**ASSIGNMENT**

Name: Rahul kumar

Batch Start Date: 2025-08-04

Batch Name: WiproNGA\_DWS\_B5\_25VID2550

User ID: 34722

Topic Assignment:

1. Application Packaging
2. Windows 10 Vs Windows 11 basic relevant to App Pack to be discussed
3. Difference between User, Admin, and System Context. in msi
4. Logon Scripts to populate User Profile Data in msi application packaging

Practise activity:-

* Application Packaging:

1. What is Application Packaging?

* It’s the process of converting software into a standardized format (like MSI, MSIX, or App-V) for easy deployment across an organization.
* Why? To manage hundreds of apps efficiently, reduce IT support costs, and ensure security/stability.

2. Key Benefits of Application Packaging

* Consistent and reliable software environments.
* Faster deployments with fewer errors.
* Better security and lower business disruption risks.
* Reduced long-term IT costs.

3. Common Misconceptions

* Packaging doesn’t fix all compatibility issues (some need containerized solutions like App-V or MSIX).
* Tools like Microsoft’s Desktop Analytics (replaced ACT) help diagnose compatibility problems.

4. The End-to-End Packaging Process

Step 1: Application Discovery

* Goal: Gather all app details (source files, configurations, prerequisites).
* Check:
  + Does the app work as expected?
  + Are there different configurations for different teams?
  + Are there prerequisites (e.g., .NET Framework)?
* Tip: Separate prerequisites for reusability (e.g., don’t bundle Java with every app that needs it).

Step 2: Application Packaging

* Create the package (MSI/MSIX/App-V) based on discovery notes.
* Follow best practices (e.g., avoid bloated packages, use silent installs).
* Tools:
  + Free options like Microsoft’s Win32 Content Prep Tool (for Intune).
  + Paid tools like PacKit (automates Intune/MSIX packaging).

Step 3: User Acceptance Testing (UAT)

* Test the package in a VM (mimics real devices).
* Validate all user scenarios (e.g., different departments may use the app differently).
* Golden Rule: Test after importing the package into SCCM/Intune (not before).

Step 4: Deployment

* Use phased rollouts (avoid mass deployment risks).
* For conflicts, rework the package instead of deploying patches separately.
* Prefer Application Model (SCCM/Intune) over Package Model for better features (dependencies, supersedence).

5. Pro Tips for Long-Term Success

* Keep apps updated: Windows 10’s frequent updates (WaaS) can break outdated apps.
* Supersedence: Link new versions to old ones in SCCM/Intune for smooth upgrades.
* Housekeeping:
  + Retire unused apps → Uninstall → Remove from SCCM/Intune.
  + Use naming conventions to stay organized.
* Avoid manual installs: Even for "just a few users," packaging ensures scalability.
* Documentation: Keep detailed discovery/UAT notes for future reference.

6. Advanced: Application Rationalization

* Regularly audit apps to retire/replace/merge redundant software.
* Saves costs (75% of IT budgets go to app maintenance).
* 2. Windows 11 vs. Windows 10 for App Packs:

Why Windows 11 is Better:

1. Sleeker Design:
   * Modern look with rounded corners, redesigned Start Menu, and Taskbar.
   * Feels more intuitive and visually appealing.
2. Stronger Security:
   * Built-in TPM 2.0 and Windows Hello for secure logins (like fingerprint/face recognition).
3. Faster Performance:
   * Quicker logins, app launches, and wake-up times.
   * Smaller, faster updates with less downtime.
4. Better Multitasking:
   * Snap Layouts/Groups let you organize multiple apps easily.
5. Upgraded Microsoft Store:
   * Supports Android apps (via Amazon Appstore) and has a wider app selection.
6. AI Assistant (Copilot):
   * Helps with tasks, boosting productivity.
7. Gaming Boost:
   * Features like DirectStorage and DirectX 12 Ultimate improve graphics and load times.

Why Windows 10 Might Still Be Good:

1. Familiarity:
   * Feels "classic" if you’re used to older Windows versions.
2. Wider Compatibility:
   * Works with almost all older apps and hardware.
3. Stability:
   * Proven reliable for work/school with fewer surprises.
4. Cost:
   * Cheaper for upgrades from very old Windows versions.

Things to Consider for Your App Pack:

* Check App Compatibility: Most apps work on both, but test older ones.
* Performance: Windows 11 is faster, but older apps might lag slightly.
* Security Needs: Windows 11 is safer for sensitive tasks.
* Features: Snap Layouts/Copilot are great for productivity.
* 3. Difference between User, Admin, and System Context. in msi

1. User Context

* What it is: Runs as *you* (the logged-in user).
* Access: Can only tweak *your* files/settings (e.g., app preferences in your profile).
* When to use: Installing apps *just for you* (e.g., a personal browser or game).

2. System Context

* What it is: Runs as the *SYSTEM* account (highest privilege, like "superuser").
* Access: Can change *anything* on the PC (system files, services, all users’ data).
* When to use: Installing system-wide apps (e.g., antivirus, drivers).

3. Admin Context (Not a Separate Mode)

* What it is: Some MSI installs *ask for Admin rights* to do system-wide tasks.
* Access: Needs your *Admin password* to modify shared system resources.
* When to use.
* 4.Logon Scripts to populate User Profile Data in msi application packaging

1. Active Setup in MSI Packages

* What it does: Runs actions (like copying files or updating settings) *when a user logs in*.
* Use case: Perfect for deploying user-specific data (e.g., config files to AppData) after installing an app for all users.
* Example:
  + MSI installs the app for everyone → Active Setup copies personalized settings to each user’s profile at logon.

2. Logon Scripts

* Types: Batch files, PowerShell, or VBScript.
* What they do: Automate tasks during login (e.g., copy files from a server to the user’s profile).
* Assignment:
  + Manually: Attach scripts to individual user accounts.
  + At scale: Deploy via Group Policy (e.g., for entire teams or departments).

3. Deployment Strategies

* Group Policy: Push scripts/MSI packages to users/computers in your Active Directory.
* Script Choice:
  + Simple? Use batch files.
  + Advanced? Use PowerShell (e.g., for complex file operations).

4. Real-World Example

Problem: An app needs user-specific config files in %AppData% at login.  
Solution:

1. MSI Package: Adds an Active Setup entry to trigger a script.
2. Logon Script: Copies files from \\server\settings → %AppData%\MyApp.
3. Deploy: Use Group Policy to roll out the MSI + script.

5. Best Practices

* Error Handling: Scripts should fail gracefully (e.g., if the network is down).
* Security: Avoid hardcoding sensitive data in scripts.
* Testing: Try scripts on a small group before company-wide rollout.
* Documentation: Write down how scripts work (for future troubleshooting).