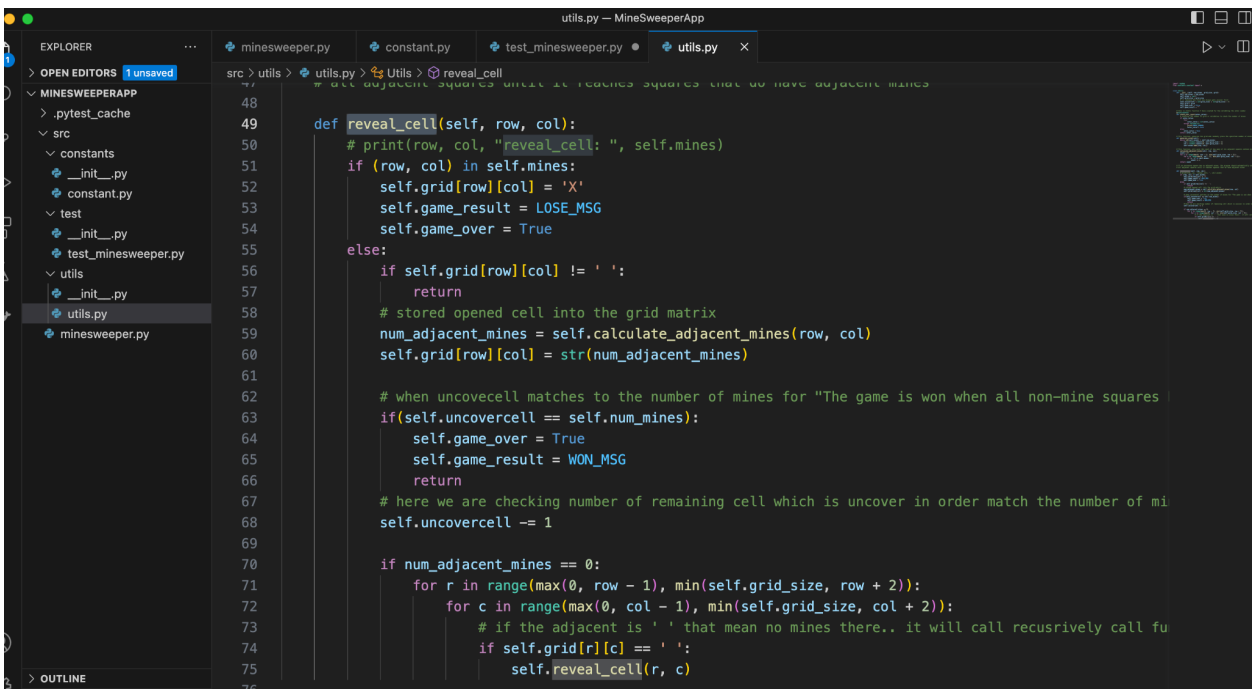
```python
        print(chr(65 + i) + " " + " ".join(cell for cell in row))
    print()

def play(self):
    # firstly placed on the grid random mines
    self.utils.generate_mines()

    # game_over is boolean which manage the end results for Game Over or You Won!.
    while not self.utils.game_over:
        self.display_grid()
        choice = input(SELECT_CELL).strip().upper()
        if len(choice) != 2:
            print(INVALID_ENTRY)
            continue
        # Getting the row index of from Alphabet input and and column from the enter subst
        row, col = ord(choice[0]) - 65, int(choice[1]) - 1
        if row < 0 or row >= self.grid_size or col < 0 or col >= self.grid_size:
            print(ERROR_MESSAGE)
            continue
        self.utils.reveal_cell(row, col)

    # Draw final uncovered Grid to visualise next cell to uncover
    self.display_grid()
    print(self.utils.game_result)
```

Step 4: In this step, based on the number input for mines, we will generate random places where mine in **utils.py file**.

`



Step 5: Select a specific square cell and call utils function "**reveal_cell**" if the cell "**X**" i.e. Mines, then you lose, else it will check the adjacent count to uncover all possible cells automatically.



<u>Approach:</u>

1. Check the min exists (X) in the place of the selected cell square. If yes then Game Over!
2. Else, it will be checked whether the cell is uncovered already or not. If not then I am calculating the number of mines in that row, col adjacent. Automatically, it will look for all other adjacent recursively by calling the function "**reveal_cell()**" if the cell squared **default value** is **" "**
3. If all adjacent uncovered and remaining adjacent are equal to the number of covered cells then the user will be a winner.

The environment required to run the application: In Windows/MacOS

Install Python3 on the local machine. Make sure in the local machine the python is installed.

Download and Install Python:

https://www.python.org/downloads/

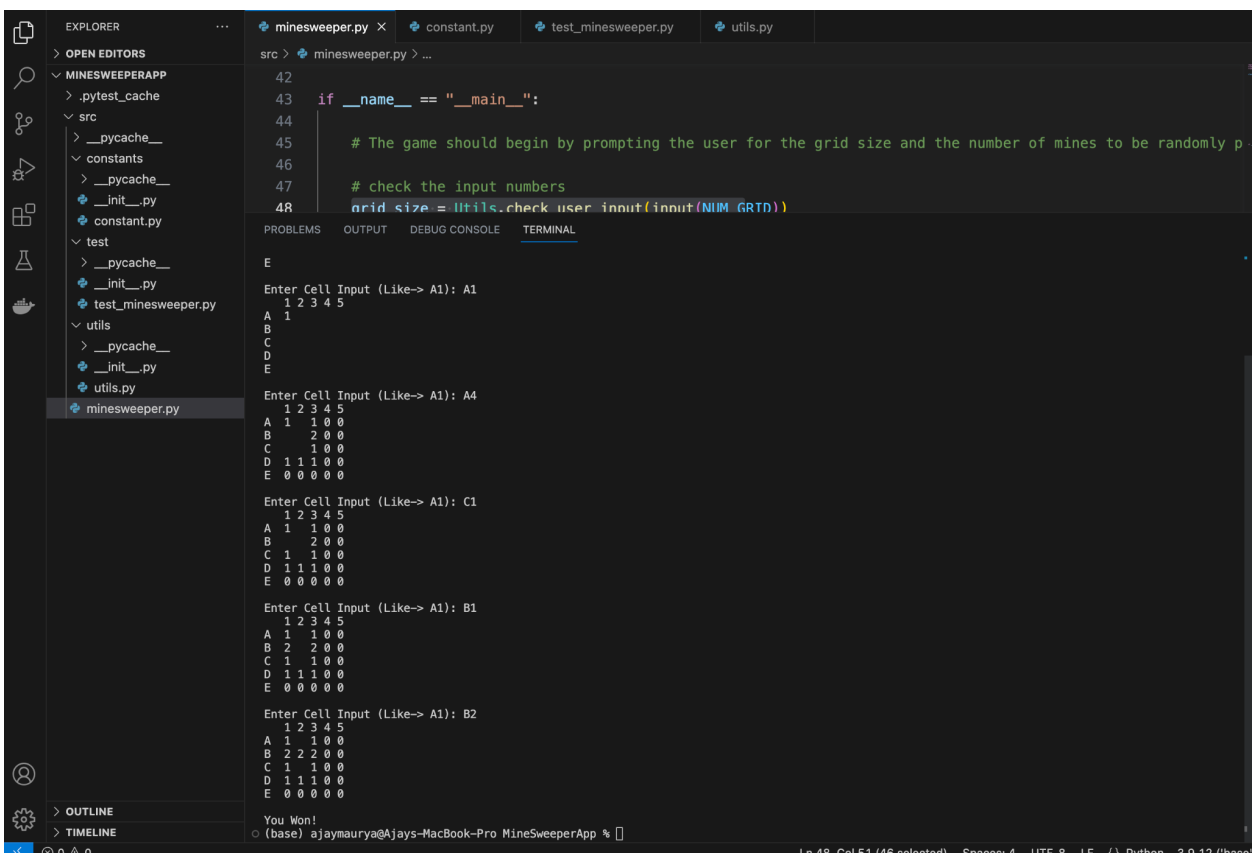
Make sure Python is installed using the below command.

> python -V

Python 3.8.4

Open Visual Studio Code or IDE which you prefer: Run the Application using the below command to start playing the minesweeper Game:

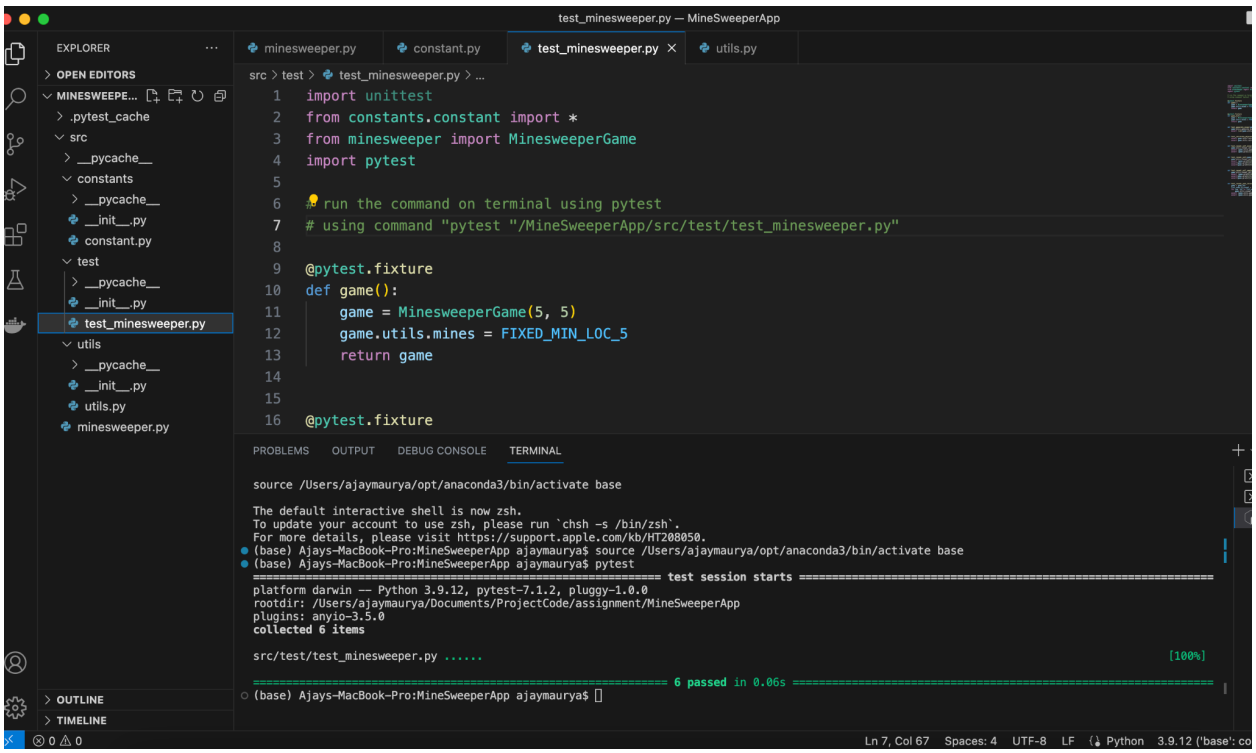
1. **python minesweeper.py**



`

TDD pyTest: Run the below command on the terminal it will automatically execute the created **test_minesweeper.py** class. Please refer below screenshot

> pytest



Note* How to Install pytest on local machine?

1. Type "cmd" in the search bar and hit Enter to open the command line.
2. Type “**pip install pytest**” (without quotes) in the command line and hit Enter again. This installs pytest for your default Python installation.
3. The previous command may not work if you have both **Python** versions **2** and **3** on your computer. In this case, try "**pip3 install pytest**" or “**python -m pip install pytest**“.
4. Wait for the installation to terminate successfully. It is now installed on your Windows machine