implementing the dynamic maze generation system

1. **Set up Project Structure**:
   * Create a new directory for your project.
   * Inside the project directory, create subdirectories for organizing your code, such as **maze\_generator**, **agent**, **pathfinding**, **visualization**, and **evaluation**.
2. **Implement Dynamic Maze Generation**:
   * Create a Python script (**maze\_generator.py**) inside the **maze\_generator** directory.
   * Use NumPy to create a grid-based maze representation.
   * Implement algorithms for generating dynamic maze layouts, such as recursive backtracking, randomized Prim's algorithm, or cellular automata.
   * Use Pygame to visualize the dynamic maze environment and display it on the screen.
3. **Test the Maze Generation**:
   * Test the dynamic maze generation system to ensure that it produces valid and visually appealing maze layouts.
   * Experiment with different parameters and algorithms to generate a variety of maze configurations.

With implementing the dynamic maze generation system using Python and libraries like NumPy and Pygame for visualization.

Install pygame

* + Cmd 🡪 pip install pygame
  + Pip show pygame

Trouble

A screen shot of a computer

Description automatically generated

Here's how you can set up VS Code to use the Anaconda Python interpreter:

1. Open VS Code and navigate to your Python script.
2. Open the Command Palette by pressing Ctrl+Shift+P.
3. Type "Python: Select Interpreter" and press Enter.
4. You should see a list of available Python interpreters. Look for the one located in your Anaconda environment (**C:\Users\chand\anaconda3\...**) and select it.
5. VS Code will now use the selected Python interpreter for running your script, and it should be able to import Pygame from the Anaconda environment.