Dataverse

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

DESIGN DATA AND ACCESS

- **Tables** Structure
- **Relationships** Connect
- **Security** Authorize
- **Audit** Trace

01 Configure Microsoft Dataverse

Tables, ownership and external data

- **Standard tables** store business records in Dataverse. Reuse suitable tables before adding custom ones. **Activity tables** represent work such as tasks and appointments and can appear on a timeline.
- **User or team ownership** supports owner-based and business-unit access. **Organization ownership** suits shared reference data with organization-wide privilege scope. Choose ownership when designing the table.
- **Virtual tables** retrieve external data through a provider without copying business rows into Dataverse. Source availability, provider capabilities and external security remain part of the design.
- Virtual tables do not provide the full security and history features of stored tables: do not assume **column security**, **auditing**, **change tracking** or mobile offline support. **Data Source Secret** protects connection settings, not business columns.

Column types and derived values

- Use **Whole Number** for integers, **Decimal** for precise fractional values, **Currency** for monetary amounts, **Yes/No** for binary values and **Choice** for a controlled list. A **Lookup** relates a row to another table.
- **Formula columns** use Power Fx; **calculated columns** use configured calculation logic; **rollup columns** aggregate related rows. Derived values are read-only to users. Rollup recalculation can lag behind row changes.
- Convert a currency input with **Decimal** when using it in a Dataverse formula column, for example **Decimal(Fee) * 0.20**. The formula result is numeric; a currency output type is not supported.
- Choose **Date Only**, **User Local** or **Time Zone Independent** behavior deliberately. A stored age at registration is a historical value; current age changes with time. Use date-aware comparisons rather than dividing elapsed days by 365.
- **Autonumber** creates formatted identifiers; gaps are possible. An **alternate key** identifies rows by business columns and can enforce uniqueness for populated key values. Ensure key columns are populated when duplicates must be prevented.

Dataverse

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

DESIGN DATA AND ACCESS

- **Tables** — Structure
- **Relationships** — Connect
- **Security** — Authorize
- **Audit** — Trace

Relationships, mapping and cascading

- **1:N** and **N:1** describe opposite sides of the same lookup relationship. **N:N** links many rows on both sides; use an explicit junction table when the association needs its own business attributes.
- **Relationship column mappings** prefill a new related row when it is created from its parent context. They do not continuously synchronize later edits between parent and child.
- **Parental** behavior cascades supported actions to related rows. **Referential** behavior keeps rows more independent. Configure assignment, sharing, reparenting and deletion separately when requirements differ.
- **Cascade All** can reassign all children with a parent; narrower options affect only selected child rows. **Restrict Delete** blocks parent deletion while dependent rows exist. Test both data integrity and access after reassignment.

Privileges and business units

- **Security roles** combine privileges such as Create, Read, Write, Delete, Append, Append To, Assign and Share with access depth. Permissions from multiple roles and teams accumulate.
- **User**, **Business Unit**, **Parent: Child Business Units** and **Organization** define increasing row-access scopes for owner-based tables. Organization-owned tables use organization-level access rather than per-owner scoping.
- **Append** allows a row to be attached to another row; **Append To** allows the target row to accept that association. A lookup operation can require privileges on both tables.
- **Sharing** grants access to specific rows; it does not replace the table privileges required to use them. Hiding a field, view or command does not secure the underlying data.
- **Manager hierarchy** and **Position hierarchy** add access through reporting or position structures. Configure depth and business-unit relationships deliberately; hierarchy access supplements the role model.

Teams and sensitive columns

- **Owner teams** can own rows and receive security roles. **Access teams** provide shared access to particular rows, do not own them and do not receive security roles. Members still need applicable table privileges.
- **Microsoft Entra group teams** use directory group membership to simplify access administration. A team in Microsoft Teams and a Dataverse owner or access team are different concepts.
- Enable **column security** on eligible columns, then assign a **column security profile** to users or teams. Configure Read, Create and Update separately; read-only sensitive data needs Read without Update.
- Test permissions with a representative least-privilege account. **System Administrator** access can conceal security defects. Recheck inherited roles, sharing and team membership when access exceeds requirements.

Dataverse

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

DESIGN DATA AND ACCESS

- **Tables** — Structure
- **Relationships** — Connect
- **Security** — Authorize
- **Audit** — Trace

Auditing, uniqueness and retention

- **Dataverse auditing** requires auditing at environment, table and selected column levels to record column changes. Enable only the required data and give reviewers permission to read the history.
- **Audit history** answers who changed a value and when. **Change tracking** identifies changed rows for synchronization. Sign-in and read-access investigations may require separate identity or activity logs.
- Set **audit retention** for the required history and monitor log capacity. Removing audit history is separate from deleting business rows; use **bulk deletion** jobs for scheduled business-data cleanup.
- **Duplicate detection rules** define matching criteria and must be published. Configure save-time and import-time detection separately; scheduled duplicate detection jobs find existing duplicates for review.
- Duplicate detection is not a guaranteed unique constraint or automatic merge. Use an **alternate key** where strict uniqueness is required; resolve existing conflicts before activating the key.

Search, filters and currencies

- **Dataverse search** returns relevance-ranked results across configured tables and can search supported notes and attachments. Search configuration and security determine which data can appear.
- **Quick Find** uses a table-specific search configuration. **Advanced Find** and view filters express structured conditions, including related-table criteria, null checks and relative dates.
- A column's **Searchable** setting controls its availability in query/filter designers. **Quick Find find columns** control searched fields. Neither setting is a substitute for data security.
- **Base currency** supports common reporting amounts and cannot be deactivated. Deactivate an unused transaction currency to prevent new selection while retaining historical records; changing a language does not change currency availability.

Power Apps & Pages

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

CHOOSE THE EXPERIENCE

- **Model-driven** — Records
- **Canvas** — Tailored UI
- **Pages** — Websites
- **Permissions** — Protect

02 Build Power Apps and Power Pages

App types and model-driven forms

- **Model-driven apps** build on Dataverse tables, forms, views and security. **Canvas apps** offer explicit layout and connector-based data access. **Power Pages** provides websites for external or internal audiences using Dataverse.
- **Main forms** provide full record work. **Quick create forms** collect a small set of values without navigating away. **Quick view forms** display a related row as read-only information.
- **Subgrids** list related rows; supported editable grids permit inline updates. A **form component** can support editing a related row. Calculated and other read-only values remain noneditable.
- **Custom pages** add focused canvas-style experiences inside model-driven apps. **Embedded canvas apps** provide contextual interfaces on forms; pass record context and handle data refresh deliberately.

Views, charts and navigation

- **Public/system views** are solution components created in the maker experience; publish changes and include the views in the app. **Personal views** are user-owned and can be shared with appropriate users.
- A view defines **columns**, **sort order** and **filter criteria**. Use relative-date filters for lists that must stay current, and null conditions to identify missing data such as an alternate phone number.
- **System charts** are organization components; **personal charts** can be shared with a team. A chart's series defines the measure and aggregation, while categories define grouping.
- Use **dashboards** to combine lists and charts. The **timeline** presents activities, notes and supported posts. Configure **site map/navigation** areas, groups and pages to expose the correct tables and dashboards.
- **Calendar**, **Kanban** and editable-grid controls serve different tasks. Match the control to the table, client and interaction requirements, then verify sorting, grouping, editing and business-rule support.

Business rules and UI behavior

- **Business rules** combine conditions with actions such as setting values, requirement levels, validation messages and form recommendations. Form rules can show, hide, lock or unlock supported controls.
- **Table scope** applies supported server-side logic beyond a single form. **All Forms** and **Specific Form** scope client behavior to model-driven forms. Include columns needed by a form rule on that form.
- A canvas app can benefit from supported **table-scoped validation and data logic**. Configure its visual behavior with Power Fx; model-driven show/hide actions do not control canvas controls.
- Use **Visible**, **DisplayMode** and other canvas properties for conditional UI. A password-style input or masked label changes presentation; enforce sensitive-data permissions at the data source.

Power Apps & Pages

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

CHOOSE THE EXPERIENCE

- **Model-driven** — Records
- **Canvas** — Tailored UI
- **Pages** — Websites
- **Permissions** — Protect

Power Fx, events and state

- **Power Fx** uses Excel-like expressions. Property formulas recalculate from dependencies; behavior formulas perform actions in events such as **OnSelect**, **OnChange**, **OnCheck** and **OnUncheck**.
- **Set** creates or updates an app-wide global variable. **UpdateContext** creates or updates a screen-scoped variable; the third argument of **Navigate** can pass context into the destination screen.
- **Collect** adds records and can create an in-memory collection. **Clear** removes collection rows; **ClearCollect** replaces them. Collections are app-wide tables and do not automatically persist between sessions.
- **Navigate** changes screens; **Back** returns through navigation history. **OnVisible** runs when a screen becomes visible. Initialize shared state deliberately to avoid overwriting it each time users revisit a screen.
- **If** chooses outcomes, **Coalesce** selects the first nonblank value and **Filter** selects rows. Use **Text** for display formatting and **Color** for conditional emphasis without changing the stored business value.

Canvas data, layout and offline work

- **SubmitForm** saves a form; run dependent actions such as notifications from **OnSuccess**. **Patch** creates or updates selected values; handle errors before reporting success.
- **Gallery** controls repeat records. Track selected items in a collection through OnCheck/OnUncheck and bind checkbox defaults to that state. **Reset** restores a control's Default; it does not inherently mean false.
- For **responsive layout**, disable Scale to fit where appropriate and use containers plus formulas based on available size. Check touch targets, labels, keyboard navigation and contrasting text across screen sizes.
- **Delegation** sends supported queries to the data source. A nondelegable query can inspect only a limited local subset, producing incomplete results on large datasets. Address delegation warnings before rollout.
- **Offline** requires a deliberate local-data and synchronization design. SaveData/LoadData can persist collections on supported clients; test connection detection, queued updates and conflicts. An embedded app does not gain offline support automatically.

Power Apps & Pages

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

CHOOSE THE EXPERIENCE

- **Model-driven** — Records
- **Canvas** — Tailored UI
- **Pages** — Websites
- **Permissions** — Protect

Modern commands and contextual actions

- **Power Fx commands** can save changes, navigate to records or URLs and request confirmation. Use the functions supported by the command host, such as **Patch**, **Navigate**, **Launch** and **Confirm**.
- **Self.Selected.Item** represents a selected record; **Self.Selected.AllItems** represents selected grid rows. A bulk command's Visible formula can require a nonempty selection and a qualifying status on every selected row.
- Use **CountRows** and **Filter** to evaluate the selection. Visibility improves usability; the action must still respect permissions and validate records whose status may have changed since selection.

Power Pages identity and registration

- An authenticated **Power Pages user** is associated with a Dataverse **Contact**. Configure the identity provider, registration method and contact linking separately from authorization.
- **Microsoft Entra ID** supports corporate identities. Other configured external providers support their own audiences. **Open registration** allows self-service sign-up; invitation-based registration targets known contacts.
- **Invitations** can include an expiry, invitation code and web-role assignment on redemption. Send a redemption link through the invitation process; email confirmation establishes control of the supplied email address.
- **Web roles** collect site permissions. A role marked **Authenticated Users** applies to signed-in users; **Anonymous Users** applies to unauthenticated visitors. Do not grant anonymous access to sensitive records.

Power Pages permissions and site design

- **Page permissions** control access to website pages and can inherit through a page hierarchy. **Table permissions** control Dataverse row operations. Access to a page does not itself authorize its data.
- Associate table permissions with **web roles** and grant only needed operations. **Contact**, **Account**, **Self** and parent-related scopes restrict rows through relationships; **Global** scope reaches all applicable rows.
- For a user to see only their own submissions or store records, model the contact/account relationship and use the matching **table-permission scope**. Authentication identifies the user; row scope determines their data.

Power Apps & Pages / Process Automation

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

FROM EVENT TO OUTCOME

Trigger Start → **Condition** Decide → **Action** Execute → **History** Verify

- Use suitable **scheduling**, **registration** or **starter layout** templates to reduce custom work. Configure branding, supported languages and translated content; use separate sites when audiences need independent URLs and access design.
- **Search facets** help filter indexed website results. SharePoint integration exposes supported documents; virtual-table providers can expose supported external lists. Confirm each source's supported integration and permissions before combining experiences.

03 Automate processes

Choose the automation mechanism

- **Automated cloud flows** react to events; **instant flows** run on demand or from an app; **scheduled flows** use recurrence. Use connectors to orchestrate services and external notifications.
- **Desktop flows** automate desktop or browser interfaces when a suitable API is unavailable. **Business process flows** guide users through stages. A guided process does not replace the automation that performs its tasks.
- **Real-time classic workflows** run synchronously and can stop a Dataverse operation on validation failure. **Background workflows** run asynchronously. Cloud flows are asynchronous and cannot guarantee blocking the original save.
- Keep independent notifications and bulk processing asynchronous where possible. Use synchronous processing only when the transaction must wait for a result or must be rejected before it commits.

Cloud triggers, loops and approvals

- A flow has a **trigger** followed by actions. A **connector** defines available operations; a **connection** supplies authentication. Match the action to the service: Dataverse uses rows, SharePoint uses list items.
- For a Dataverse change trigger, select the table, change type and scope. Use **filter columns** and **filter rows/trigger conditions** to limit runs and prevent a flow from repeatedly triggering itself.
- **Apply to each** processes an array; **Do until** repeats until a condition is met, subject to limits. **Condition** branches on true/false; **Switch** handles discrete values; parallel branches run independent work.

Process Automation

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

FROM EVENT TO OUTCOME

Trigger Start → **Condition** Decide → **Action** Execute → **History** Verify

- **Approvals** collect decisions from assigned approvers. Test approval, rejection, timeout and reassignment paths. A recurring overdue-record process should query eligible rows rather than rely on a manually refreshed list.
- When a parent may complete only after all children finish, query the related rows and test for remaining incomplete work. Handle concurrent updates and repeated triggers so completion messages are sent only as intended.

Expressions, values and time

- **Dynamic content** inserts trigger or action outputs. Workflow expressions such as **if**, **equals**, **and**, **or** and **concat** combine values. Power Automate expression syntax differs from Power Fx.
- Use **or** when any criterion qualifies and **and** when every criterion must hold. Compare numbers as numbers, booleans as booleans and Choice codes using the actual connector output type.
- **coalesce** supplies a non-null fallback; an empty string may need a separate **empty** check. Null-safe property access helps with absent fields; validate missing required values before downstream actions.
- **utcNow**, **addDays**, **convertTimeZone** and **formatDateTime** support date calculations and display. Set recurrence time zones explicitly and distinguish a formatted date string from a value used for comparison.

Failures, retries and transactions

- Inspect **run history**, trigger inputs, action outputs and connection errors. **Flow checker** identifies design issues. Turn off an overactive flow while correcting its trigger or filters, then retest before enabling it.
- Use **Scope** to group work and **Configure run after** for failure, timeout or skipped-action paths. Retry transient failures; avoid retrying a permanent validation error indefinitely.
- **Resubmit** reprocesses a run, so design for duplicate requests. Use stable identifiers and idempotent updates; check which runs actually failed before using any bulk reprocessing capability.
- A Dataverse **changeset** makes supported Add, Update and Delete row actions atomic. It cannot include arbitrary connector actions, loops or references to outputs from earlier actions inside the changeset.
- SharePoint and other external services need explicit **compensating actions** to undo completed work after failure. A generic scope is not a database transaction. A developer-supplied **custom API** can encapsulate more complex server logic.

Process Automation

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

FROM EVENT TO OUTCOME

Trigger Start → **Condition** Decide → **Action** Execute → **History** Verify

Classic workflows and Dataverse actions

- Configure classic workflow **start conditions** for creation, assignment, status or selected column changes. **Check Condition** branches; **Wait Condition** delays eligible background work; on-demand workflows are started explicitly.
- Use **Update Record** for values and **Change Status** for supported state/status transitions. An update that needs the original value requires processing that still has access to that value; an after-change read returns the new value.
- **Bound actions** operate in a table/row context. **Unbound actions** are not bound to a specific row. Match the Power Automate action to the binding declared by the action or custom API.
- An **action step** in a business process flow needs an activated, compatible action. Check its table binding and supported parameters. Test workflow owner permissions, activation, process sessions and email delivery when automation appears silent.

Business process stages and access

- **Stages** describe major phases; **steps** collect required values within a stage. Add conditions for branching, and keep the intended path understandable to users.
- **Required steps** guide stage progression. Enforce mandatory data integrity with appropriate server-side logic too, because imports or other clients can bypass a form's guided experience.
- Activate the **business process flow**, include it in the model-driven app and enable appropriate **security roles**. Process order influences the default choice; deactivation or removing access prevents use of an unwanted process.
- A flow definition and its **process instances** are distinct. Test changes with both existing and new records. Stage-entry/exit workflows can automate related work; offline availability also depends on the supported app and process configuration.

Desktop flows and browser automation

- **Power Automate for desktop** provides prebuilt actions and recording. Prefer stable **UI elements/selectors** and browser HTML values over screen coordinates when automating changing websites.
- **Attended** automation uses an interactive user session. **Unattended** automation uses a configured machine and credentials without an operator. Verify machine connectivity, session eligibility and runtime permissions for the selected mode.
- Define **input variables** for data supplied by a calling cloud flow and **output variables** for extracted results. Keep integration orchestration and Dataverse updates explicit in the calling flow.
- Test selector changes, waits, missing elements and error paths. Move solution-aware desktop flows with their dependencies and configure machine connections in the destination. Legacy recorder setup instructions are not a reliable deployment checklist.

Chatbots & AI

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

RECOGNIZE AND RESPOND

Topic Intent → **Entity** Recognize → **Variable** Remember → **Action** Respond

04 Configure chatbots and AI capabilities

Topics, entities and conversation state

- **Copilot Studio** and **Power Virtual Agents** terminology appears in chatbot scenarios. A **topic** defines a conversation path; trigger phrases or intent recognition route a user into it.
- **Question nodes** collect responses. **Entities** recognize structured information such as a number, location or product category. **Slot filling** stores the recognized value in a variable.
- **Custom entities** model domain vocabulary. **Synonyms** map known variations to a canonical item; **smart matching** broadens recognition. Multiple-choice questions provide explicit selectable options.
- **Topic variables** stay within their topic; **global/bot variables** retain context across topics during a conversation. Use conditions to branch on numeric ranges and define boundaries without overlaps or gaps.
- Extracted or suggested topics from FAQ content require review, enabling and testing. Reusable product data belongs in a governed data source or reusable definition; a conversation variable is not shared persistent storage across independent bots.

Fallback, actions and live-agent handoff

- **Fallback topics** handle unrecognized input. Offer clarification, capture the unresolved request or escalate rather than repeatedly routing users to unrelated content.
- **Escalation/handoff** transfers a conversation to a configured live-agent service, such as Omnichannel. Pass relevant context and route by information such as location when the integration supports it.
- **Power Automate actions** let a bot query or update backend systems. Define inputs and outputs, check user authentication and return a usable response when a service fails.
- **Resolution**, **escalation** and **abandonment** describe different session outcomes. Use conversation analytics and unsuccessful topics to improve recognition; an escalated session was not fully resolved by the bot.

Publishing, channels and bot access

- **Publish** makes tested changes available to configured channels. A **demo website** supports stakeholder testing; a custom website uses the supported embed/channel configuration; Teams requires its channel and distribution setup.
- Separate **coauthor access** from permission to chat with the published bot. Teams organization-wide discovery can require administrator approval; a team-only listing does not automatically reach the whole organization.

Chatbots & AI / Operations & Integration

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

RECOGNIZE AND RESPOND

Topic Intent → **Entity** Recognize → **Variable** Remember → **Action** Respond

- **Authentication** establishes the caller's identity. Configure manual authentication or supported single sign-on for the intended channel. Secure backend data independently and retrieve profile attributes through authorized services.
- **Web channel security** protects the channel connection; authentication protects user access. Check environment eligibility, Dataverse provisioning, region, language and maker permissions when creating or deploying a bot.

AI Builder model selection and quality

- **Document processing** extracts configured fields from documents. **Text recognition** extracts text. **Business card** and receipt models handle specialized inputs; select a model by the business input and required output.
- **Sentiment analysis** evaluates tone; **key phrase extraction** identifies important phrases; **entity extraction** identifies information types. **Prediction** learns outcomes from historical business data.
- For a custom model, prepare representative data, train, evaluate and **publish** before use in an app or flow. Prebuilt models avoid custom training where their supported output meets the requirement.
- Evaluate on separate, realistic data. Reusing the same complete dataset for training and testing can hide poor generalization. Compare predicted and actual outcomes and monitor drift before relying on business decisions.

05 Manage environments, integration and deployment

Environment boundaries and administrator roles

- **Environments** isolate apps, flows, connections and Dataverse data. Separate development, test/UAT and production. Sandbox environments support controlled testing and administration; protect production from unintended test writes.
- An environment **security group** controls eligible users, while **Dataverse roles** control their data operations. App sharing and licensing are additional access requirements. A maker role alone does not grant business-data access.
- **Environment Maker** creates resources; **System Customizer** supports Dataverse customization; **System Administrator** has broad environment data and configuration access. Tenant/service administrators manage a different administrative scope.
- **Microsoft Entra ID** owns user sign-in identities. Update an existing user's sign-in name through authorized identity administration to preserve identity continuity. Do not recreate the identity merely to rename it.

Operations & Integration

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

PROMOTE A TESTED SOLUTION

Develop Unmanaged → **Package** Dependencies → **Deploy** Configure → **Validate** Access

- Use session/inactivity policies and **Conditional Access** where supported for shared-device and location restrictions. **Data policies** classify connectors and govern permitted combinations; they do not replace row permissions.
- **Dataverse for Teams** provides a team-scoped environment provisioned through supported app/bot creation or installation in Teams. Publish the app and configure team/table access; adding a tab alone does not grant all required permissions.

Solution packaging and dependencies

- Author in an **unmanaged solution** and export a **managed solution** for controlled deployment to downstream environments. A solution packages components; ordinary business rows need a separate data-migration process.
- A **publisher prefix** distinguishes custom schema names and helps avoid collisions. Choose it before creating components. Include the required tables, columns, forms, views, flows, apps and navigation changes.
- **Segmented solutions** include selected table assets. Include table metadata when changing table-level properties, and selected columns/forms for focused changes. Check dependencies instead of adding unrelated components indiscriminately.
- **Connection references** are portable solution components mapped to actual target connections. **Environment variables** hold configurable values such as service URLs or report IDs; provide target-specific values during deployment.
- Avoid carrying a development-specific **current value** into another environment. Publish changes, run **solution checker**, resolve missing dependencies and test import before production. Checker results do not replace functional testing.

Solution layers, updates and upgrades

- **Managed properties** control which supported customizations consumers may change. Configure them in the source before export; deploying as managed does not automatically make every component immutable.
- **Update** adds or changes components while retaining components omitted from the new package. **Upgrade** can remove obsolete components and consolidate earlier patches; assess dependencies and data impact first.
- **Patches** deliver focused changes to an installed parent solution. Roll them into a later full solution when appropriate. Shared components require coordinated ownership and dependency management across solutions.

Operations & Integration

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

PROMOTE A TESTED SOLUTION

Develop Unmanaged → **Package** Dependencies → **Deploy** Configure → **Validate** Access

- Deleting an **unmanaged solution** removes its container while leaving customizations. Uninstalling a **managed solution** removes its managed layers and can remove owned components and associated data, subject to dependencies.
- Automate authorized deployments with a suitable **service identity/application user** and least privilege. Use solution layers and import logs to diagnose effective behavior rather than changing production components ad hoc.

Languages, labels and branding

- Enable required **languages** for the environment, export **translations** from the source solution, translate labels, then import and publish. Users select an enabled UI language through their preferences.
- Translated metadata does not automatically translate business-row text, canvas strings, emails or website content. Plan those resources separately and test locale-specific dates and numbers.
- **Themes** define supported app branding such as colors and logos; **web resources** package reusable files such as icons, scripts and styles. Include required libraries rather than depending on an unrelated installed solution.

Imports, Power Query and refresh

- **Import maps** match source columns to destination columns even when headings differ. Map lookups to an unambiguous related-row identifier; self-referencing parent relationships require correct lookup mapping too.
- **Dataflows** use **Power Query** to extract, clean, transform and load data on a schedule. Canvas apps use **Power Fx** for app-side queries. A dataflow copies data; a virtual table reads it at source.
- Map a stable source identifier to an **alternate key** for repeatable updates/upserts. Use grouping and aggregation to produce summary rows; avoid a refresh design that creates the same records repeatedly.
- For Power Query errors, inspect the error cell's **reason, message and detail** and trace the failing transformation. Check types, locale conversions, missing columns and credentials before replacing error values.
- **Incremental refresh** processes eligible recent/changed data rather than full history. Select appropriate date/time fields and ranges. Refresh schedules, query folding, gateway availability and destination support affect performance.
- Use an **on-premises data gateway** for supported connectors to private/on-premises data. Confirm credentials and network reachability. A gateway provides connectivity; the connector and permissions determine the available operations.

Operations & Integration

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

PROMOTE A TESTED SOLUTION

Develop Unmanaged → **Package** Dependencies → **Deploy** Configure → **Validate** Access

Documents and reusable templates

- A **File column** stores one file value per row/column; **Notes** attach text and files to a record. These consume Dataverse storage. Choose a document-management service for extensive libraries and versioning.
- **Server-based SharePoint integration** exposes documents alongside Dataverse records while storing files in SharePoint. Configure sites, table document management and document locations; Dataverse and SharePoint permissions must both be correct.
- For related tables, plan the **document location hierarchy** and parent lookup so files appear under the intended account or contact context. Confirm online/on-premises integration support for the chosen app or website.
- **Word templates** generate standardized documents from row and supported related data. **Excel templates** support analysis. Organization templates and personal templates have different visibility; uploading a template does not grant access to its source data.

Email tracking and server-side synchronization

- **Server-side synchronization** connects supported email systems with Dataverse. Configure the server profile and mailbox synchronization methods, obtain required approval, then **Test & Enable** and review mailbox alerts.
- Mailbox approval requires the applicable privileged role and approval permission; a general data role alone may be insufficient. Diagnose approval separately from connection, authentication or message-delivery failures.
- **Dynamics 365 App for Outlook** supports manual email tracking and setting a **Regarding** row. **Folder-level tracking** can associate email moved into mapped folders; supported category tracking marks Outlook items for synchronization.
- Use synchronization filters to control eligible **contacts, appointments and tasks**. A connector-based email can be sent without creating a Dataverse email activity; create/track the activity explicitly when a customer history is required.

Operations & Integration

Microsoft Power Platform Functional Consultant

KNOWLEDGE CHEATSHEET

PROMOTE A TESTED SOLUTION

Develop Unmanaged → **Package** Dependencies → **Deploy** Configure → **Validate** Access

Power BI embedding and data access

- **Power Apps visual in Power BI** hosts a canvas app and passes selected report data through **PowerBIIntegration.Data**. **Power BI tiles/reports in Power Apps** display analytical content inside an app; these are different integration directions.
- Publish updated canvas apps so embedded users receive the change. App changes, report data refresh and changes to the fields passed by the visual have separate update paths.
- Grant access to both the **app** and the relevant **Power BI report/dashboard**. Embedding does not bypass authentication, licensing or source security. Use supported guest sharing for external users and restrict resharing when needed.
- **Row-level security** filters Power BI data for viewers through model roles. Dataverse column security profiles do not define Power BI RLS. Test report access as the intended consumer.
- Add reports through supported **dashboards/navigation** in model-driven apps. Use environment-specific workspace/report configuration during deployment. **Power BI data alerts** on supported numeric tiles can trigger a cloud flow for notifications.