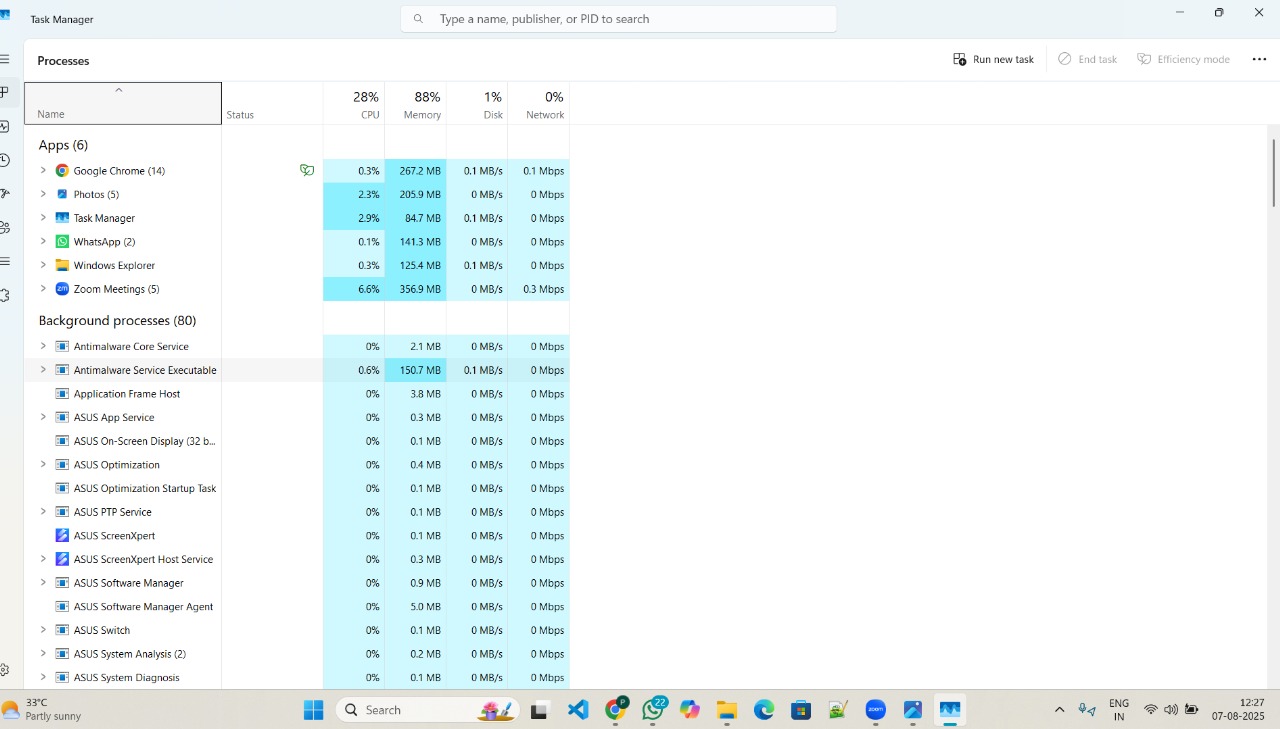
**Today Assignment – Date: August 7,2025**

**Name: 34733-Prasang Kumar Singh**

**Gmail:** [**thakurprasang23@gmail.com**](mailto:thakurprasang23@gmail.com)

**Today topic:-**

**Interactive applications and Non- Interactive applications**



**Interactive Applications**

**Non-Interactive Applications**

**Interactive Applications**

* These apps **need user input** — like clicking, typing, or selecting options.
* They usually have a **graphical interface (GUI)** for users to interact.
* You use them directly — like **web browsers, MS Word, media players**, or games.

**Non-Interactive Applications**

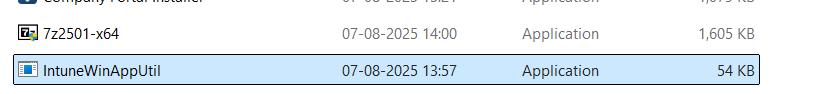
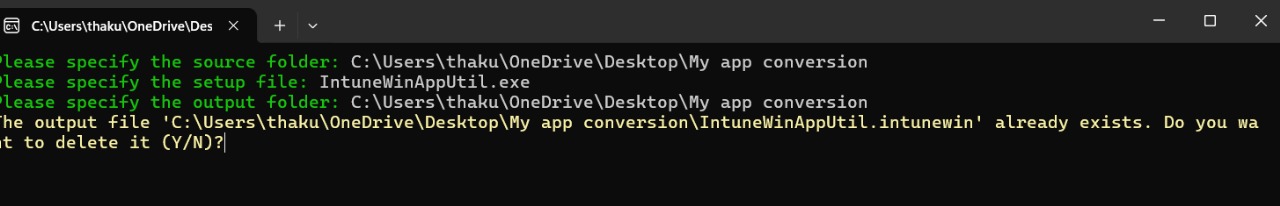
* These apps **run in the background** without any user input.
* They often **don’t have a GUI** and perform tasks on their own.
* Common in **services like printing, software updates**, or antivirus scans.

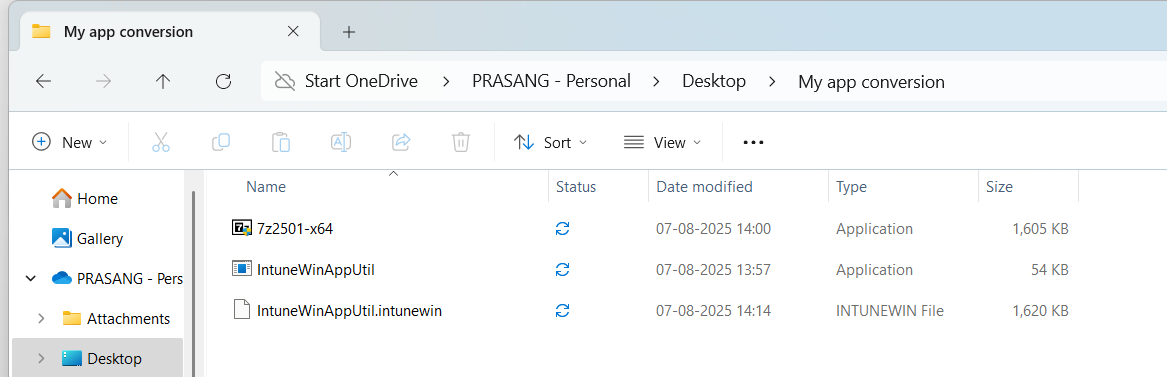
**Intunewin conversion - Compatible version**

**What Is an Intunewin File?**

To upload a Win32 application to **Microsoft Intune**, it must be converted into a **.intunewin** format. This is done using the **Win32 Content Prep Tool** provided by Microsoft.

**Steps to Create a .intunewin File**

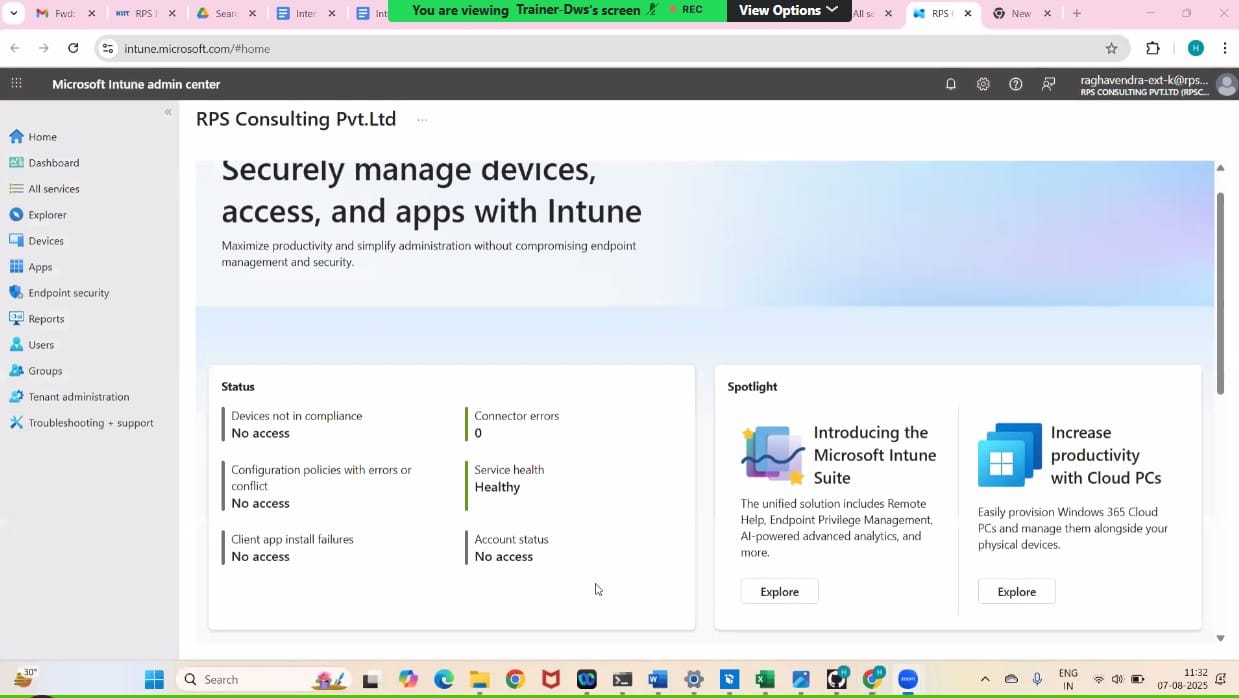
1. **Collect App Files**
   * Gather all installation files needed to install your application.
2. **Use the Prep Tool**
   * Download and run **IntuneWinAppUtil.exe** (Win32 Content Prep Tool) from Command Prompt.
   * It will ask for:
     + Source folder
     + Setup file (setup.exe )
     + Output folder where .intunewin will be saved
3. **Create .intunewin File**
   * The tool will generate a .intunewin file with all required app instructions and content.

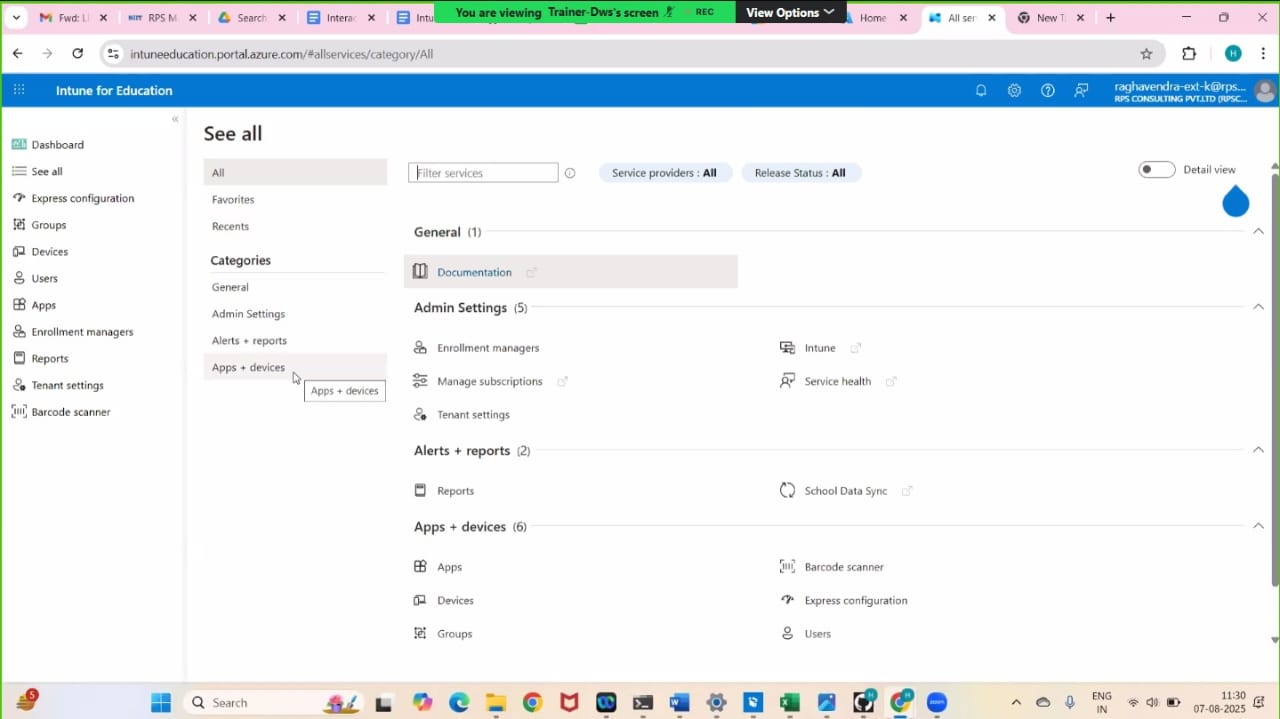


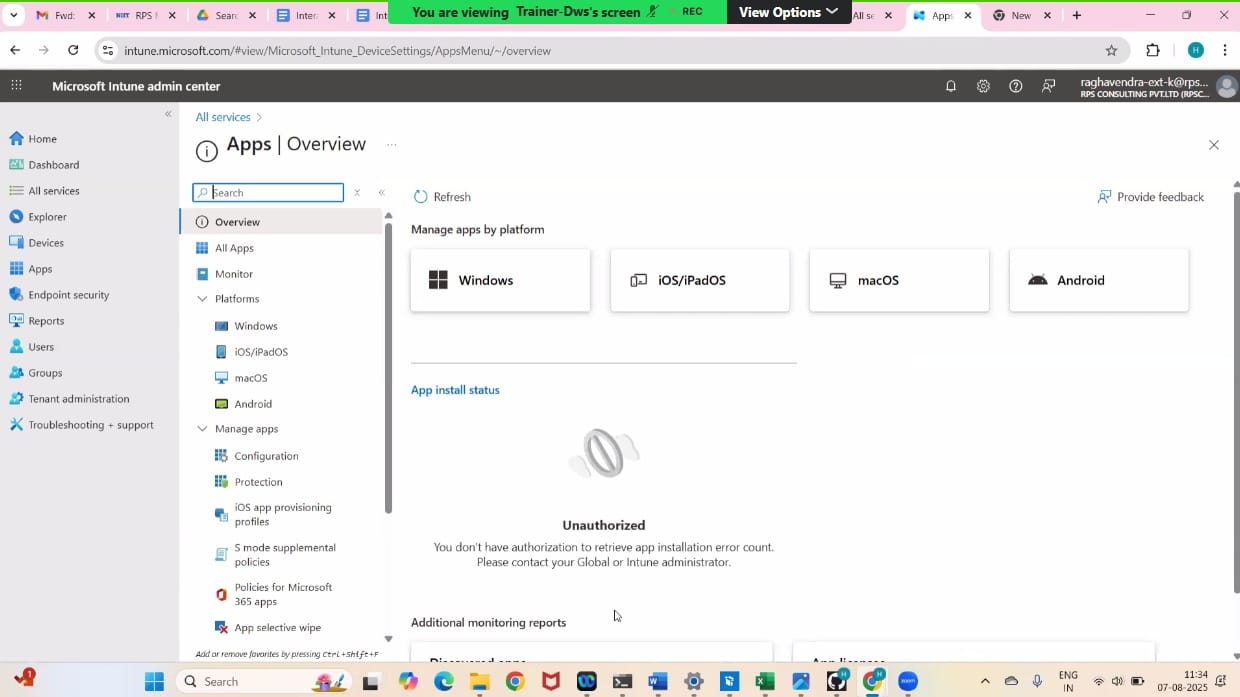
4. **Upload to Intune**

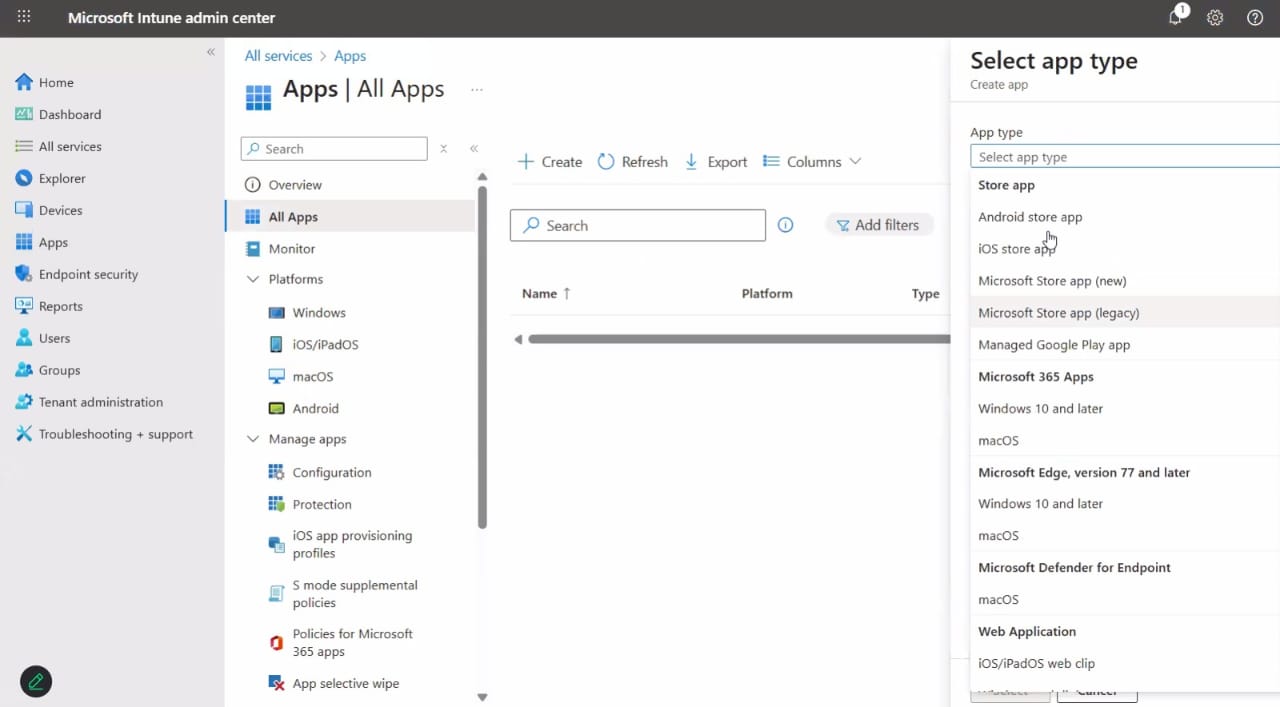
Go to **Intune Admin Center**

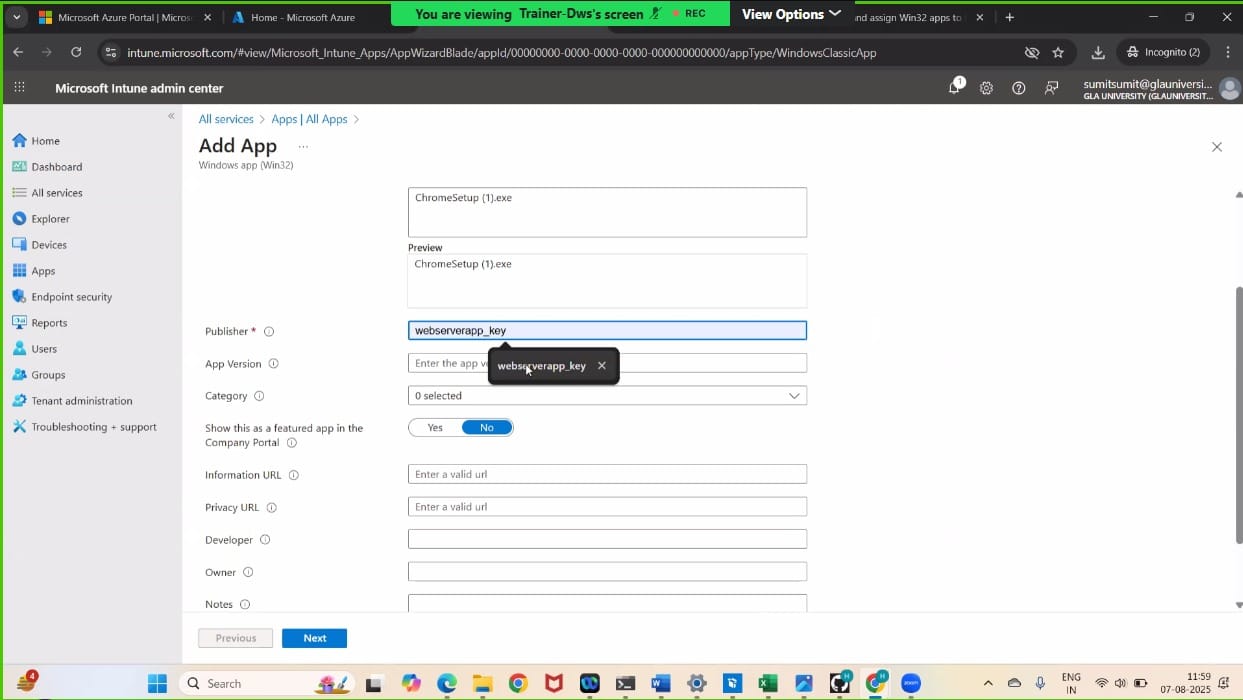
* + Add a new **Windows app (Win32)**
  + Upload the **.intunewin** file
  + Enter install and uninstall commands











**Process Flow for an Application on Windows client via IME service. (From Polling to detection, to installation, to detection and toast notifications as success/failure)**

**Intune App Deployment Flow via IME (Simplified)**

**1. Polling**

* IME service checks Intune servers regularly (default: every 60 mins).
* It looks for new or updated apps assigned to the device.
* Keeps the client in sync with Intune.

**2. Detection**

* Before install:  
  IME runs detection rules (file, registry, process, etc.) to check if the app is already installed.
* After install:  
  Detection runs again to confirm if the app was installed successfully.

**3.Installation**

* If the app is not found:
  + .intunewin file is downloaded and unpacked.
  + Install command is executed silently (e.g., using msiexec).
  + If install takes too long, it's marked as failed.

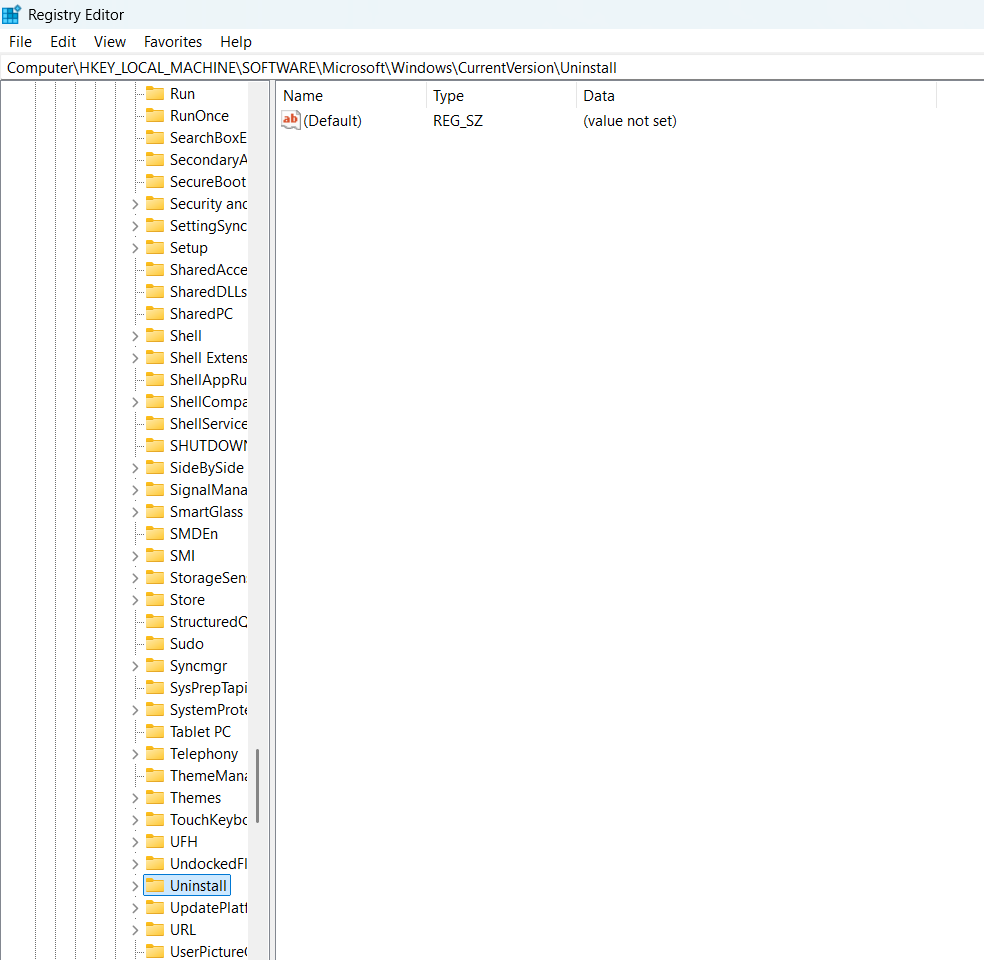
**4.Detection & Notifications**

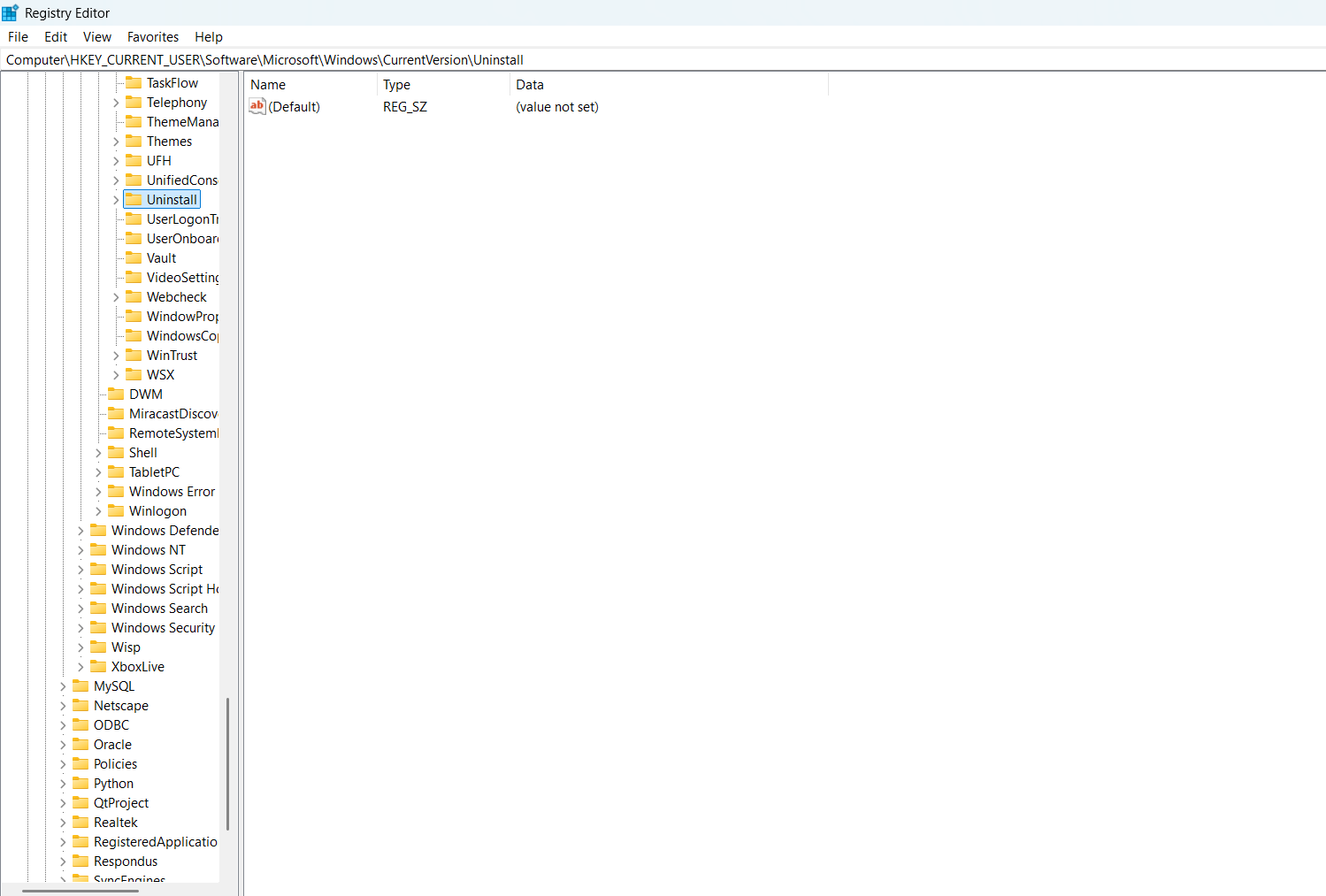
* Post-install detection confirms success/failure.
* Toast notification appears:
  + App installed successfully
  + App failed to install
* Notifications can include text, icons, and actions.
* Device may restart if required by the app or policy.

**Registries with respect to LOB and Win32Apps**

1. **Registry Keys for Uninstall Information:**

* HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall: This key stores information about programs installed for all users on the machine.



* HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall: This key stores information about programs installed specifically for the current user. 

**Finding the GUID (Product Code):**

* Within each of the uninstall keys, you'll find subkeys representing individual applications.
* The subkey name itself is often the application's display name or a unique identifier.

**Using the GUID for Uninstall:**

* The Windows Installer (MSI) uses the GUID (Product Code) to uninstall an application. [2
* You can use the msiexec.exe /x {Product GUID} /QN command to uninstall an application, where {Product GUID} is the application's GUID.

**Log Files**

Log files store records of activities/events on a system or app, helping in tracking issues, analyzing behavior, and debugging. They typically include:

* **Timestamps:**  
  Shows the exact date and time when an event happened – helps in tracing event order.
* **Event Types:**  
  Classifies events like **Error, Warning, Information, Success/Failure Audit** – makes it easy to understand the event's purpose.
* **Severity Levels:**  
  Indicates how serious an event is – like **Critical, Error, Warning, Informational** – helps prioritize what needs attention.
* **Descriptions:**  
  Gives a brief explanation of the event – includes **error codes, affected components, or what the user/system did.**
* **Event IDs:**  
  Each event has a unique ID to quickly search, filter, or identify specific issues.
* **Categories:**  
  Events are grouped into types like **System, Application, Security, or Audit** for better organization and targeted analysis.