PL-900 Microsoft Power Platform Fundamentals – Cheat Sheet (2025 Update)

Overview:

The Power Platform is Microsoft's low-code platform for building business solutions – from custom apps and automated workflows to external-facing websites and AI-powered chatbots – **all without extensive coding**. This cheat sheet is updated to align with the **latest PL-900 exam** (post-June 2025 changes), which focuses on **Power Apps, Power Automate, Power Pages, Microsoft Dataverse, connectors, and Copilot AI features**. (Note: **Power BI and AI Builder are no longer covered on PL-900** – Power BI now has its own certification (PL-300) and AI Builder has been de-emphasized in favor of new Copilot features.)

Each of these works in tandem. **Microsoft Dataverse** and **connectors** form the foundation: Dataverse provides a central, secure data store for your apps/bots, while connectors link the Power Platform to hundreds of other services (Microsoft 365, third-party apps, databases, etc.). The integration of these tools enables makers to build end-to-end solutions: for example, a Power Pages site can display data from Dataverse, a Power Automate flow can send emails or write to SharePoint, and a Copilot chatbot can trigger a flow to retrieve data.

Business Value of the Power Platform and Key Concepts

Why Power Platform? It empowers "citizen developers" to solve business problems quickly with minimal code. Things that once required months of traditional development can often be done in days with Power Apps, Power Automate, and related services. This means faster innovation and the ability to tailor solutions exactly to your organization's needs without a full development team.

Core benefits include:

- **Rapid App Development:** Drag-and-drop designers and pre-built templates allow creation of apps, workflows, and sites fast. Businesses can respond to needs (like a new approval process or a customer portal) in a fraction of the time of coding from scratch.
- **Integration of Data and Services:** The Power Platform has **a rich set of connectors (over 900)** to integrate with Microsoft services and third-party systems. Connectors are modular

plugins that let your app or flow *"talk"* to another service's API. For example, there are connectors for SharePoint, Outlook, SQL Server, Twitter, Salesforce, and many more. This makes it easy to pull data from or push data to various sources. **Connectors are key to integrating services and data** – a common exam point is understanding that if you need to connect to (or between) external systems, you'll use connectors, and knowing the difference between **standard vs. premium connectors** (more on that in the licensing section).

- **Microsoft Dataverse (Unified Data):** Dataverse provides a **central data store** that all Power Platform components can use. By having a single source of truth for business entities (like Customers, Orders, etc.), different apps and processes can all share and update the same data seamlessly. This avoids data silos and synchronization headaches. The exam will test your understanding of Dataverse's role and how it compares to a traditional database (spoiler: Dataverse is much more than just tables – it has built-in security, logic, and integration, covered in the next section).
- **Seamless Microsoft 365 Integration:** Solutions built on Power Platform can **work with Microsoft 365 apps and Teams natively**. For example, you can embed a Power Apps canvas app in Microsoft Teams, add a Copilot chatbot to a Teams channel, trigger flows from Excel or Outlook events, or surface a Power Pages site as a tab in Teams. This tight integration means users can access custom solutions in the tools they already use daily.
- **Low Cost and Scalability:** Often, organizations already have some Power Platform usage rights via Office 365 subscriptions (for standard features), reducing the need for large upfront investment. As needs grow, solutions can scale to use more advanced (premium) capabilities. (Licensing specifics are below – know what's free vs. requires premium for the exam).
- **Copilot and AI Integration:** A big recent addition is the integration of **generative AI "copilots"** across the Power Platform. These AI features (powered by large language models) can assist makers in building solutions and end-users in interacting with your apps/bots. For example, Copilot can help build an app or flow from a simple description, or end-users can chat with an AI bot you create to get answers from your Dataverse data. **Understanding the value of Copilot** is now on the exam: it increases productivity by enabling natural language interactions in both the creation and use of solutions. We will cover specific Copilot capabilities under each component (like Copilot in Power Apps, Copilot in Power Automate, etc.) and in the dedicated Copilot section.
- **Together is Better:** The components of Power Platform are designed to work together. Many solutions will use multiple components. For instance, a solution could use Power Apps for the interface, Power Automate for the process logic, Dataverse for data storage, Power Pages to expose data to external users, and a Copilot agent to provide a Q\&A experience – **all integrated**. The exam might ask how Power Platform products can be combined; e.g., *"How can you build a solution that allows customers to check order status (Power Pages site)*

and also ask a question to a chatbot (Copilot agent) that can create a support ticket (Power Automate flow)?" – you'd identify which parts of the platform handle each requirement.

Real-World Business Value Examples:

- A company can **digitize a manual process** (like employee onboarding or expense approvals) by building a Power App (to collect input), automating notifications with Power Automate, and storing data in Dataverse – significantly reducing paperwork and errors.
- An organization can **provide self-service** to customers by creating a Power Pages website where customers can log in to view their account or submit requests, with data secured in Dataverse and the site available 24/7. This reduces load on internal staff.
- A team can **improve decision making** by consolidating data in Dataverse and using Power Platform to automate data updates and alerts. While detailed analytics are now handled in other exams (Power BI), Power Platform can still send simple reports or notifications (e.g., an automated email summary of key metrics via a flow).
- With **Copilot AI**, users can now interact with systems more naturally. For example, instead of searching through an app for info, a user could ask a Copilot chatbot "What is the status of Order 123?" and get an immediate answer drawn from Dataverse. For makers, Copilot can reduce the learning curve by generating an initial app, flow, or page design from plain instructions, accelerating development.

Keep these benefits in mind, as the PL-900 exam often asks about the *"business value"* of Power Platform – essentially **why** an organization would choose Power Platform solutions (the points above) and **what problems it can solve**. If a question is framed as *"How can Power Platform help with X?"* or *"Why use Power Platform for Y?"*, think of these advantages: speed, integration, central data (Dataverse), automation of manual work, extending M365, and AI-driven improvements.

Managing the Power Platform Environment (Dataverse and Governance)

One of the newer emphases in the PL-900 exam is understanding the platform's foundation: **Microsoft Dataverse** and the **Power Platform environment** in which solutions reside, including basic governance and security concepts. Let's unpack these:

Microsoft Dataverse – The Central Data Platform

Definition: Microsoft Dataverse is a secure, cloud-based data platform that stores and manages business data used by all Power Platform components. It organizes data into **tables** (formerly called entities), which have **columns** (fields) and hold **rows** (records) – similar to a database or Excel tables. Dataverse comes with a set of **standard tables** for common business concepts (such as Accounts, Contacts, Cases – aligning with the Common Data Model) and allows creation of unlimited custom tables for specific needs.

More Than a Database: A key concept is that Dataverse provides a lot of out-of-the-box functionality that a traditional database would not. It's often said that Dataverse is **"not just a database, but a platform"**. In a traditional app development, you'd need to set up the database, write code for data access and security, implement business logic, and create a user interface. Dataverse simplifies much of that because it includes:

- **Security Model:** Built-in role-based security and row-level security. Data in Dataverse can be owned by users or teams. Administrators can define roles that permit read/write on specific tables or even specific rows. For example, you can easily restrict users to see only their own records or their business unit's records, something you'd have to custom-build in a SQL database. Dataverse also supports column-level security (masking sensitive fields) and field-level auditing.
- **Rich Metadata & Logic:** You can define relationships between tables (one-to-many, many-to-many) visually. You can also implement business rules (logic that enforces conditions or sets default values when data is entered) with a no-code interface. There are also calculated and rollup fields (which auto-calc values based on formulas or related records) and synchronous workflows/business process flows for guided steps – all without writing code.
- **Automations & Extensibility:** Dataverse works with Power Automate for process automation (and classic Dataverse workflows for background processes). It also allows advanced development: pro developers can write plug-ins (code that runs on data events) or custom APIs if needed, but the exam likely won't go deep here. Just know that Dataverse can be extended with code for complex scenarios, though it covers most needs with no-code features.
- **Integration & API Access:** Everything in Dataverse is accessible via web API and SDK. That means external applications (or Power Platform components via the Dataverse connector) can read/write Dataverse data easily. The platform automatically handles details like concurrency, transactions, and API endpoints. If you change the data schema (add a new table/column), those are immediately available through the API and connectors – no additional dev work. This is a huge advantage: with a traditional DB, you'd need to build your

own API or data access layer; with Dataverse, it's provided. The exam may not quiz the API, but it might ask *how Power Platform connects to Dataverse* – answer: via the Dataverse connector (which uses those under-the-hood APIs).

- **Data Management Features:** Importing and exporting data is user-friendly. You can import data from Excel or CSV through a guided process. There's also **Dataflows** (Power Query) to perform ETL from various sources into Dataverse. Dataverse ensures data quality with features like duplicate detection rules and supports **auditing** (tracking changes to data for compliance).
- **Offline and Mobility:** Apps built on Dataverse (especially model-driven or via Power Apps mobile) can have offline capabilities, because Dataverse can cache data on device and sync later. This is important for scenarios like field workers with poor connectivity.
- **Common Data Model (CDM):** The standard tables in Dataverse follow the CDM, which is a set of common definitions (for e.g., Account, Contact, Product). Knowing this isn't heavily tested, but it's a selling point: if two apps use Dataverse's standard tables, they inherently share meaning (like Dynamics 365 Sales and a custom app both using the Account table).
- **Dataverse for Teams vs. Dataverse:** There are two "flavors" – **Dataverse for Teams** is a lightweight version included with Microsoft 365 licenses, which allows building Power Apps/Flows inside Teams with a Dataverse-backed data store (with some limitations in table count, capacity, and missing advanced features). **Full Dataverse** (in a normal environment) requires premium licensing but offers the full capabilities. For PL-900, just remember: Dataverse for Teams = included with Teams, but only usable within the Teams interface for smaller scale apps; full Dataverse = the real deal for enterprise apps (and requires Power Apps/Automate premium licenses). Generally, exam questions that involve Dataverse outside of Teams assume premium licensing is needed.

Dataverse vs Traditional Database: You might be asked why use Dataverse instead of, say, a SQL database or SharePoint list. Key points:

- *Dataverse has built-in enterprise features* – **security, auditing, form UI (model-driven apps), integration** – out-of-the-box. A SQL database is just raw data storage; you'd need separate tools to add those capabilities.
- *Dataverse is low-code friendly* – Makers can define tables and relationships with clicks (no complex DDL scripts), and use the data immediately in Power Apps or Power Automate. Also, Power FX formula language can be used in calculated columns or canvas apps to manipulate Dataverse data easily.
- *Integration* – Dataverse is natively integrated into Power Platform. E.g., to build a Power App on SQL, you need to set up a connection and work out delegation for large data, etc., but a Power App on Dataverse can leverage things like direct filtering (via views) and option sets

easily. Connectors exist for SQL too, but Dataverse is "first-class" in the platform. There's also one less moving part (no on-prem gateway needed since Dataverse is cloud).

- *When might you choose something else?* If you have extremely large volumes of data or very complex queries (beyond Dataverse limits), an Azure SQL or Data Lake might be more appropriate. But for most business app scenarios, Dataverse's limits (which are quite high for normal use) and features make it the best integrated choice. The exam may not go deep here, but recognizing that *Dataverse is designed for business application data and ease of use*, whereas a traditional DB gives more manual control (and complexity), could be useful if asked.

Using Dataverse in Power Platform solutions:

- Power Apps **model-driven apps** require Dataverse – they are essentially a UI on top of Dataverse tables.
- Power Apps **canvas apps** can use Dataverse as one of many possible data sources (via the Dataverse connector) if you need those enterprise data capabilities.
- Power Pages sites typically use Dataverse to store and retrieve web content and user-submitted data (you can also expose Dataverse forms and views directly on a site).
- Power Automate flows have triggers/actions for Dataverse (to perform operations when records change or to create/update records).
- Copilot/Virtual Agents use Dataverse both as knowledge source (you might store Q\&A or docs in Dataverse) and to take actions (bots creating or querying records via flows).

Dataverse and Exam Tips:

- Learn the basic **terminology**: tables (entities), columns (fields), rows (records), relationships. A question might ask, for example, how to model a one-to-many relationship (answer: in Dataverse, by creating a Lookup column on the "many" side that links to the "one" side table).
- The exam may ask about **business rules or processes** in Dataverse. Know that you can do things like enforce that a date must be in the future (business rule) or require a progression of stages (business process flow) without code.
- Understand **environments** (coming next) because Dataverse is created per environment.
- *Differences vs other data options:* e.g., SharePoint lists are easy but lack relational structure and have delegation limits; SQL is powerful but requires more expertise; Dataverse is a middle ground giving near-SQL like capabilities plus lots of extras without heavy coding.

Power Platform Environments and Governance

Environment Basics: An *environment* is like a container or space in which you store, manage, and share your Power Platform solutions (apps, flows, pages, chatbots, and their data). Each environment can have **one Dataverse database** associated with it (or none, for environments that only contain, say, Power Automate flows with no Dataverse). You can think of environments as separate workspaces – often used to separate development, testing, and production or to separate different departments' solutions. For example, you might have a "HR Apps" environment and a "Finance Apps" environment to keep their apps and data segregated.

- **Default Environment:** Every tenant has a default environment where all users with Power Platform can create apps/flows (unless restricted). By default, everyone is a Maker in default. Microsoft now encourages organizations to limit building in the default environment for production, and instead create specific environments to better manage and govern (this new exam focus on governance hints at this best practice).
- **Environment Roles:** Every environment has **Admins** and **Makers**. Environment Admins can manage the environment (add users, create Dataverse, set DLP policies for that environment, etc.), while Environment Makers can create resources (apps, flows, etc.) in it. A **Power Platform Admin** or global tenant admin can create and manage all environments via the admin center.
- **Admin Centers:** There are specialized admin portals for different parts:
    - The **Power Platform Admin Center** is the main one for managing environments, Data Policies (DLP), monitoring resource usage (like flow runs, Dataverse storage), creating data integration projects, etc..
    - There are also product-specific admin or maker portals: e.g., the Power Apps maker portal (make.powerapps.com) which also includes flow management, the Power Automate portal (flow.microsoft.com, though it redirects to make.powerapps.com for unified experience now), the Power Pages design studio (portal itself for building pages), and Copilot Studio portal for bots. The exam might expect you to know of these at a high level (for instance, that you build Power Pages sites in the Power Pages design studio, not in the Power Apps maker portal).
    - Microsoft Teams also has integration: if using Dataverse for Teams, the environment is actually tied to a Team and you build apps/bots directly within Teams (via the Power Apps or PVA app in Teams).
- **Data Loss Prevention (DLP) Policies:** These are an important governance tool. A DLP policy can be set at environment or tenant level to control which connectors can be used together in apps/flows. Connectors are classified as *Business* (trusted for sensitive data) or *Non-*

Business, or *Blocked*. A typical policy might say "Any connector tagged Non-Business cannot be used in the same app/flow with any connector tagged Business" – to prevent data from, say, your secure systems (SQL, SharePoint) being accidentally sent to social media or personal email. For example, you might block a flow that tries to take data from Dataverse and post it to Twitter if Twitter is marked Non-Business. The exam might give a scenario like *"IT wants to ensure no one can create a flow that shares company data on social media. What feature would you use?"* – answer: a DLP policy.

- **Environment Lifecycle and ALM:** Large organizations might implement Application Lifecycle Management by using multiple environments (Dev → Test → Prod). Solutions (a packaging of components) can be used to move apps/flows from one environment to another in a managed way. PL-900 won't dive deep into solutions, but it might mention that solutions exist to export/import apps and their assets between environments as part of good governance.
- **Security and User Access:** Apart from environment roles, if Dataverse is used, there's an extra security layer – **Dataverse security roles**. Users must have a Dataverse security role (like Basic User, or custom roles) to access data or the model-driven apps in that environment. If a user is just consuming a canvas app that uses Dataverse, behind the scenes they still need at least basic Dataverse read on that table or the app has to run under a service principal or something (beyond PL-900 scope). For the exam, remember that **security exists at both environment level (who can create or administer) and at data level (who can see or edit records)**. Administrators can leverage Azure AD groups with these roles to manage access at scale.
- **Data Privacy and Compliance:** Microsoft Power Platform is built on Azure, so it inherits a lot of compliance certifications (like GDPR, ISO, SOC, etc.). Power Pages, for example, runs on Azure App Services and supports modern encryption and threat protection measures. For the exam, you should know that using Power Platform can help meet privacy requirements because you can *restrict access to data via roles*, *track/audit usage*, and *host data in specific regions*. Accessibility is another aspect: the platform's components (Power Apps, Power Pages) have features to build accessible apps (like screen reader support, responsive design, proper semantic HTML in Pages) – aligning with accessibility guidelines is part of good solution design, though PL-900 doesn't test deep on it, just acknowledges it as a consideration.
- **Monitoring and Analytics:** Admins can monitor usage of apps and flows (there's a Power Platform Analytics section in the admin center that shows things like how many times an app is launched, flow run success/failure, etc.). This helps govern by spotting problematic apps or high resource usage. Also, the **Power Platform Center of Excellence (CoE) Starter Kit** is a set of tools Microsoft provides (in GitHub) to help organizations track and manage Power

Platform adoption (contains Power BI dashboards, workflows for approvals, etc.). While PL-900 won't require knowing the CoE kit, being aware of governance tools like that or basic analytics in admin center might help answer a conceptual question about "how do you track Power Platform usage or enforce governance?".

TL;DR for Governance: Microsoft wants new Power Platform practitioners to appreciate that creating solutions isn't just about clicking together an app – one must also ensure those solutions are built in a managed environment with proper security and data policies from day one. So expect a few questions around *where* you build (environments), *who* can see/use the solutions or data (security roles), and *keeping data safe* (DLP policies). For example:

- "What is the purpose of a DLP policy?" → To prevent corporate data from being mishandled by restricting which connectors can be used together.
- "What are environments used for?" → To compartmentalize apps and flows, often by department or lifecycle, and to host a Dataverse for those apps.
- "Who can create resources in an environment?" → Environment Makers (by default all in default env, or specific users added elsewhere).
- "How does Power Platform enforce security of data?" → Via Dataverse security roles and environment access control, and integration with Azure AD for authentication.

Power Apps – Building Custom Applications (25–30% of exam)

What is Power Apps? Power Apps is a suite within the Power Platform to build **custom business applications** with a graphical interface and low-code logic. These apps can run on the web or on mobile devices (via the Power Apps mobile app). Power Apps enables users to create two main types of apps: **canvas apps** and **model-driven apps** (and a third type, Power Pages for web sites, is now its own product). There's also the concept of **Power Apps Portals**, now rebranded as Power Pages for external sites, which we will cover separately. Power Apps is about rapidly delivering the UI and business logic for specific tasks or workflows.

Canvas Apps vs Model-Driven Apps: This is a critical distinction:

- **Canvas Apps:** Think *"blank canvas"*. You design the app's interface by dragging and dropping controls (text boxes, buttons, galleries, forms, images, etc.) onto screens, much like designing a slide in PowerPoint. You have full control over the app's layout and style. You write Excel-like formulas (Power Fx) to handle interactivity – for example, to filter data for a gallery (`Filter(Accounts, City="Wellington")`) or define what happens when a button

is clicked (`OnSelect` property). Canvas apps can connect to a variety of data sources (over 300 connectors, including SharePoint, Excel, SQL, Dataverse, etc.). They are best for highly customized UX or when you want to start from a particular data source (e.g., generate a quick app from an Excel or SharePoint list). Canvas apps can be designed for **phone** or **tablet** (or responsive to different screen sizes). They are often embedded in Teams or SharePoint, or run standalone via the Power Apps mobile player or browser.

- **Model-Driven Apps:** These apps are driven by the data model and processes in Dataverse. Rather than pixel-perfect design, you primarily configure the underlying Dataverse tables, forms, views, and business process flows, and Power Apps will render a standardized user interface for you. Model-driven apps have a uniform look (header, navigation, lists of records, forms to edit, all auto-generated based on your Dataverse schema). You don't worry about placement of each field – you design forms (which fields in what sections) and views (which columns and filters) in the Dataverse schema editor, and the app uses those. Model-driven apps are ideal for **data-centric applications** that follow standard UI patterns (like a CRM system: you have menus for Accounts, Contacts; you click an account to open its form; you have related records, etc.). They **require Dataverse** as the data source (you can only use Dataverse tables in model-driven apps). A big benefit is they come with capabilities out-of-the-box: *responsive design* automatically, *rich filtering and sorting*, *export to Excel*, etc., without extra work.

Key capabilities of Power Apps (common to both types):

- **Low-Code Logic with Power Fx:** Both canvas and model-driven can use formulas and expressions. Canvas apps use Power Fx extensively (similar to Excel formulas or JavaScript) for actions, dynamic properties, validation, etc.. Model-driven apps allow some calculated fields and business rules without code.
- **Connectors:** Canvas apps can use connectors to bring in data. Model-driven apps mainly use Dataverse (and you could use virtual tables to show external data as if it were Dataverse).
- **Integrations:** Canvas apps can embed Power BI visuals, call Power Automate flows (via the Flow button or a formula action), and even include AI components (like scanning business cards, text recognition, etc., from AI Builder). Model-driven apps can also trigger flows (there's an integration to add a flow button in the command bar).
- **Forms and Controls:** Canvas apps have a rich gallery of control types (e.g., drop-downs, sliders, date pickers, pen input for signatures, etc.), and you can import more (canvas components or PCF controls for code components). Model-driven apps use Dataverse forms – you place fields (including option sets, lookups, subgrids for related lists) on the form. Model apps also support embedded charts and dashboards configured in Dataverse.

- **Sharing and Publishing:** To share a canvas app, you publish it and share with users or security groups (they can access via Power Apps site or Teams if integrated). Model-driven apps are shared by giving users security roles that include access; they then see it in the Dynamics 365-style app menu. The exam expects you to know that **you can share Power Apps with others in your org easily** (provided those users have appropriate license and data permission). There is no app store for external users—Power Apps are primarily for internal use (unless it's a Power Pages site for external).
- **Templates:** Canvas apps have some templates (like "Expense Tracker app") to kickstart. More commonly, you might auto-generate a canvas app from data (e.g., connect to a SharePoint list and let Power Apps generate a simple three-screen CRUD app – Browse, Detail, Edit screen). Model-driven apps can be quickly made by selecting which tables to include, and it will generate a basic navigation and forms automatically.
- **Offline Capability:** Canvas apps can be made to work offline (using the `LoadData` and `SaveData` functions to cache data locally and logic to check connectivity). Model-driven apps also have an offline mode (with some setup, using the Power Apps mobile app offline profiles) primarily for Dataverse data. The exam might not go deep on offline, but it's worth knowing canvas apps can work offline with planning.
- **Copilot in Power Apps:** This is new and likely to be on the exam. There are two main aspects:
    - *Copilot for Makers:* Within Power Apps maker portal, Microsoft has introduced Copilot features that allow makers to create an app using natural language. For example, you can type "Create a table for Event Registration with columns Name (text), Event Date (date), Email (text)…" and it will generate a Dataverse table and a quick app for you. Or describe an app and it proposes a starting point (this uses GPT under the covers). The exam outline specifically says *"Create and modify tables and columns through conversations"* and *"Create and modify canvas apps through conversations"* (implied by Copilot controls and such) – meaning they expect you to know you can use conversational AI (Copilot) to build Dataverse schema and apps, **a big productivity win**. If an exam question mentions a feature where you type what you want and the system builds the table or app for you, that's **Power Platform Copilot**.
    - *Copilot control (for end-users):* A new control available in canvas apps is the **Copilot control**. This is essentially a chat interface you can embed in your app that allows the user of the app to ask questions in natural language and get AI-generated answers from your data. For example, if you have a Dataverse table of product inventory, you can add a Copilot chat control connected to that table; an end user could then ask "What products are low on stock?" and it will use GPT to generate an answer by

querying the data. This is a way of adding an AI assistant *inside* your app. The exam likely will expect you to recognize what a Copilot control is – a component that provides an AI-powered Q\&A over your data, to enhance app user experience.

Common Use Cases for Power Apps:

- Replacing paper forms or Excel processes with a mobile app (canvas app) – e.g., safety incident reporting app for on-site workers, an expense submission app, or a visitor check-in app in a lobby.
- Extending or customizing existing systems: e.g., building a front-end over a SQL database to allow employees to perform CRUD operations with a nicer interface; or building a quick app that writes to Dynamics 365 (Dataverse) to handle a process that D365 doesn't directly cover.
- **Canvas vs Model use case example:** For a tailored task app (say a tailored UI for scanning barcodes and updating an inventory record one by one with custom buttons and coloring), canvas is perfect. For a more data-heavy solution (like an internal CRM or a trouble ticket system) where you want standard grids and forms and you have relationships among many tables, model-driven might be faster to configure.
- **Personal productivity apps:** Many business users build small apps for their own team – e.g., a volunteer sign-up app, or a tool to track internal assets – often these start in the default environment with SharePoint as data, but as maturity grows, they move to Dataverse in a proper environment.
- **Forms within Teams:** Using Power Apps integrated in Teams (which uses Dataverse for Teams), people build simple apps like a FAQ resource catalog, IT support request form, etc., accessible to everyone in that Team. This has become popular since it's included in Office licenses and directly in the Teams context.

Exam Tips for Power Apps:

- Make sure you know the **difference between canvas and model-driven apps**. This is frequently tested. For example, a question might describe an app scenario and you have to choose which type fits. Clues:
  - If custom UI or multiple data sources or very specific layout = **Canvas app**.
  - If leveraging Dataverse with standard forms, need responsive out-of-box and maybe multi-step business process flows = **Model-driven app**.
  - If external users need access = **Power Pages**, not Power Apps (internal apps are Power Apps; external websites are Power Pages).
- Understand the basics of **Power Apps Studio** (for canvas) vs. the app designer (for model-driven). And that model-driven apps are built by modeling data in Dataverse first.

- **Controls & Properties:** Know some common controls (gallery, form, text input, label, button) in canvas apps and that formulas can dynamically change their properties. E.g., setting a button `Visible` property to `User().Email = CreatedByEmail` can show it only to the creator of a record – illustrates conditional visibility with no code.
- **Data sources and connectors:** Recognize if a scenario is using a premium connector. For instance, *"build an app for employees to browse a SQL Server inventory database"* – SQL Server is a premium connector, meaning the users will need Power Apps licenses (unless it's Azure SQL which is also premium anyway). The exam might indirectly test licensing via scenario: standard connectors (SharePoint, Excel, Teams, Outlook) vs premium (Salesforce, SAP, on-prem SQL).
- **Limits of Power Apps:** If asked what Power Apps cannot do: it's not for complex public websites (that's Power Pages), it's not for heavy coding (though you can extend with custom connectors or code components if absolutely needed), and it's not a reporting tool like Power BI (though you can show basic charts). Also, Power Apps is not the right tool for scheduled automation without user interaction (that's where Power Automate comes in).
- **Copilot related:** A question might be, *"How can a maker quickly create an app by describing requirements in natural language?"* Answer: By using the Power Apps Copilot feature (the AI-assisted app builder). Or *"What is the Copilot control used for in a canvas app?"* Answer: To provide an AI chat experience to app users for querying or interacting with the app's data.
- **Publishing & Sharing:** Remember that after building, you **publish** the app (which makes the latest version available to users) and **share** it by assigning user or group access. If a user can't see an app, either it's not shared with them or they lack license or Dataverse permissions. PL-900 might ask something like how to share an app with colleagues – straightforward: use the share function in Power Apps and ensure those users have the necessary license.
- **Power Apps and Teams:** If the scenario is specifically about an app in Teams and want no extra licensing: use **Dataverse for Teams** – build the app using the Power Apps app in Teams, which stores data in a special Dataverse for Teams environment (no additional cost). But note the limitations (max 2GB, etc., not likely in exam detail).
- **Model-Driven specifics:** Know terms like forms, views, dashboards, business process flow in context of model-driven apps. E.g., an exam question: *"In a model-driven app, how do you control which columns of a table are displayed in a list view?"* – by editing the View in Dataverse for that table. Or *"What feature of model-driven apps helps ensure users follow sequential steps for data entry?"* – Business Process Flow (BPF).
- If the exam mentions **Power Fx**, that is the canvas app formula language (also now expanding to model-driven calculated columns, etc.). Basic understanding: it's like Excel formulas for logic. They may check if you know that Power Fx is used to write logic in Power Apps (and now in other components too as the unified low-code language).

Power Apps is the largest portion of the exam (25–30%). Focus on scenarios, e.g., identifying which type of app or which tool to use, and understanding capabilities like connectors, controls, and Copilot features within Power Apps.

Power Automate – Automating Processes (15–20% of exam)

What is Power Automate? Power Automate is Microsoft's cloud-based **workflow and task automation** service. It allows you to create "flows" that implement business logic: basically, if X happens, do Y and Z, or do A at a scheduled time, etc., across various apps and services. It connects to the same wide array of connectors, enabling integration and automation between both Microsoft and many third-party systems. Power Automate includes capabilities for **cloud flows** and **desktop flows (RPA)**.

Types of Flows:

- **Cloud Flows:** These run in the cloud (in Microsoft's datacenters) and are triggered by some event or schedule. There are three sub-types:
    - *Automated flows* – triggered by an event in a connector. For example, "When a new item is added in SharePoint, then …". The event could be an incoming email, a new row in Dataverse, a tweet with a certain hashtag, etc., as long as a connector provides that trigger.
    - *Instant flows* – triggered manually by a user. This could be via a button in the Power Automate mobile app, a button on a SharePoint or Power BI, or even a selectable flow in some UI (e.g., "Flow" menu in SharePoint lists). These are for on-demand tasks, e.g., a user selects a record and runs a flow to perform an action on it.
    - *Scheduled flows* – run on a schedule (e.g., every day at 8AM). Good for daily data syncs, batching, or periodic checks.
- **Desktop Flows (RPA):** These automate actions on a desktop computer using the Power Automate Desktop application. RPA stands for Robotic Process Automation – essentially, using software "robots" to mimic user actions (clicks, keystrokes) in software that doesn't have easy APIs. Desktop flows can be:
    - *Attended RPA* – runs while a user is logged in and maybe even actively triggers it. For example, an employee could run an attended RPA flow to fetch data from one legacy system by automating the UI, while they watch.

    - *Unattended RPA* – runs in the background on a machine without user intervention (requires a special license). This is for automating processes fully hands-free, often scheduled or triggered from a cloud flow.
- **Business Process Flows (BPFs):** Not to confuse with the above, BPFs are a Dataverse/Power Apps feature to guide users through stages in model-driven apps. They are configured in Power Automate (old interface) but are more of a UI guidance than automation. The exam might mention BPFs when talking about model-driven app capabilities. Just recall: BPF = guided process bar (e.g., stages "New -> Under Review -> Approved" for a record).

Triggers and Actions: A flow is composed of a **trigger** (the event that starts the flow) and **actions** (the steps the flow takes). For example, in an automated flow triggered by "When a new lead is created in Dataverse," actions could be: "Send an email to the sales manager" and "Post a Teams message to #Leads channel." Triggers can also be a schedule or manual button as noted. Actions can include conditions (if/else), loops (apply to each item in a list), and even parallel branches. You can have a flow wait for an approval and then continue based on response. **Exam-wise**, know that triggers come from connectors (or built-in schedule, manual) and actions often involve connectors too. A question might ask what trigger to use for a scenario: e.g., to run a flow every Monday = Schedule trigger; to run when a file is added to OneDrive = OneDrive connector's "When a file is created" trigger.

Connectors in Power Automate: Power Automate uses the same connector library as Power Apps. Popular ones: Office 365 Outlook (send email, calendar events), SharePoint (list items/documents), Microsoft Teams (posting messages, etc.), Forms (when a form is submitted), Approvals (special connector for sending approval requests), etc. Also third-party like Twitter, SQL, DocuSign, etc. **Standard vs Premium** matters here too: e.g., "When an HTTP request is received" trigger or the SQL Server connector are premium, so requiring a premium license. If a question describes integrating with a 3rd party or on-prem system, likely a premium connector scenario.

Approvals: Power Automate has a built-in approval system, which is a common use case to highlight. An **Approval action** can send an approval request to 1 or more people (they get a notification in email and in Teams or the Power Automate app) and it tracks their response (Approve/Reject plus comments). The flow can then act on the outcome (if approved, do X, if rejected, do Y). For example, an expense report approval flow: once submitted (trigger), send manager approval. If approved, update SharePoint list status and send email; if rejected, notify the employee with reasons. The exam may present a scenario that basically describes an approval workflow and ask which Power Platform tool or feature would fulfill it – answer: **Power Automate's Approval actions** (part of cloud flows).

Integration with other Power Platform:

- A **canvas app** can trigger a flow (via the Flow action in Power Fx or a Flow button control) to perform tasks that the app alone can't (like send an email or update multiple records).
- A **Power BI** (though not in exam now) dashboard tile can trigger a flow via a button.
- A **Power Pages** site can use Power Automate behind the scenes for form submission logic or to retrieve data via an API call, etc.
- **Copilot/Virtual Agents** definitely use flows: if a chatbot needs to perform an action (like create a record or retrieve info that requires a complex query), PVA/Copilot calls a Power Automate flow (formerly known as Bot Framework Skills in PVA).
- **Dynamics 365** uses a lot of Power Automate under the hood for new cloud flows (replacing many classic workflows).

Process Mining: This is new on the exam (it was explicitly mentioned in the skills outline). Process Mining in Power Automate (also called Process Advisor) is a capability where you can **analyze the logs of business processes** to identify inefficiencies or bottlenecks. You basically record or import data about how processes occur (e.g., logs from an ERP of how long each step takes for different instances) and Process Mining generates process maps and insights – helping you understand where automation can help or how to optimize. In fundamentals, just know that *Process Mining is a Power Automate feature that helps discover and improve processes by analyzing data*, but it is more of a visualization/analysis tool than an automation itself. They might ask, *"Which Power Platform feature can analyze an organization's business process flows to suggest optimizations without manually mapping them?"* – answer: **Process Mining in Power Automate**.

Licensing for flows: We'll detail in the table, but recall:

- If only standard connectors are used and user has Office 365, they likely can create the flow without extra license.
- Premium connectors or using unattended RPA requires premium license (Power Automate per user or per flow, or RPA add-on).
- There is also a limit on runs: e.g., 5,000 API calls per day per user for a per user plan (just a factoid; fundamentals probably won't test those numeric limits but know there are limits).
- Attended RPA is free for Windows 10/11 users (which includes Power Automate Desktop for personal tasks), but orchestrating RPA from cloud flows needs premium.

Common Use Cases for Power Automate:

- **Email notifications and alerts:** e.g., instantly notify via email/Teams when an important SharePoint file is updated or when a Forms survey is submitted.

- **Data synchronization/integration:** e.g., when a new customer is created in CRM (Dataverse), automatically create a corresponding folder in SharePoint, or copy the data to an Azure SQL database. This could be a one-way or two-way sync between systems.
- **Approvals:** as discussed, automated approval workflows for vacation requests, purchase orders, publishing content, etc., replacing email back-and-forth.
- **Scheduled reports or tasks:** e.g., every Monday generate an Excel report of pending tasks (by querying Dataverse or SharePoint lists) and email it out. Or nightly, move data from one system to an archive.
- **Managing Office 365 things:** e.g., a flow that runs daily to find overdue tasks and post a reminder in Teams; or when an email with an attachment arrives from a VIP, automatically save the attachment to OneDrive and alert someone.
- **Robotic Process Automation:** e.g., automating legacy software:
    - A desktop flow could take data from an Excel file and input it into an older application (that doesn't have an API) by controlling mouse and keyboard.
    - Or scraping information from a website if no API is available.
- **Forms processing (with AI builder, though AI builder not emphasized now):** e.g., use AI to read invoices and then a flow to enter the data into a database. (This was a common scenario but since AI Builder is deprioritized on exam, maybe less focus, but know it's an example of extending flows with AI.)

Exam Tips for Power Automate:

- If you see a scenario describing *any repetitive task or multi-step process across apps*, especially involving logic (conditions) or waiting (approvals), think **Power Automate**. Many questions boil down to "Which Power Platform tool would you use for X?" and if X is automation, the answer is Power Automate.
- Know the difference between **cloud flow** and **desktop flow** scenarios. If the process involves older UI or no connectors (like "the process includes copying info from a legacy mainframe UI"), that's a hint for **RPA (desktop flows)**. If it's all cloud apps with APIs, then a normal cloud flow.
- **Trigger types:** They might test awareness of triggers. For example, *"How can a flow be triggered?"* Options might include: a schedule, an event (like an item created), a button, etc. The correct answer would be that flows can start automatically on events, manually, or on a schedule.
- **Actions and Logic:** Perhaps a question on how to handle repetitive actions (answer: use an "Apply to each" loop) or how to branch (condition control). You don't need to memorize the names of all actions, but recognize common ones (Send an email, Create item, Update

record, etc.). And know you can call one flow from another (child flows) in advanced cases (maybe out of scope).

- **Approvals:** As mentioned, if scenario is approvals, explicitly mention using the approval action in a flow.
- **Limits of Power Automate vs Logic Apps:** Unlikely to get into PL-900, but just in case: Power Automate is meant for user-friendly, individual or small-team automation; Azure Logic Apps is a very similar engine for enterprise integration (code-heavy scenarios). If the question is about a citizen developer scenario, Power Automate is the answer rather than Logic Apps.
- **Integration in questions:** For example, a question might describe, *"Employee submits a form in a Power App, then manager gets an approval request, then data is saved to SharePoint"*. Identify that the **approval and data save** part is done by a **Power Automate flow triggered from the Power App** (the Power App itself would just start the flow).
- **Monitoring:** Know that you can check if a flow succeeded or failed in the Power Automate portal (Run history). Also you can turn flows on/off. If a flow is failing, an admin or owner can see the error details in run history or use the repair tips provided.
- **Templates:** There are many templates (pre-built flows) available. The exam might not quiz specific template names, but just know they exist as a starting point library.
- **Process Advisor (Process Mining):** If they mention analyzing how a process works from event logs or recordings to recommend optimizations, that's Process Mining. They likely won't expect deep knowledge, just identification of what it's for.

Remember that **Power Automate often works behind the scenes** – so some exam scenarios may not mention it by name but describe a background process → that's your clue.

Power Pages – Low-Code Business Websites (10–15% of exam)

What is Power Pages? Power Pages is a low-code software-as-a-service for creating **secure, data-driven websites** for external or internal users. It was formerly the Power Apps Portals capability, now a standalone product in the Power Platform family. With Power Pages, you can build websites that allow users (like customers, partners, or employees) to sign in and view or submit data via web pages, forms, and lists. It's essentially giving you the ability to stand up a web portal without writing traditional web code (HTML/JS/CSS) – though you can extend with code if needed.

Key capabilities of Power Pages:

- **Design Studio with Workspaces:** Power Pages provides a design studio (a progressive web app in the browser) for building and configuring your site. It has five main work areas:
    - **Pages** – where you create and arrange web pages and content. You add text, images, sections, and configure basic UI layout. Think of it as a web page builder.
    - **Styling** – where you define the look and feel: themes, colors, fonts, and other styling aspects to match your brand.
    - **Data** – this connects to Dataverse. Here you can view and create the Dataverse tables that your site will use. You also set up **forms** and **lists (views)** that will display data on the site. (E.g., create a form for "Job Application" table, or a list of "Open Positions" that shows records on a page).
    - **Set up** – for site settings, authentication, identity providers, and general configuration (like site URL, custom domains, languages, etc.).
    - **Security** – for managing site users, roles, and permissions, and reviewing any potential security issues (like a checklist of best practices, vulnerability scanning). This structured approach is important: the exam might ask "In which workspace of Power Pages would you configure which users can see specific data on the site?" – that would be the **Security workspace** (for web roles and table permissions).
- **Templates & Rapid Start:** Power Pages offers several **site templates** for typical scenarios (customer self-service, partner portal, community forum, etc.) as well as a default blank template. Templates come with prebuilt pages and Dataverse schema to get you started. You can also start from scratch. A new feature is **Copilot in Power Pages**: you can use an AI assistant to generate text for your site or even draft page layouts. For example, you can ask Copilot "Create a hero section with a welcome message and a registration form" – it will try to build that. The exam objective *"Create and modify Power Pages sites by using Copilot"* suggests you should know Copilot can help build pages or even entire sites from natural language prompts.
- **Secure Data Access:** Power Pages is built on Dataverse, so you leverage Dataverse's security infrastructure for your site. This includes:
    - **Authentication**: Power Pages supports several identity providers out-of-the-box: Azure Active Directory (for enterprise users, including Azure AD B2C for external), Microsoft Accounts, LinkedIn, Google, or even any OpenID Connect provider. Or you can allow anonymous access for public content. So exam might ask: how can external users log into a Power Pages site? – by setting up an identity provider like Azure AD B2C, or allow anonymous if appropriate.
    - **Web Roles**: You define roles in Power Pages (e.g., "Authenticated Users", "Partners", "Administrators") and assign these roles to users (contacts in Dataverse). Roles determine what the user can do or see on the site.

    - **Table Permissions**: This is crucial – by default, even an authenticated user cannot see any Dataverse data on the site until you explicitly give permission. Table Permissions tie a web role to specific table access (with CRUD permissions and even filters). For example, you can say Role "Partner" has permission to Read and Create on table "Order", but only where the Order's Account is the logged-in user's account (owner-based rule). This ensures each user sees only their data. The exam objective *"Describe Power Pages security concepts"* means you should understand that implementing a portal requires configuring *web roles and table permissions* to enforce data security.
    - **Compliance and Protection**: As mentioned earlier, because it runs on Azure, it's compliant with major standards (the site is essentially an Azure web app behind the scenes). It has features like enabling SSL (HTTPS), cookie consent, and you can integrate with solutions like Azure Front Door or CDN for performance and extra security.
- **Integration with Dataverse components:** One of the powerful features is that you can surface **forms and views** from Dataverse on your pages. For instance, you can drag a Form component onto a page and point it to the "Contact Us Form" form on the "Inquiry" table; now that page will show that form to users, and when they submit, it creates a record in Dataverse. Similarly, a List component on a page can display a view of records (with certain columns, sorting, filtering). This greatly speeds up development because you don't have to recreate UI for data operations – you reuse model-driven forms and views.
- **Customization and Pro Dev**: While you can do a lot with the out-of-box tools, Power Pages allows advanced customization:
    - You can add code via the **VS Code integration** or the built-in text editor for webpages. This is how you add JavaScript, HTML, or CSS beyond what the design studio offers.
    - **Liquid templates** – a templating language to fetch and display Dataverse data in web pages (Liquid is used in Power Pages for dynamic content).
    - **Custom web APIs** – you can write server-side logic (as an Azure Function or Dataverse plugin exposed as an OData action) and have the portal call it if needed. These are deeper than PL-900 goes; just know that if something cannot be done with config, developers can extend a Power Pages site with code.
- **Lifecycle & ALM**: Power Pages can be packaged in solutions like other Power Platform components, meaning you can move a site from one environment to another. There's also functionality to have multiple languages on a site, enable tracking, etc., but those are beyond fundamentals.

Use Cases for Power Pages:

- **Customer self-service portal:** e.g., a bank provides a portal for customers to check account status, submit support tickets, or fill forms – all integrated to Dynamics 365 in the back-end (which is Dataverse). This reduces calls to customer service.
- **Partner or Supplier portal:** e.g., a company allows its partners to log in and see inventory or submit orders directly via a portal, rather than emailing orders. The data goes straight into Dataverse/CRM.
- **Internal web portal:** e.g., an intranet site for employees to request services (IT, HR). Although internal users could also use canvas apps, sometimes a web portal is more appropriate for broad access or anonymous scenarios (like an anonymous suggestion box).
- **Community or FAQ site with forums:** Power Pages can include forums and knowledge base (if you integrate with Dynamics 365 Customer Service's portal components). Fundamentals won't get into those specifics, but yes, even forums are possible.
- **Registration or application sites:** e.g., a city government uses a Power Pages site for citizens to apply for permits or licenses. The forms capture data to Dataverse, and staff process it internally (maybe via a model-driven app).
- **Education portals:** Students register for courses or check scores on a Power Pages site tied to an education management Dataverse.

Importantly, **Power Pages is ideal for external users**. Microsoft's own description: "Power Pages is ideal for websites focused on external audiences that require secure access to business information.". The exam may give a scenario like *"We need to build a portal where customers can create and view their support tickets. Which Power Platform component should we use?"* – answer: **Power Pages**.

Exam Tips for Power Pages:

- Recognize when a scenario calls for Power Pages. If the users are external (outside your Azure AD) or anonymous, and you need a web interface, Power Pages is the answer. If the need is an internal web app for staff, it could still be Power Pages or a canvas app – but if it's many users and you want a web URL, Pages can fit even internally.
- Know the **basic parts of building a Power Pages site**: choosing a template, using the design studio (pages, styling, etc.), and that data comes from Dataverse. For example, a question might be: *"Where is data for a Power Pages site stored?"* – In Microsoft Dataverse (and you'd secure it via roles/permissions).
- **Security concepts:** As mentioned, understand web roles and table permissions conceptually. E.g., *"How do you ensure that portal users only see their own records?"* – By configuring table permissions with appropriate filters (e.g., contacts only see records related to their contact record). Or *"What needs to be set up to allow users to sign in with a Google account?"* – configure an OAuth/OIDC identity provider (Google) in the authentication settings.

- **Power Pages vs SharePoint:** Some may wonder, why not just use a SharePoint site for external users? SharePoint external sharing is limited and not a full application platform. Power Pages provides a more robust application portal with Dataverse driving it. The exam might not compare them, but might expect you to choose Power Pages for custom web app needs.
- **Design and Templates:** Possibly a question on how to start a site quickly – using an appropriate template or the default design. Or what the Styling workspace is for.
- **Integration:** Recognize that Power Pages can integrate with other components: embed Power BI (you can show Power BI reports with appropriate permissions on a page), include a Power Virtual Agent/Copilot chatbot on a page (the snippet says that: add chatbots to your site), use Power Automate flows triggered by portal actions (like when a form is submitted, it can call a flow to perform additional tasks).
- **Licensing for Pages:** (We'll cover in the table, but in short: Power Pages is a paid add-on capacity, not included in standard Office licenses. Trial allows development, but for production, one must pay per user or per page view capacity. Internally, if employees use a Power Pages site, those employees need to be licensed either via capacity or certain Dynamics 365 licenses. However, fundamentals likely won't ask specific pricing, just maybe "do you need additional license for external users?" – Yes, you buy Power Pages user passes or capacity.)
- **Methods to create sites:** The outline said *"Describe methods for creating sites"*. This implies knowing you can start from template, or from blank, or possibly via the new Copilot (by description). So if asked, ways to create a Power Pages site: use templates (industry templates available), or start from scratch and design with the Studio, or use Copilot to generate pages from natural language.
- **Differences from Power Apps:** If they throw a curveball scenario where either a canvas app or a Power Pages could be used, focus on the user audience. Internal & small scale → Power Apps; External or broad access via web → Power Pages. Sometimes both could do it, but one fits better if external.
- If the exam asks about **Power Apps Portals**, recall that's the old name for Power Pages. They should use "Power Pages" in current wording, but just in case, know they're the same general tech.

Power Pages is a newer addition to PL-900 content, so ensure you grasp the core idea: building websites with Dataverse backend in a secure, low-code way.

Copilot and AI in Power Platform – Conversational Agents (Copilot Studio)

Microsoft has infused **Copilot (generative AI)** capabilities across the Power Platform. This includes helping makers build solutions and enabling end-users to interact with your solutions in natural language. A big part of this is the evolution of **Power Virtual Agents (chatbots)** into what is now called **Microsoft Copilot Studio** and **custom "Copilot" agents**.

From Power Virtual Agents to Copilot Studio: Power Virtual Agents (PVA) was the tool to create chatbots without code. In late 2023, Microsoft merged PVA into Copilot Studio. **Copilot Studio** is a unified portal for building conversational AI experiences (bots or "agents") that take advantage of generative AI. Essentially, everything you could do in PVA (design conversation topics, call flows, publish to channels) you can do in Copilot Studio – plus new AI-focused capabilities. The exam outline references *"Describe the value of Copilots and generative AI features"* and *"Describe the value of Copilot Studio"*.

So, what does Copilot Studio offer?

- It lets you **create custom AI assistants (chatbots)** for your organization, that can engage in natural language conversation with users (on a website, in Teams, etc.), just like PVA bots did, but now enhanced with AI-generated responses and the ability to use **plugins and connectors** to perform actions or retrieve information.
- It's also the interface to **customize Microsoft 365 Copilot** in an organization. For example, if a company has the M365 Copilot (the AI assistant in Office apps/Teams), Copilot Studio is where they can customize its behavior, define organizational knowledge it can access, or create "plugins" (like connections to internal data).
- Essentially, **Copilot Studio is a comprehensive conversational AI solution that builds on PVA's foundations**. PVA topics and bots are now "Copilot agents" in that studio.

What are Copilot (PVA) agents capable of? Many capabilities carry over from PVA:

- **No-Code Bot Building:** You define conversation **topics** with trigger phrases (what a user might ask to invoke that topic) and design the conversation flow, which can include asking questions, branching based on answers, showing messages, etc., using a visual editor. The difference now is you have assistive AI: the bot can be set to use generative answers for things you didn't script explicitly, using the language model (and data you connect).
- **Generative Answers:** Previously, PVA bots would only say what you authored. Now, with Copilot, the bot can generate answers from sources like a knowledge base or the internet (if

allowed) or just a large language model's training. For example, you might enable a "web search" fallback or connect an internal SharePoint with FAQs for the bot to draw from. The exam might not drill into how, but they expect you to know that modern Power Platform bots can use generative AI to give more fluid, expansive answers, not just pre-authored chat bubbles.

- **Actions via Power Automate:** Just like PVA, Copilot agents can call **Power Automate flows** as skills to do things. For example, if a user asks "I need to reset my password," the bot could trigger a flow that creates a ticket or even resets the password if allowed. This is how bots do enterprise tasks (each flow acts like an API the bot can invoke). On the exam, if a question says "a chatbot needs to retrieve or update info from a system," answer that the bot should call a **Power Automate action (flow)** to integrate with that system.
- **Data connections (Plugins/GPTs):** Microsoft has talked about "plugins and GPTs" in Copilot Studio. A plugin is like a connector for AI that lets the copilot access external data (similar to how ChatGPT plugins work). In Copilot Studio, you can connect your copilot to enterprise data sources via existing connectors or Graph APIs. You might not need to know the term plugin for the exam, but be aware that Copilot agents can be linked to your company's data to answer questions with that data.
- **Orchestration of Workflows:** The Copilot Studio emphasizes orchestrating workflows – meaning an AI agent can call multiple actions or handle multi-turn dialogues that involve doing something (like filling a form through conversation, then submitting). This is essentially still the PVA capability to create more complex conversations.
- **Publishing to Channels:** Copilot/PVA agents can be deployed on various channels: on a **website** (via an embed code, which Power Pages can use), in **Microsoft Teams**, via **custom chat app or mobile app embed,** etc. A very relevant one is Teams – many use PVA/Copilot to make bots for internal Teams channels (e.g., an HR Q\&A bot).
- **Analytics:** There are dashboards for bot performance (session counts, resolution rate, what questions weren't answered, etc.), which help improve the bot over time by identifying gaps.

Use Cases for Copilot Agents (formerly PVA bots):

- *Internal helpdesk bot:* Answers employees' IT or HR questions. For instance, "How do I connect to the VPN?" The answer can be in the bot's knowledge or it can use generative AI to draft an answer based on internal documentation. If the question is complex, the bot might create a ticket (via flow) or escalate to a human.
- *Customer service bot:* On a public website, a chatbot answers common customer queries (order status, product info). With Copilot capabilities, it could even accept a customer's question in natural language and formulate a detailed answer from a product manual (if connected to that data).

- *Transaction bot:* e.g., a bot that helps an employee complete a task like filing an expense. Through conversation, it asks for amount, date, etc., and then calls a flow to save it. This makes forms more user-friendly – the user just chats instead of filling a form.
- *Custom M365 Copilot extension:* Perhaps beyond PL-900, but organizations can use Copilot Studio to tweak what the global Microsoft 365 Copilot responds with. For example, ensure when someone asks Copilot "Tell me the escalation process," it knows the company's internal process (because you taught it in Copilot Studio).
- *FAQ bot with generative fallback:* You might load a list of Q\&A pairs into the bot, but if someone asks something not in the list, the generative AI can still attempt to answer (maybe by combining info or saying it doesn't know, depending on settings).

Exam Tips for Copilot/Agents:

- They might still use the term **Power Virtual Agents** in some contexts, but be prepared that **Copilot Studio = the new PVA experience**. So, an exam question might say "What is Copilot Studio used for?" → To build and manage intelligent chatbots (custom copilots) that use generative AI for conversation.
- If they ask about the **value of Copilots in Power Platform**, you should mention that Copilot features help makers build solutions faster (e.g., chatbot suggested topics, app building by description) and help end-users by making interactions more natural (e.g., asking a chatbot instead of navigating menus, or using a Copilot control in an app to query data in plain language).
- **Channels**: Recognize that a bot (Copilot agent) can be deployed in multiple places, not just one. A likely question: *"Where can you publish a chatbot created in Copilot Studio?"* – Possible answers: in Teams, in a Power Pages site, as a standalone web chat iframe, etc. (All of those are correct).
- **Integration with Power Automate:** Reiterate: to perform an action from the bot, you use flows. If a question describes a bot doing something with backend data, answer includes using Power Automate as a skill.
- **Generative AI vs. Traditional Q\&A:** Understand that older bots were all manually authored. Now, with generative AI, bots can handle a wider range of phrasing and even generate answers from content. The exam might ask something conceptual like: *"How does the integration of generative AI improve chatbot experiences in Power Platform?"* – You'd say it allows bots to give more natural, extensive answers and require less manual authoring of every response.
- There's also mention of **Copilot in other areas**: e.g., Copilot in Power Automate (describe flows) and Copilot in Power Apps (describe apps). Ensure you don't confuse those with

Copilot Studio. Copilot Studio is specifically about conversational agents. The others are Copilot features in design experiences.

- **Value of Copilot Studio:** Summarizing from an exam perspective: It enables creation of **custom AI assistants** that combine your organization's data and large language models to provide helpful responses and automate tasks via conversation. It basically democratizes creating your own ChatGPT-like bot specific to your business.
- **PVA vs Custom M365 Copilot:** Possibly too advanced, but Copilot Studio covers both scenarios. They likely won't ask that specifically, but just note that Copilot Studio is central for both standalone bots and extending Copilot.

In short, **Copilot and AI features** in Power Platform = helping makers (like natural language to create apps/flows) and enabling more powerful bots (Copilot agents that use GPT). Since this is new, if you see any question about AI or GPT in context of Power Platform, tie it to Copilot features. For example, *"Which Power Platform component would you use to build a question-answer chatbot that can use your company's internal documents to generate answers?"* – That's **Copilot Studio (formerly PVA)**, with possibly an intelligent bot that uses your data via plugins.

Licensing and Pricing Overview (Updated for new scope)

Understanding licensing is important to avoid implementing something that users can't actually use. Here's an **updated summary of licensing** for the Power Platform components in scope. The exam may present scenarios indirectly requiring you to know if a feature is free/included or needs extra licensing.

Product / Feature	Included (Standard Licensing)	Requires Premium Licensing
Power Apps	- **Included with M365:** Many Office 365 plans include the ability to create and use **canvas apps with standard connectors** (SharePoint, Excel, Outlook, Teams, etc.) without a separate Power Apps license. Also includes use of **Dataverse for Teams** (build apps in Teams with a limited Dataverse).	- **Using Dataverse (Full) or Premium connectors:** If an app connects to **Dataverse (outside of Teams)** or uses a **premium connector** (e.g., SQL, Oracle, Salesforce), it requires a Power Apps premium license. There are two main plans: **Power Apps per App plan** –

	- This means an Office user can build simple apps storing data in SharePoint or using Excel, and share with colleagues who also have Office licenses, with no extra cost. - **Dynamics 365 license holders** can use Power Apps within the scope of their D365 environment (because D365 uses Dataverse). E.g., a D365 Sales user can access model-driven apps in that environment.	allows a user to run 1 (or a few) specific premium apps. Good if you have a small number of apps to share widely. **Power Apps per User plan** – allows a user to run unlimited apps with premium features. - **Power Pages for internal use:** If internal users (employees) access a Power Pages site, they either need a Power Apps license that covers that or the Pages must be licensed for them (Power Pages has separate model for external vs internal). (Usually, internal use of a portal might require each user to have a Power Apps Per User or a special "authenticated internal user" license add-on for Power Pages. This detail might be beyond PL-900.) - **Advanced capacity:** Very large Dataverse storage or high throughput may require add-ons, but not in fundamentals scope.
Power Automate	- **Included with M365:** Office 365 licenses include use of Power Automate for **standard connectors** in cloud flows. For example, an Office user can create flows that move data between SharePoint and Outlook or Teams. These are subject to some limits (number of runs, etc.) but generally available.	- **Premium connectors or actions:** Using any **premium connector** (e.g., SQL, DocuSign, Azure DevOps, HTTP custom requests) needs a premium license (even if you have O365). Plans: **Per User plan** – unlimited flows for that user (cloud and attended RPA).

	- **Attended RPA Free:** Windows 10/11 users can use Power Automate Desktop free for personal tasks. You can manually run desktop flows on your machine without additional license (it's part of Windows). - **Dataverse for Teams flows:** If the flow is built within a Dataverse for Teams environment, using that data, it's included (similar to Power Apps in Teams).	**Per Flow plan** – allows a specific flow to run for the organization (any user triggers it) with up to n runs/month, useful for organizational workflows. - **Unattended RPA:** Requires an add-on or a special plan. Unattended automation (running RPA bots without a human) is **premium**. Microsoft offers an **Unattended RPA add-on** per bot, or a more expensive **Power Automate RPA plan** that includes unattended capacity. If a question scenario is "automate a legacy app overnight with no user," answer implies needing an unattended RPA license. - **AI Builder:** While PL-900 doesn't cover AI Builder now, using AI Builder models in flows (e.g., form processing, prediction) requires AI Builder credits which usually you purchase (some free quota might come with a Per User plan). Premium context anyway. - **Power Automate Process Mining:** Currently available with certain licenses; likely considered premium or trial-only at this point. (Not deeply tested yet, just know it exists).
Power Pages	- **Trial / Developer:** Anyone can sign up for a free trial environment to build Power Pages. But for	- **Authenticated External Users:** Power Pages is licensed mainly by **user logins** or **page views** for

production, Pages is not included in standard M365 plans by default.
- **For Internal users with certain licenses:** If you have Dynamics 365 licenses, they often include limited use of portals (Power Pages) for the same environment (e.g., D365 Customer Service includes one self-service portal access for those customers). But as a standalone, no free internal use except trial.

external audience. You purchase capacity like “100 authenticated users per month” packs or “500k page views” etc. If a question asks how to license a customer portal, the answer: purchase a suitable Power Pages subscription for the expected usage (previously called Power Apps Portals licensing).
- **Unauthenticated (Anonymous) Users:** Also paid by page views in tiers. If your site allows browsing without login, you pay per 100k anonymous page views, for example.
- **Internal user access:** If internal employees (who already have say Power Apps licenses) need to use the Power Pages site, Microsoft expects those internal users to be covered by a Power Apps Per User license or a capacity equivalent. Essentially, internal use of a Power Pages portal can use either each user’s existing Power Apps license (if it’s simple usage) or also count against capacity. This detail is tricky – but fundamentals likely only asks about external user licensing.
- In summary, **any production use of Power Pages beyond trial = additional licensing** (pay per user or view).

Copilot / Power Virtual Agents	- **Included for Teams (limited):** When PVA was separate, there was a version called “PVA for Teams” that allowed creating bots within Teams without extra license. With Copilot Studio, if building a bot in a Dataverse for Teams environment, it’s likely still included for that context (similar to how other Teams integration is included). Internal-facing low-volume bots can often be made available without extra cost. - If you have Office 365, you can add the Power Virtual Agents app in Teams and create a bot that colleagues can use in Teams – this is included (with some limits on number of sessions per month, etc., default ~2000 sessions/tenant on the “Teams” plan).	- **Stand-alone bots (external or high-volume):** For customer-facing or extensive internal bots, you need a **PVA/Copilot license**, which is billed per bot session. Microsoft sells add-ons like “Power Virtual Agents 2,000 sessions per month” etc. If a question says “the organization wants to deploy a chatbot to their public website,” answer should include that they need to purchase Power Virtual Agents/Copilot capacity for external usage (since that’s not free). - **Custom M365 Copilot**: To use M365 Copilot (the AI in Word/Excel/Outlook), an org needs Microsoft 365 Copilot licenses (very premium). And customizing it via Copilot Studio presupposes you have that. However, PL-900 likely won’t ask about licensing M365 Copilot specifically – just know it’s separate from Power Platform licensing. - **If using premium connectors in flows called by bot:** Indirectly, if your bot calls a Power Automate flow that uses premium connectors, the person triggering that flow (the bot service account) needs a Power Automate license. Microsoft actually included some

		limited Power Automate usage with PVA sessions, but that detail is probably beyond fundamentals. - **Full Dataverse:** To use Dataverse in a production environment, you need a **Power Apps or Power Automate premium license**. For example, if you build a canvas app on Dataverse, all users need Power Apps Per App or higher. If you trigger flows on Dataverse triggers (outside Teams env), you need premium. Essentially, Dataverse is a premium feature (it even has a "premium connector" label) in non-D365 contexts. - **Dataverse capacity add-ons:** Each Power Apps/Power Automate premium license gives you some capacity (storage) pooled at tenant level (like 250 MB database per license, etc.). If your org goes beyond that (more data, files, log storage), you need to buy extra Dataverse capacity. Fundamentals might not test amounts, but be aware this exists for large deployments. - **API usage add-ons:** There are limits on API calls per user per day (to protect service). If big scale automation needed, can get add-on – but unlikely in PL-900 questions.
Dataverse	- **Dataverse for Teams:** included with any Office/Teams license, but limited to use inside the context of Teams apps/bots. Max 2GB storage, max ~10k rows per table, and some features (like complex auditing, offline) not available. Great for small-scale apps. - **Dynamics 365 apps:** come with full Dataverse use for that app's environment. D365 licenses also typically include a certain number of Power Platform usage rights (like running canvas apps within that environment).	

Quick Licensing Scenarios:

- If Bob has an Office 365 E3 license, can he create a flow that saves Outlook attachments to OneDrive? **Yes.** (All standard connectors, O365 provided).
- Can Bob create a canvas app that connects to Salesforce without extra licensing? **No – Salesforce is premium;** he'd need a Power Apps premium license.
- Can your company's 50 customers access a Power Pages portal to check status of orders for free? **No,** you'd need to pay for 50 authenticated external users (via Power Pages add-on) because external usage isn't covered by default.
- You have a Team in Microsoft Teams that wants a simple FAQ bot. Can you do it without extra cost? **Yes,** use Copilot/PVA in Teams – included (with tenant session limits, but likely fine).
- You want an unattended automation that runs on a server every night. What license? **Power Automate with Unattended RPA add-on** (or a special license covering it).
- Always recall: **Standard connectors = included with O365; Premium connectors/Dataverse = premium license needed.** And **external facing stuff (Power Pages, external bots) = separate licensing.**

Choosing the Right Tool: Common Scenarios

One of the classic challenges (and exam tasks) is picking which Power Platform component(s) to use for a given scenario. Here is an updated **scenario mapping** to help:

Scenario	Best Power Platform tool(s)	Explanation
Employees need to **collect data in a custom app** (for example, a factory inspection form app) on their phones. The app requires a custom UI and will use data from SharePoint and write to a new list.	**Power Apps (Canvas app)**	A canvas app is ideal for building a tailored form with offline capabilities if needed. It can connect to SharePoint (standard connector) for data. The app can be shared with employees via the Power Apps mobile app.

The company wants to **display data to external customers** and let them submit requests through a web portal. Authentication should be with personal emails or Microsoft accounts. Data should go into Dataverse for internal team to process.	**Power Pages site** (with Dataverse)	Power Pages is designed for external-facing websites with secure login. You'd build pages for displaying customer-specific data (with table permissions) and forms for new requests. External users can sign up with, say, Microsoft or Google accounts by configuring those identity providers.
A **workflow** is needed: when an employee submits a request (in an app or SharePoint list), approvals from manager and finance should happen, then an email confirmation sent.	**Power Automate (cloud flow)**	This is a classic approval workflow scenario. A Power Automate automated cloud flow can trigger when a new item or record is created, use the Approval action for manager/finance sign-off, and then send emails. This flow can be triggered from the app or list automatically.
We have data in an on-premises SQL Server. We want to build an internal app for employees to view and update that data in a user-friendly interface.	**Power Apps (Canvas app) + On-premises Data Gateway**	A canvas app can provide the front-end. SQL Server is a premium connector, so this requires Power Apps premium licensing. The on-premises gateway would be used so that the cloud app can securely access the on-prem SQL database. The result: employees use the Power App instead of directly editing SQL or using Access forms.
People often ask a set of common questions (either employees about HR policies, or customers about products). We want a **chatbot** to handle these Q\&A, and perform actions like creating a	**Copilot/Power Virtual Agents bot**	A conversational agent built in Copilot Studio (formerly PVA) is the solution. It handles natural language questions 24/7. You would author topics for known FAQs and enable generative answers for variations. For creating a support ticket, the bot can call a Power Automate flow to log the issue. You can deploy this bot on

support ticket if the question can't be answered.		your website (for customers) or in Teams (for employees).
A manager wants a daily summary of sales orders from yesterday, emailed every morning.	**Power Automate (scheduled flow)**	A scheduled cloud flow can run each morning, query the orders (via a connector to the data source, say Dataverse or SQL), compile a summary (maybe using an HTML table or Excel template), and send it via Outlook connector. This automated report saves the manager time.
The HR department needs an internal system to track employee training and certifications. They want to use standard data model and have forms for data entry, with maybe an approval for course enrollment. They also might expand it later or integrate with existing Office data.	**Power Apps (Model-driven app) + Dataverse**	This scenario sounds like it involves relational data (Employees, Courses, Registrations) and would benefit from Dataverse's structure and security. A model-driven app can provide a quick interface (forms for training records, views of who's enrolled). It requires Power Apps licenses, but HR staff likely get full functionality. Power Automate can handle any approval or notification flow (e.g., when someone enrolls, request manager approval).
A small team wants to build a solution **within Teams** for tracking team tasks and issues, ideally without extra licensing. They'd like it to be accessible in a Teams channel tab.	**Power Apps in Teams (Canvas app with Dataverse for Teams)**	This leverages existing Office licenses. In Teams they can create a Power Apps personal app or tab that uses Dataverse for Teams to store tasks/issues. It's all within Teams UI, and they won't need premium licenses as long as they stay in that boundary. If they later need more, they could upgrade to full Dataverse.
The finance team has an old accounting system that has no API. They export data from it daily and then manually copy	**Power Automate Desktop (RPA) flow**	This is a use case for RPA. A desktop flow can be programmed to open the old system, navigate the UI, copy the needed data, and perhaps directly enter it into the web system or at least output to a file. If

some figures into Excel and another web system. This takes an hour every day.		the goal is to schedule it nightly without human, you'd combine with an unattended flow trigger from cloud (requires premium RPA). If attended is okay, they could just run it on demand.
A department needs a quick way to tally survey responses and display results on SharePoint. They already use Microsoft Forms for the survey.	**Power Automate (for automation) + maybe Power BI or Excel for analysis**	If purely Power Platform: A Power Automate flow can trigger on each Form submission, add the data to a SharePoint list or Dataverse. To display results, one might use Power BI (though that's outside PL-900 now) or simply use the list with calculated columns. However, since Power BI isn't in exam scope, maybe the expected answer: use Power Automate to compile results as needed or update an Office document. (This scenario might not be asked since it edges into Power BI, but demonstrates thinking of flows for integration.)

The key to these scenarios is identifying **who the users are (internal vs external)**, **what form of interaction is needed (app UI, web UI, chat, automated process, analysis)**, and **what systems are involved**. Then match to the component:

- **Internal app/UI -> Power Apps**,
- **External web -> Power Pages**,
- **Automation or integration -> Power Automate**,
- **Chat -> Copilot/PVA**,
- **Data storage for app -> Dataverse** (if relational or needing security; otherwise maybe SharePoint or SQL which involves connectors).

Also remember, solutions often use multiple components. An exam question might have multiple parts, e.g., *"Design a solution with a portal and an automation"* – answer could involve Power Pages + Power Automate together, etc.

Tips for Success on the PL-900 Exam

Finally, here are some tips and important points to keep in mind as you prepare to take (or review for) the PL-900 exam:

- **Use the Microsoft Learn Resources:** Microsoft provides free learning paths and modules for PL-900 (updated for the latest exam). These are gold – they often cover exactly what's needed and sometimes the language in the questions matches how things are described in Learn. Be sure to go through the "Skills Measured" and check you've touched each point in this cheat sheet or via Learn.
- **Hands-on Experience:** Even though PL-900 doesn't have a hands-on lab, having tried the tools helps you remember details. If possible, sign up for a free Power Apps community plan or use the free developer environment in Teams, and experiment:
    - Build a simple canvas app and a model-driven app.
    - Create a basic cloud flow (e.g., use a template and modify it).
    - Spin up a Power Pages trial and create a page with a form (feel the design studio).
    - Create a simple bot in Teams or the PVA trial, add a topic, and call a flow from it. This practical exposure cements concepts like where to click to share an app, how to add a connector, etc., which helps on questions that test conceptual understanding.
- **Remember the New Focus:** As highlighted, governance and environment questions will be there. Don't neglect studying up on what environments are, how security roles work in Dataverse, and what DLP policies do. A few questions will likely target those areas. For instance, *"What's the best way to prevent citizen developers from using certain connectors?"* – create a DLP policy.
- **Standard vs Premium Connectors:** Drill this into memory. If you see "SQL", "Oracle", "Salesforce", "ServiceNow" – think *premium*. If you see "SharePoint", "Excel Online", "MS Teams", "Outlook" – think *standard*. The exam might not directly ask "is this premium?" but as part of scenario where license is implied.
- **Terminus Technicus:** Pay attention to specific terms:
    - **Canvas App**, **Model-driven App** – difference.
    - **Automated/Instant/Scheduled flow** – know these subtypes for triggers.
    - **Environment**, **Solution**, **DLP policy** – governance terms.
    - **Copilot** – means AI stuff, could refer to Copilot bot or Copilot maker features, use context.
    - **AI Builder** – (likely not in exam now, but if mentioned as a distractor, remember it's deemphasized).

    - **Power Fx** – that's the formula language for canvas (maybe a question like "What language is used to write expressions in Power Apps?").
    - **Business Process Flow vs Automated flow** – one is a guided UI in model apps, one is a background automation. Could be a trick.
    - **Portal** – old name for Power Pages. If exam uses "portal", they mean Power Pages site.
- **Eliminate Irrelevant Options:** Some questions are multiple choice with more options than correct answers. You can often eliminate:
    - Anything with Power BI (since it's not in scope – unless the question is about integration, but they probably won't).
    - Anything that suggests writing custom code (the exam sticks to Power Platform capabilities, not custom dev, except maybe mention custom connector or PCF – but unlikely).
    - If a question is in the Power Automate section, options might include unrelated things (like "create a model-driven app" – irrelevant to an automate question).
    - Use logic: e.g., they might give both "create a canvas app" and "create a model-driven app" as options – use scenario cues to pick one.
- **Time management:** There will be ~40-60 questions. You'll have around 60 minutes (though exact time can vary). That's about 1-1.5 minutes per question. Most questions are not super long. Some are case studies or have exhibits – read the question first, then the content if needed. The cheat sheet knowledge will help you quickly identify what the question is getting at so you don't overthink.
- **Practice Questions:** If you have access to the official practice assessment or other reputable practice tests, use them. They reveal how questions may be phrased. Just be cautious: any practice material dated before mid-2025 will include Power BI and exclude Power Pages, which is outdated. Stick to updated sources.
- **Stay Updated:** The exam can evolve. We updated as of 2025. If you happen to take this later, double-check if the skills outline changed again (Microsoft Learn page will list any changes).
- **Answer from Microsoft's Perspective:** Sometimes you might know multiple ways to do something, but consider what the exam expected answer is. For example, to build a basic internal FAQ, one could use a SharePoint page, but since this is Power Platform exam, they likely expect "Power Virtual Agents bot" as the solution for a Q\&A scenario, because that showcases the technology. Think of the **Power Platform-first solution**.
- **Licensing Questions Approach:** They generally won't ask you to compute cost, but rather "Can person X do Y with license Z?" or "What license is needed in this scenario?". Use the rule: need premium license if any premium tech is used (Dataverse, premium connectors, external sharing). If only O365 services are involved, then likely covered by existing license.

- **Use case alignment:** If stuck, match keywords:
    - "Automate", "workflow", "notify", "schedule" -> Power Automate.
    - "Customize Teams with an app" -> Power Apps in Teams.
    - "External users" -> Power Pages (or possibly PVA bot on a website).
    - "Chatbot" or "questions" -> PVA/Copilot.
    - "Analyze data" -> (Though Power BI is out, if it's light analysis within an app, maybe refer to a chart control or Excel export – but they likely avoid heavy analysis questions).
    - "AI" or "GPT" -> Copilot either in building or a bot.
- **Review the question end-to-end:** Some are tricky. E.g., "Which two of the following are benefits of Dataverse?" The answers might have three seemingly correct statements and one that's incorrect – pick the two that align with official messaging (like data encryption, role security are correct; something like "unlimited storage free of charge" would be incorrect since capacity is limited).

By mastering the above topics and tips, and using this cheat sheet to fill gaps, you'll be well-prepared for the PL-900 exam. Remember to answer as if you are recommending Microsoft's solutions to a scenario. Good luck – you've got this fundamental knowledge down, now it's about applying it confidently on exam day!

References

[1] Navigating the New PL-900 Exam Landscape (Post-June 2025)
[2] Power Platform Fundamentals PL-900 Training - Dynamics Edge
[3] Power Pages capabilities | Microsoft Learn
[4] Study guide for Exam PL-900: Microsoft Power Platform Fundamentals
[5] cmty_blog_detail
[6] Microsoft Power Virtual Agents, now part of Microsoft Copilot Studio