This is the explanation that AI had given me:

Shader Graph in Unity is a visual tool that allows developers and artists to build shaders using a node-based interface, without writing code. It enables the creation of both simple and complex shaders by connecting nodes that represent mathematical operations, textures, lighting models, and other shader functionalities. Shader Graph is especially useful for real-time visual feedback and experimentation, making it accessible to users with limited programming experience while still offering the power needed for advanced graphics.

Shader Graph integrates tightly with the **Universal Render Pipeline (URP)** and **High Definition Render Pipeline (HDRP)**, providing optimized and flexible workflows tailored to different project needs. Users can create custom surface shaders, unlit effects, or procedural visuals, and reuse subgraphs to maintain clean and modular designs. Once compiled, Shader Graphs generate HLSL code behind the scenes, which Unity uses to render the desired material effects on 3D objects. This tool significantly streamlines shader development, especially for teams with both technical and non-technical members.

How I interacted with it:

On the most basic level, the shader graph is a node-based coding interface similar to Unreal's system. This system streamlines all accessible parameters within a material from code to a less abstracted visual form. Which makes automating shader qualities and mesh materials more approachable for "both technical and non-technical members". This system allows interactivity through scripts as well. For example: you can call the system through script to play a "animation" on collision. Wether this be a color flash or the geometry shifting, this can be a easly and effective to add visual distinction/interactivity in a game for the non-artist.

To implement:

1. **Change Render Pipeline**: Shader Graph only works with **URP (Universal Render Pipeline)** or **HDRP (High Definition Render Pipeline)** thess are rendering pipelines that allow for access and manipulation of the material and geometry of an GameObject.

2. **Switch to URP or HDRP**:
    - Go to *Edit → Project Settings → Graphics* and assign a URP/HDRP pipeline asset.
    - Use the *URP/HDRP Wizard* (Window → Rendering → Render Pipeline Converter) to upgrade materials automatically if needed.

3. **Create a Shader Graph**:

    - Right-click in the Project window → *Create* → *Shader* → *URP (or HDRP)* → *[Shader Type]*.
    - Double-click the new shader to open it in Shader Graph

4. **Create a Material** and assign the shader you create. To add this to a Gameobject, select the GameObject and locate the MeshRender → materials → press + on elements → drag in shaderGraph → press - on element 0 and it should be up and running.