# Certified Agentic & Robotic AI Engineer – Quarter 1, Class 1 (Full Step-by-Step Report)

**Date:** Sunday, August 17, 2025
**Time:** 2:00 PM – 6:00 PM
**Venue:** Boys Scout Auditorium
**Program Quarter:** 1
**Instructors:** Mr. Aneeq & Mr. Hamza
**Session Type:** Orientation + Hands-on (Python • VS Code • n8n)
**Document created by:** Taha Jalal

---

## 1) What This Quarter Covers (as announced by instructors)

Quarter 1 focuses on three building blocks. Each item below reflects exactly what the teachers said we will learn, plus precise explanations so a newcomer can follow:

1. **Basics of Python**
   *What it is & why it matters:* Python is the core programming language we'll use to write logic, build utilities, and later connect automation and agentic systems.
   *How we will learn it:* From installing Python to writing simple programs, running them in different environments, and understanding the difference between quick tests (REPL) and saved scripts.
2. **n8n (Workflow Automation)**
   *What it is:* A low-code/no-code workflow tool where you design flows by adding nodes and connecting them visually.
   *What we will do:* Learn what n8n is, how flows are designed, where drag-and-drop is used, and how to create, run, and schedule workflows.
3. **OOP – Object-Oriented Programming**
   *What it is:* A way to organize code into **Objects** (with data) and **Classes** (their blueprints).
   *Why it matters:* This makes real-world modeling and maintainable agent systems possible. You will learn exactly what OOP is and how to apply it.

---

## 2) AI Types Explained:

During class we discussed three kinds of AI. Below are precise definitions and the examples used by the teachers, clarified in professional English:

### 2.1 Predictive AI

**Definition:** Systems that use existing data to **predict** a likely outcome or next event.
**Teacher's example (explained):** You may notice that after you discuss a product, related ads start appearing on Facebook or Instagram. Behind the scenes, platforms use data signals and predictive models to decide which ads you're most likely to engage with.
**What to remember:** Predictive AI estimates what might happen next; it doesn't generate new content or take actions by itself.

### 2.2 Generative AI

**Definition:** Systems that **generate** new content (text, images, code, etc.) from a prompt.
**Teacher's examples:**

- Asking ChatGPT (or similar tools) to write a social media post for you.
- Requesting a Pakistan tour plan and getting suggested flight timings, destinations, and hotels.
  **Key concept mentioned: LLMs (Large Language Models)** — the underlying models that understand and produce human-like language in response to your prompts.

### 2.3 Agentic AI

**Definition:** Systems that can **take actions on your behalf**, not just predict or generate. They can plan, make calls to services, and execute tasks end-to-end.
**Teacher's examples:**

- Design a post and publish it automatically at a specified time.
- Book flights and hotels, pick destinations and restaurants, and return a ready-to-approve travel package.
  **Design guidance shared in class:**
- Build **one specialized agent per process** (e.g., HR, Finance, Sales) instead of one "does-everything" agent.
- Optionally create an **administrator/orchestrator agent** that analyzes a problem and delegates to the right specialized agent.
  **Upcoming topics (as stated):** Agentic model development, **A2A (Agent-to-Agent communication)**, **OpenAI Agent SDK**, and cloud usage on **Microsoft Azure**.

---

## 3) Python – From zero to running code (every step done in class)

### 3.1 Downloading & Installing Python

1. Open your browser and go to **python.org**.
2. Download the latest stable Windows installer (64-bit).
3. Run the installer. On the first screen, **check** the box **"Add Python to PATH."**
4. Complete the installation with default options unless instructed otherwise.

### 3.2 Verifying that Python is installed (three methods shown)

1. **Windows Search method:** Press the **Windows** key, type **Python**. If Python is installed, you'll see the app and can view its version.
2. **Command Prompt version check:**
   - Open **Command Prompt** (press Windows key, type **Command Prompt**, press **Enter**).
   - At a prompt that looks like:
     ```
     C:\Users\hp>
     ```
     type the command and press Enter:

   -
     ```
     python --version
     ```

     You should see output similar to:
     ```
     Python 3.13.7
     ```

3. **Entering the Python environment (REPL):**
   - In the same Command Prompt, type:

- `python`
- You’ll see information like:
  ```
  Python 3.13.7 (tags/v3.13.7:bcee1c3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32
  ```
  followed by the prompt:
  ```
  >>>
  ```

### 3.3 Running your first line of Python in the REPL

1. At the `>>>` prompt, type:
2. `print("Hello World")`
3. Press **Enter**. The screen prints:
   ```
   Hello World
   ```
4. To exit the Python environment, type:
5. `exit()`

**Important note:** Code typed directly in the REPL **is not saved**. For saved, reusable programs, use a file (script) and an editor like VS Code.

---

# 4) VS Code – Creating and running your first saved Python program

### 4.1 Preparing a dedicated course folder

1. Open **File Explorer** and create a folder on **D:** named:
   `D:\Agentic AI Class`
   This is the folder where all your work for this course will be stored.

### 4.2 Opening the folder in Visual Studio Code

1. Install **Visual Studio Code (VS Code)** if you have not already.
2. Open VS Code → **File** → **Open Folder…** → select `D:\Agentic AI Class` → **Open**.
3. Inside VS Code, create a new file and name it:
   `foundation.py`
   (The **.py** extension tells VS Code this is a Python file. File with different programming language can also be saved with its extension)

### 4.3 Writing and saving the program

1. In `foundation.py`, type the following lines exactly as shown:
2. `print("Hello World")`
3. `print("First Name: Taha")`
4. `print("Last Name: Jalal")`
5. Save the file (**Ctrl + S**).

### 4.4 Running the program from the VS Code Terminal

1. Open the integrated terminal: **View** → **Terminal** (or **Terminal** → **New Terminal**).
2. Ensure the terminal is at your course folder. Examples of prompts you may see (both were shown in class; yours may differ based on where you saved the file):
   - `PS C:\Users\hp\Documents\Agentic AI Class>`
   - or `D:\Agentic AI Class>`

3. Run the program in terminal by typing:
4. `python foundation.py`
5. You should see exactly this output:
6. `Hello World`
7. `First Name: Taha`
8. `Last Name: Jalal`

---

# 5) n8n – Building your first automation exactly as demonstrated

## 5.1 Creating your n8n account

1. Open your browser and search for **n8n.io** (or directly visit **n8n.io**).
2. Click **Get started**.
3. Create your account. You will receive a **14-day trial**.
4. Answer the onboarding questions to set up your workspace.

## 5.2 Creating a new workflow (Manual Trigger → Set → Gmail) (e.g. You want to receive an email daily at 8am but this is a manual workflow)

1. Click **Start automating** → **Create Workflow**. A blank canvas opens.
2. Click **Add first step** and search **Manual Trigger**. Select it. (This is your first *node*.)
3. Click the **+** icon to add another node. Search for **Set**, and then choose **Edit Fields**.
    - In the **Manual Mapping / Add field** area, create the following fields exactly:
        - **Name** → **String** → `Taha Jalal`
        - **Email** → **String** → `abc@gmail.com`
    - Click **Execute step** to preview the JSON output produced by the Set node.
    - Click **back to canvas**.
4. Add a third node: click **+**, search **Gmail**, and select it. Choose the action to **send a message**.
   Fill the fields as follows (exactly as done in class):
    - **Credential to connect with:** Create new credentials → sign in with the Gmail account that will send the email → allow n8n limited access.
    - **Resource:** `Message`
    - **Operation:** `Send`
    - **To:** drag the Email field from the previous node, or type the expression:
      `{{ $json.Email }}`
    - **Subject:** `Reminder for Python`
    - **Email Type:** `Text`
    - **Message:**
    - `Hi {{ $json.Name }},`
    - `This is an Email from n8n.`
5. Click **back to canvas**, and then click **Execute workflow**.
    - n8n will run the nodes from Manual Trigger → Set → Gmail.
    - Check your inbox: the email should arrive at the address you specified in the **Email** field.
6. To **deactivate** this workflow (so it does not keep running), use the **Deactivate** option above the canvas.

## 5.3 Turning the workflow into a schedule (automation) (e.g. You want to receive an email daily at 8am)

The previous steps sent an email manually. To automate it:

1. Delete the **Manual Trigger** node.

2. Click the **+** on the right and select **Run a schedule**.
3. In the schedule form, set when the workflow should run. Examples used in class:
    - **Every minute:** choose **Minutes** and set **1** minute as the interval.
    - **Daily at 8:00 AM:** choose **Day** and set the time to **08:00**.
4. **Activate** the workflow to enable the schedule.
5. If you want to pause the automation later, click **Deactivate**.

**Why we used n8n today as per instructor:** To demonstrate how routine tasks (like sending a daily email) can be automated with a visual workflow. **This is automation, not an agent** — agent behavior comes later when we add decision-making and multi-step task execution via code and AI logic.

---

## 6) Class logistics and expectations (as announced)

- **Discord:** Most lectures and resources will be shared on Discord. A **Class Representative (CR)** will be selected through interviews; the CR will help share lecture notes and materials.
- **WhatsApp:** The Discord invite link will be shared in the class WhatsApp group.
- **Equipment:** Students should bring their **laptop** and **internet data** to every class to follow along with hands-on work.
- **Today's deliverable:** Summarize today's class and **share a PDF** of your summary with the instructors on WhatsApp.

---

## 7) What's coming next (per instructors)

- **Agentic model development**
- **A2A – Agent-to-Agent communication**
- **OpenAI Agent SDK**
- **Working on Microsoft Azure cloud**

---

## 8) Exact commands and prompts used today (for quick reference)

- Command Prompt prompt example:
  `C:\Users\hp>`
- Check Python version:
- `python --version`
- Enter Python REPL:
- `python`
- Print "Hello World" in REPL:
- `print("Hello World")`
- Exit REPL:
- `exit()`
- Run saved script from Terminal (inside VS Code):
- `python foundation.py`
- n8n expression used for dynamic email recipient:
  `{{ $json.Email }}`

**You can hand this document to anyone who missed the session. Reading it end-to-end will recreate the entire 4-hour class experience in precise, professional English, with no steps skipped.**