First Week Content

Introduction of Microsoft Dynamics 365 Customer Engagement

Link for this doc. Click to see the most up-to-date version online-
https://docs.google.com/document/d/1IFFWVS2rpfyRojM-weMHqtRJhMjhdwHa7G2HKSmj0EY/edit?usp=sharing

*Dynamics 365's customer engagement applications are customer focused business applications that are built on the **Microsoft Power Platform**. They are designed to help organizations more effectively engage their customers. The sales and marketing applications help identify sales prospects and turn them into long lasting relationships. Our service focused applications help your organization service your customers, whether you have a call center, service center, or are providing service in the field. Dynamics 365 customer engagement applications ensure that you have the tools necessary to fully engage your customers across all aspects of your business.*

Why Microsoft Dynamics 365?

Microsoft Dynamics 365 is an innovative cloud-based enterprise software solution that brings ERP, CRM, and supporting business applications together as one streamlined product for a modern, simple, and unified experience. It helps run your entire business and deliver greater results by connecting products/services with people and data.

The real-time, predictive, and AI-driven insights that you derive will help you meet your business goals in sales, marketing, operations, services, and more. Microsoft Dynamics 365 is designed to provide you maximum flexibility and extensibility and can adapt to fit almost any business.

Define customer engagement

A customer is an individual or company that receives, consumes, or buys a product or service. The primary goal of a commercial enterprise is to attract customers and maintain a strong relationship with them over-time, by ensuring they want to continue to do business with your organization.

Organizations accomplish this through customer engagement. Customer engagement represents the emotional connection between a customer and a brand. How an organization engages their customers directly impacts their relationship with them. Highly engaged customers buy more and demonstrate more loyalty.

Many organizations leverage customer engagement software to assist them in engaging with their customers. Customer engagement applications help organizations fundamentally re-imagine how they engage customers. They help organizations create personalized marketing, sales, and service experiences using data and intelligence to improve every interaction.

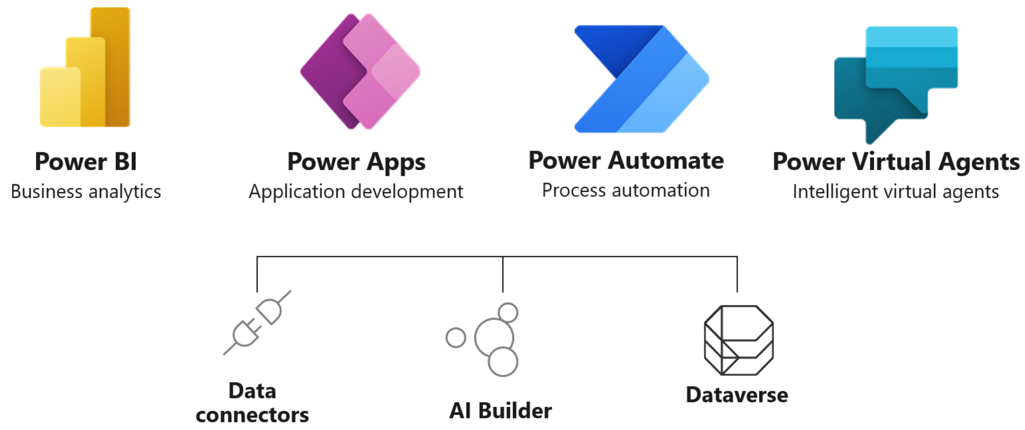
Dynamics 365’s Customer engagement applications are Microsoft’s first party applications built on the **Microsoft Power Platform**. Organizations can leverage these applications individually or together to create powerful end-to-end relationships with their customers.

Microsoft Dataverse

As mentioned previously, Dynamics 365’s customer engagement applications are built on the Microsoft Power Platform. The Microsoft Power Platform is a low-code application development platform that not only spans Dynamics 365, but also Office 365, Azure, and standalone applications. It combines the power of PowerApps, Power BI, and Power Automate into one business application platform that provides quick and easy app building and data insights.

Microsoft Power Platform

The low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications



One of the key elements to building applications on this platform is Microsoft Dataverse. Dataverse lets you securely store and manage data used by your business applications. For example, you might store all your customer data in Dataverse. Dataverse data is stored in tables. A table is a set of rows and columns. Each column in a table stores a specific type of data such as names, locations, ages, dates, salaries, and so on.

Dataverse includes a base set of standard tables that cover typical business scenarios such as accounts, contacts, and activities. The advantage to Dataverse is that organizations can also create custom tables specific to their needs and populate them with data.

For example:

- A real estate company might create tables to store the properties they sell, represent open houses, or store showings.
- A financial company might add tables to represent loan applications or bank accounts.
- An auto repair company might add tables to represent the parts they sell or the services they provide.

Application makers can use tools like Power Apps to build rich applications that leverage this data.

Leveraging Dataverse provides these benefits:

- Easy to manage: Both the metadata and data are stored in the cloud. You don't need to worry about the details of how they are stored.
- Easy to secure: Data is securely stored so users can only access what they need to. Role-based security allows you to control access to tables for different users within your organization.
- Rich metadata: Data types and relationships are used directly within Power Apps.
- Logic and validation: Define calculated columns, business rules, workflows, and business process flows to ensure data quality and drive business processes.
- Productivity tools: Tables are available within the add-ins for Microsoft Excel to increase productivity and ensure data accessibility.

Dynamics 365 first-party customer engagement applications such as Dynamics 365 Sales, Dynamics 365 Customer Service, or Dynamics 365 Marketing use Dataverse to store and secure their data. This enables you to build apps by using Power Apps and Dataverse directly against your core business data, which is already used within Dynamics 365, without the need for integration.

Examine the customer engagement apps

Dynamics 365 customer engagement applications help organizations fundamentally reimagine how they engage with customers. Organizations can create personalized marketing, sales, and service experiences using data and intelligence to improve every interaction.

There are five core Dynamics 365 customer engagement applications:

- **Microsoft Dynamics 365 Sales:** Go beyond sales force automation to better understand customer needs, engage more effectively, and win more deals.
- **Microsoft Dynamics 365 Customer** Service: Built-in intelligence delivers faster, more personalized service and adds value to every interaction.

- **Microsoft Dynamics 365 Field Service**: Built-in intelligence helps you resolve service issues before they occur, reduce operational costs, and deliver positive on-site experiences.
- **Microsoft Dynamics 365 Project Operations:** Build trusted customer relationships and deliver outstanding project experiences by delivering profitable projects on time and within budget, while increasing employee productivity.
- **Microsoft Dynamics 365 for Marketing:** Find and nurture more sales-ready leads by moving beyond basic email marketing. Connect sales and marketing, automate processes, and make smarter decisions to maximize your marketing Return on Investment (ROI).

When used alone, each application empowers organizations to manage the daily aspects related to that item. For example, Dynamics 365 Sales helps you take control of, manage, and automate the tasks related to your sales cycle. However, when combined with Dynamics 365 Customer Service, you now have a complete solution that ensures you are providing high levels of service to your customers after they are sold. Service-related events can be used to trigger sales-related events so sellers can get out and sell to customers. Even though each application provides a specific type of functionality, they each share common elements to provide a unified experience whether you are using them separately or together.

Work with customers and activities

Completed

As mentioned previously, a customer represents an individual or company that receives, consumes, or buys a product or service.

There are three ways organizations define customers:

- **Business-to-Business (B2B):** Customers are other businesses. For example, a business that sells and implements telecommunication equipment to hotels.
- Business-to-Consumer (B2C): Customers are individuals. These are businesses you likely come in contact with every day through activities such as purchasing groceries, getting a haircut, or stopping at a favorite coffee shop.
- Hybrid: Customers might be either businesses or individuals. For example, an automotive repair shop will certainly have customers who are individuals. Customers go there to get their oil changed, tires rotated, engines overhauled, and so on. What if that automotive

business has their mechanics outsourced to organizations with fleets of trucks, buses or automobiles as well? In this example, they service both organizations, and people. Other examples might include: Software Vendors, Event Centers, Insurance Companies.

Dynamics 365 applications support all three customer models. There are two primary records used to support this functionality:

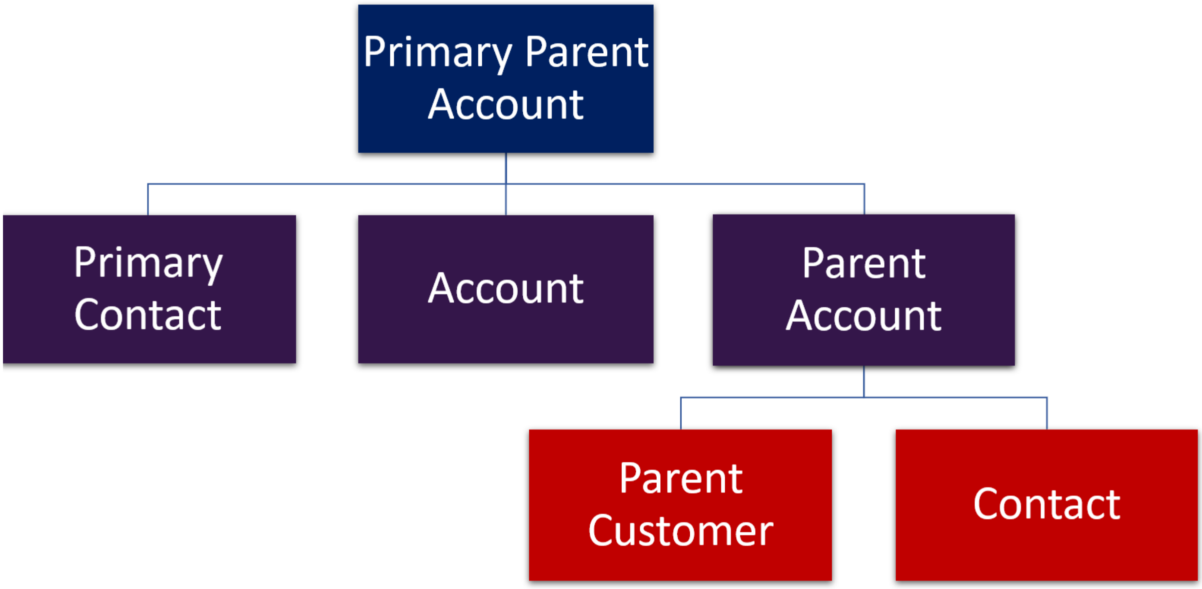
- Account: Represents a company or an organization that you do business with such as a customer, vendor, partner, or reseller.
- Contact: Represents a person. A contact can be a standalone person, or an individual at an organization.

Accounts and contacts can be related to one another in various ways. Individual contacts can be associated with an account. For example, if Microsoft were an account within the Microsoft Dynamics 365 system, then the contacts might include Bill Gates. Each contact record in Dynamics 365 includes the parent customer field that lets you link a contact to a specific account record.

While an account might have multiple contacts, it is likely that it will have a primary contact. The primary contact represents a key person at the organization that you primarily deal with. This might be the person you deal with as it relates to billing, sales, and so on.

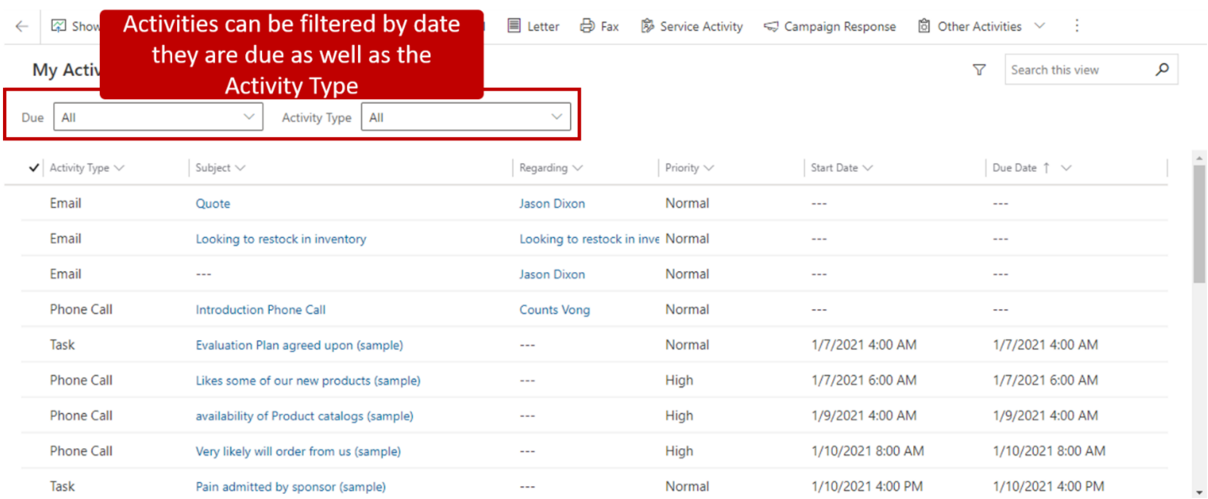
While many accounts will be stand alone accounts, there are occasions where an account can be considered as the parent account to another account. This might occur when one account is a division or subsidiary of a parent company.

The image below shows what this might look like.



Activities

Activities in Dynamics 365 help keep track of customer communications. For example, as you work with customers, you might do things like take notes, send email, make phone calls, set up appointments, or assign yourself tasks as you resolve a service case. By entering these activities in Dynamics 365, you will have a historic record of your communication with customers.

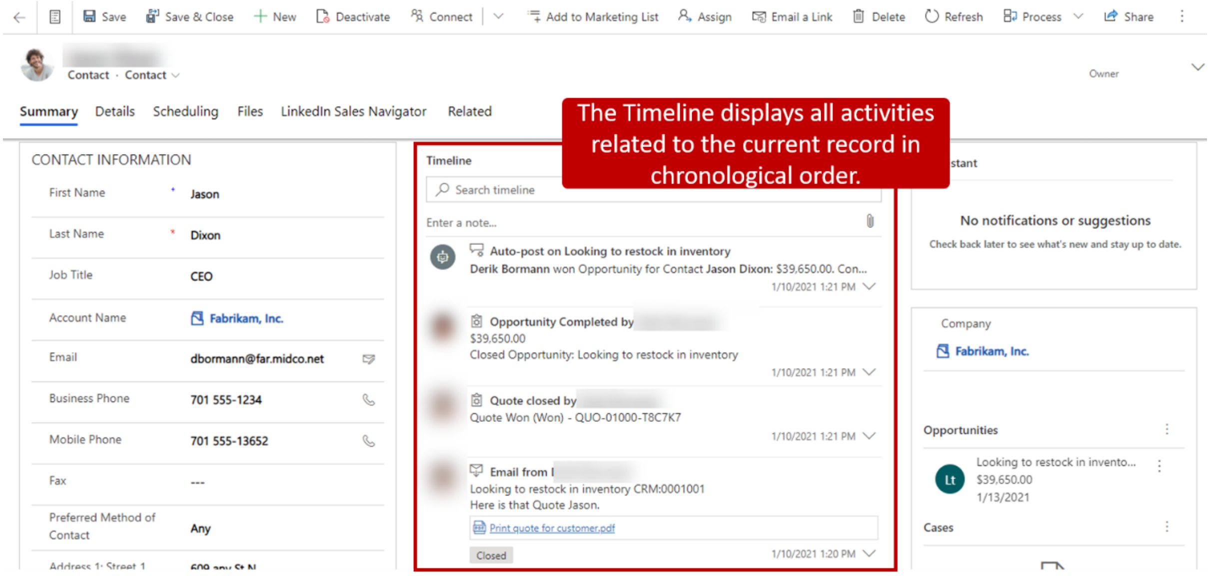


Activities are often associated with contacts and accounts but, depending on the Dynamics 365 application you are using, they can also be associated with other types of records.

- In sales scenarios, activities might be added to records like leads, opportunities, or quotes.

- In service scenarios, activities might be added to records like cases, work orders, or projects.

As your organization's relationship with a customer builds over time, you and other people on your team can look through the record time-line to see the history of your interactions.



Microsoft Dynamics 365 CE Architecture

Microsoft Dynamics 365 Customer Engagement Architecture:

Microsoft Dynamics 365 CE has a N-tier architecture which hasn't changed for a long time.

This N-tier architecture can be extended with the help of UI tools and code means developers can customize the standard CRM functionality by developing workflows, plugins and custom workflow activities.

Knowing the architecture of Dynamics 365 CE is quite important for the developers to extend/customize the application.

The 3 tiers (or layers) of the Dynamics 365 CE architecture are:

1. Web Tier

2. Application Tier

3. Data Tier

Web Tier:

Web Tier handles the web pages of the applications such as CRM entity forms, views, charts, dashboards and all other UI components. The web tier or web layer is configured on the IIS server which is responsible for the rendering of the web components.

Application Tier:

Application tier manages the business logic of the application such as plugin, processes, security and data elements. The access to the application layer is controlled through the web services, means to access any of these components the user has to be authenticated.

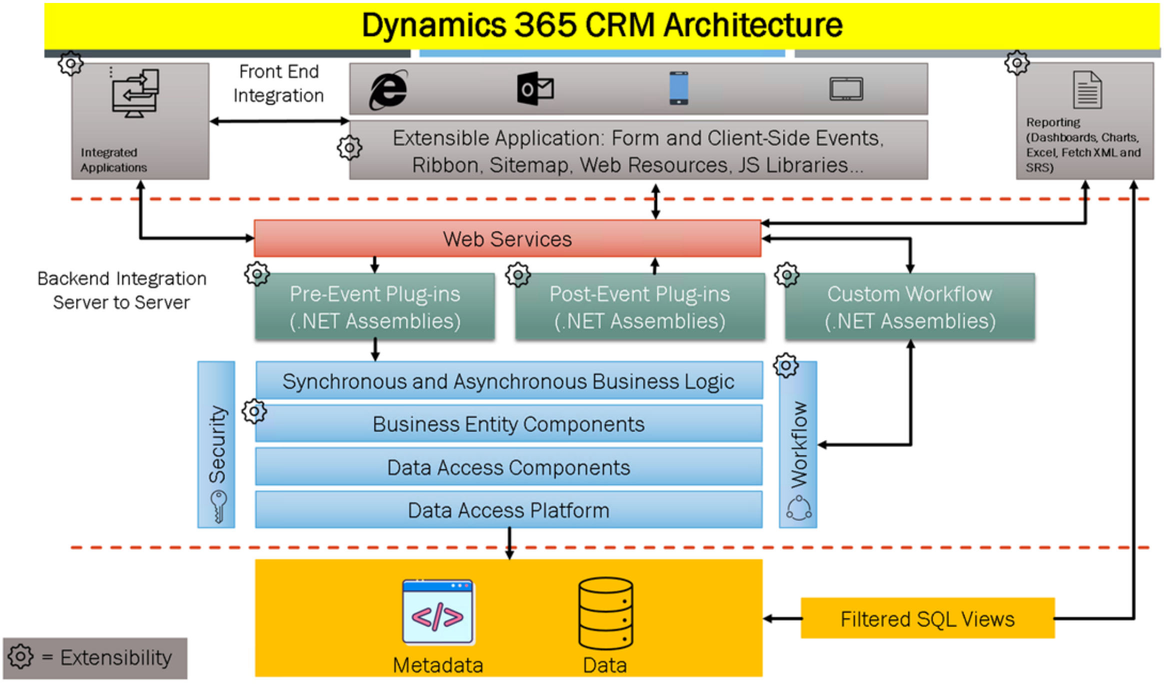
Data Tier:

The data or database layer is responsible for the storing the data and metadata of the Dynamics 365 CE application. The database used is SQL Server and the database will be configured when we install the CE application.

For every entity in the application there will be a corresponding table available in the application.

D365 – Architecture Overview

Following is an overview of D365 CRM architecture.



Understanding the architecture is important to understand the entire ecosystem of development tools here.

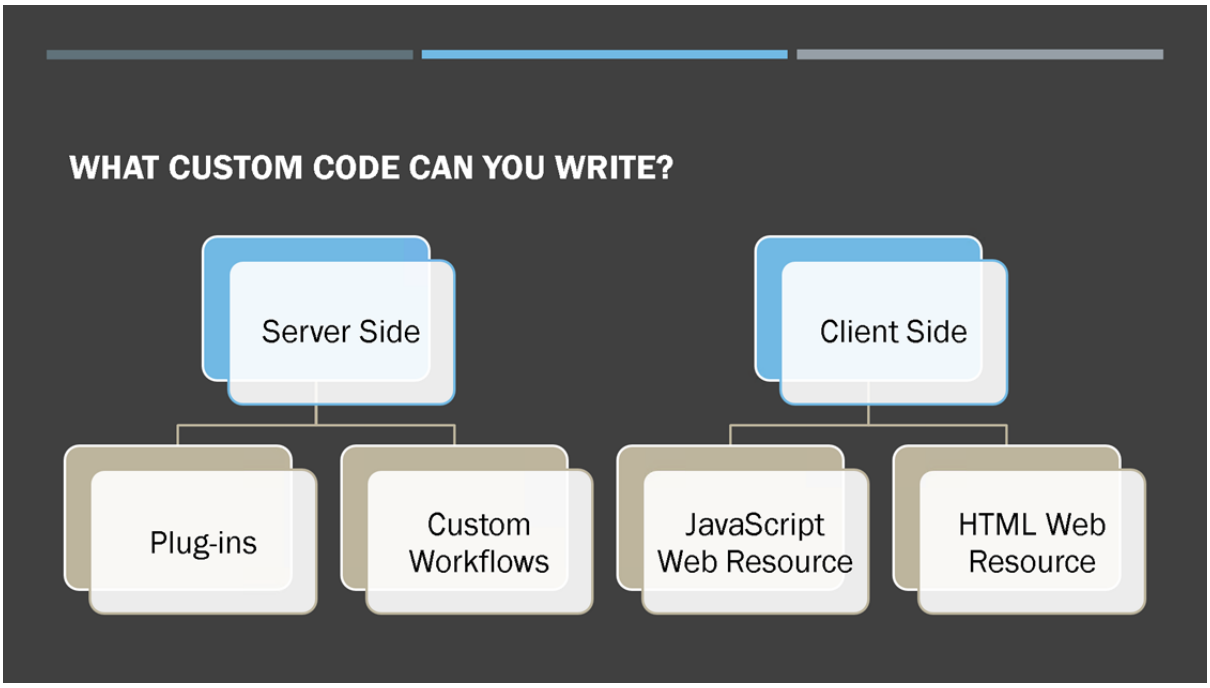
- So the first dotted line separates the client side components from the server side components.
- Wherever you see the gear icon, those components are customizable. So you can write some custom code where you see the gear icons and these components are available on client side as well as on the server side.

- Every D365 CRM instance has Metadata database and Data database. We don't need to do much with it as the platform takes care of these databases.
- Also, you cannot directly access or make changes to the database with any code.

What Custom Code can you Write?

You can broadly categorize the custom components into following types:

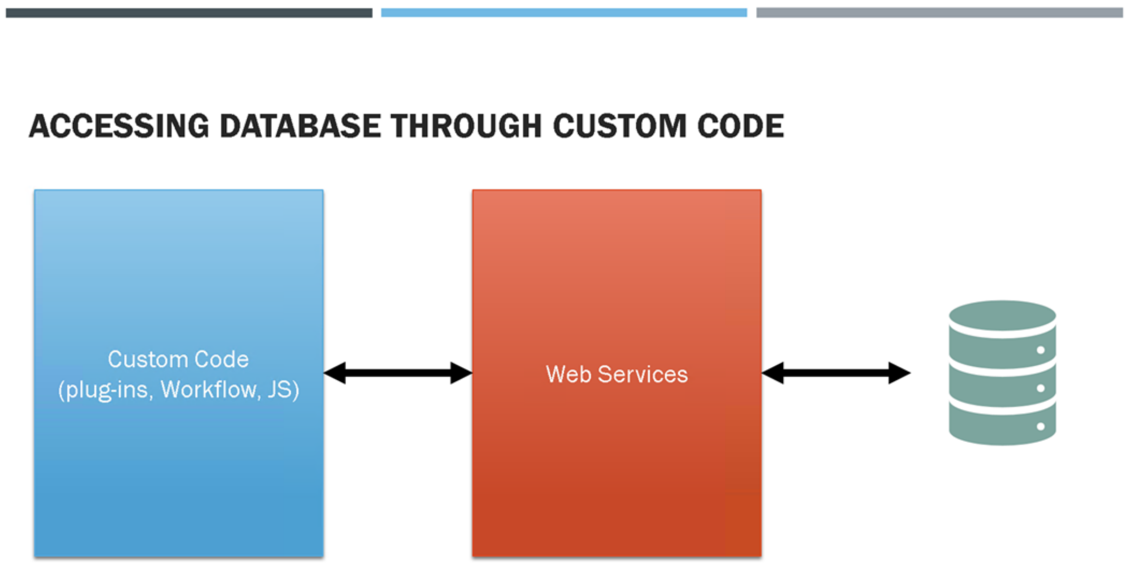
- Server Side
- Client Side



How does Custom Code access CRM data?

Custom code cannot access data directly from the D365 CRM database.

It can access data through API (Webservices) only. The connection between web services and databases is taken care of by the platform.



Second Week Content

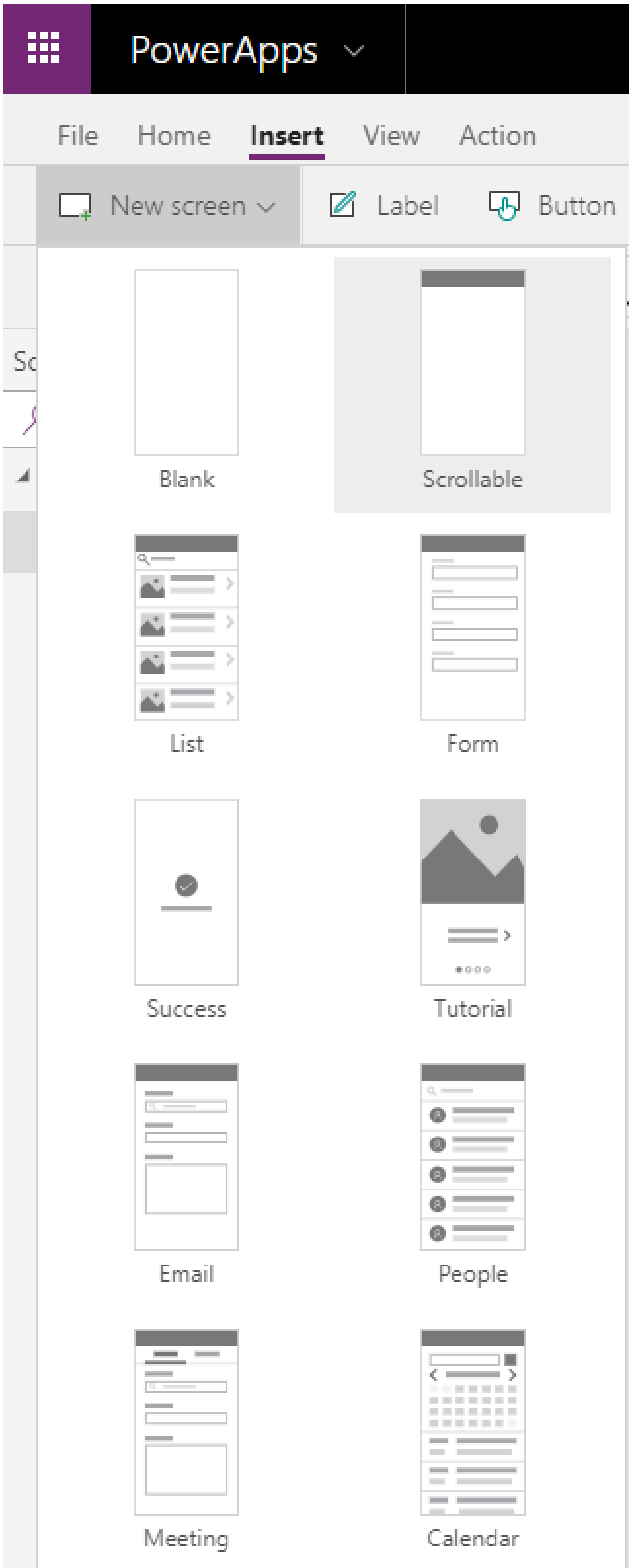
6 major components of PowerApps

POWER APPS

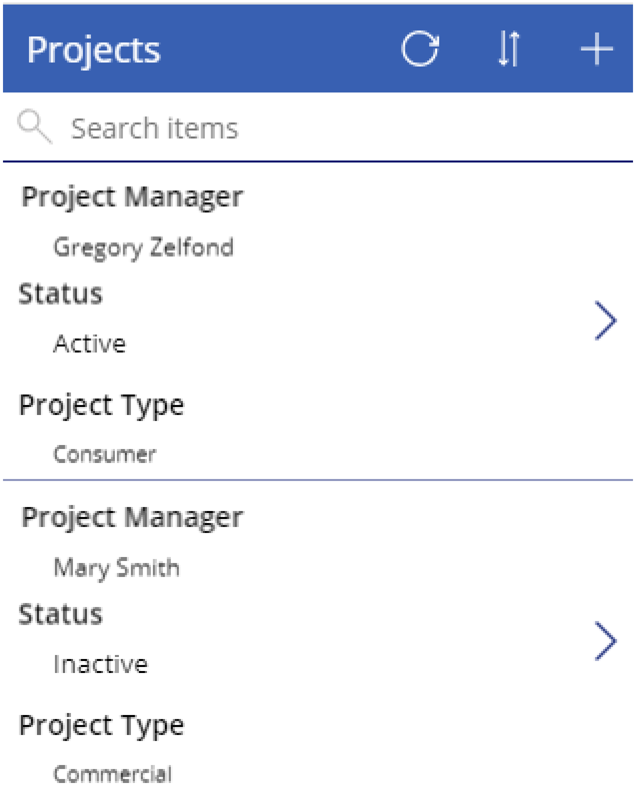
If you want to embrace PowerApps, you first need to understand its anatomy and basic elements. With this post, I would like to highlight, what are, in my opinion, the major components of PowerApps one needs to understand first before creating the first app.

1. GALLERY

A **gallery** is a way of visualizing data in the app. It is a template of screens that allows you to see and navigate the data. For example, you might have a gallery that contains a screen to see all records, then a screen to view a given record and a screen to edit a record.



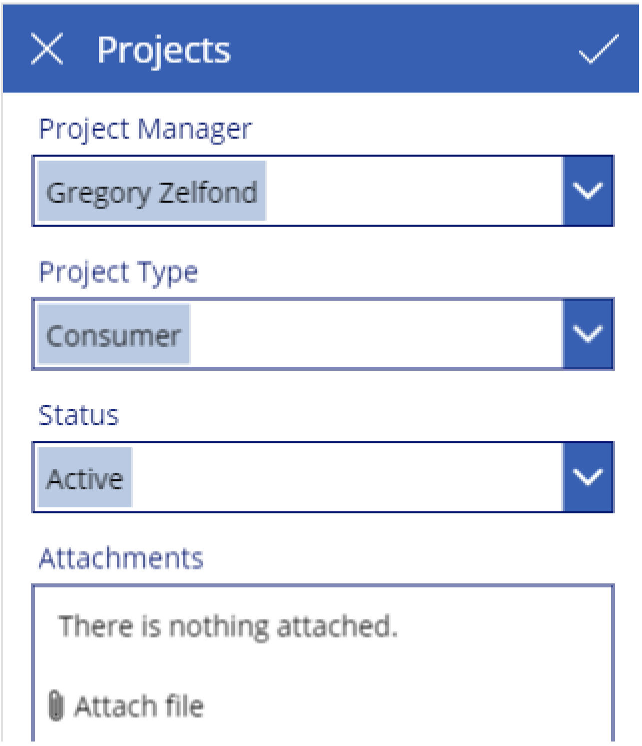
All of the screens combined make up a **gallery**. You have different galleries to choose from when you create a new app. When you create an app from SharePoint list, you get a default gallery already pre-built for you.



Example of a Screen based on a certain Gallery template

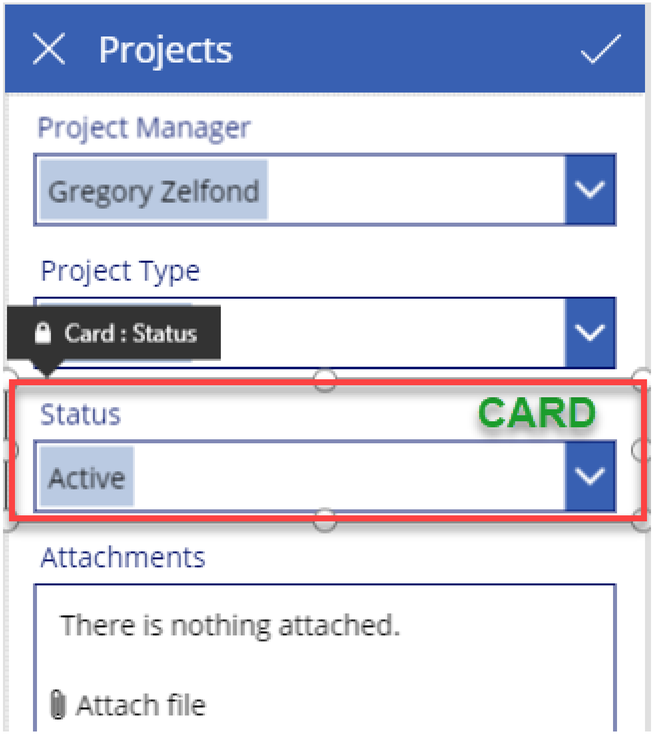
2. SCREEN

As alluded to in the previous section, a **screen** is a way to visualize a particular data set or record on a screen (mobile, iPad, Desktop). You typically have one to view all records, one to view a particular record, one to edit. Once again, when you create an app from the template, you have screens already pre-built for you (they are part of a gallery you choose above). But you can add additional screens if necessary.



3. CARD

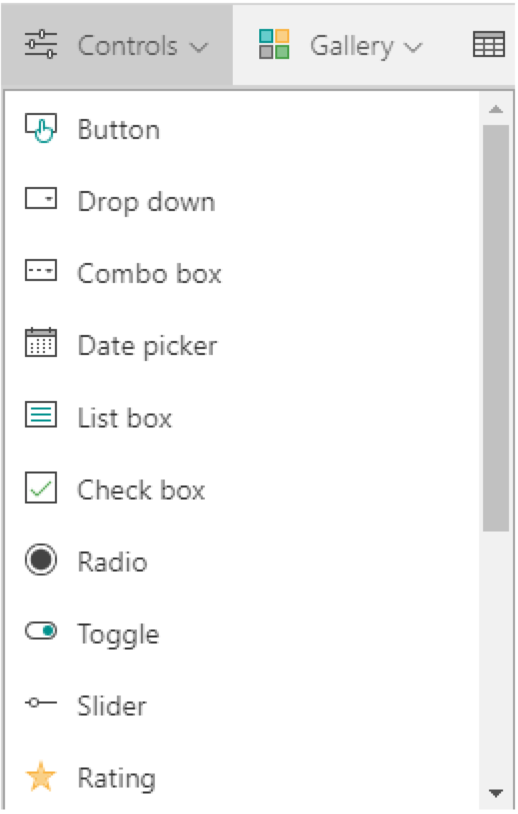
A screen consists of **cards**. A **card** is an area on the screen that shows a given record from your **SharePoint list** or any other database you used to build an app. For example, if you had a SharePoint list storing project names and built a PowerApp from this list, a single card might contain a **Project Name** field or **Project Manager** field or **Project Type** field or a **Status** field.



A card would contain all various attributes (called **Controls** in PowerApp – described below) related to the display of the record. For example, for **Status** field depicted in the image above, a card contains a **Drop-Down control** to enter Status as well as **a label** called "Status".

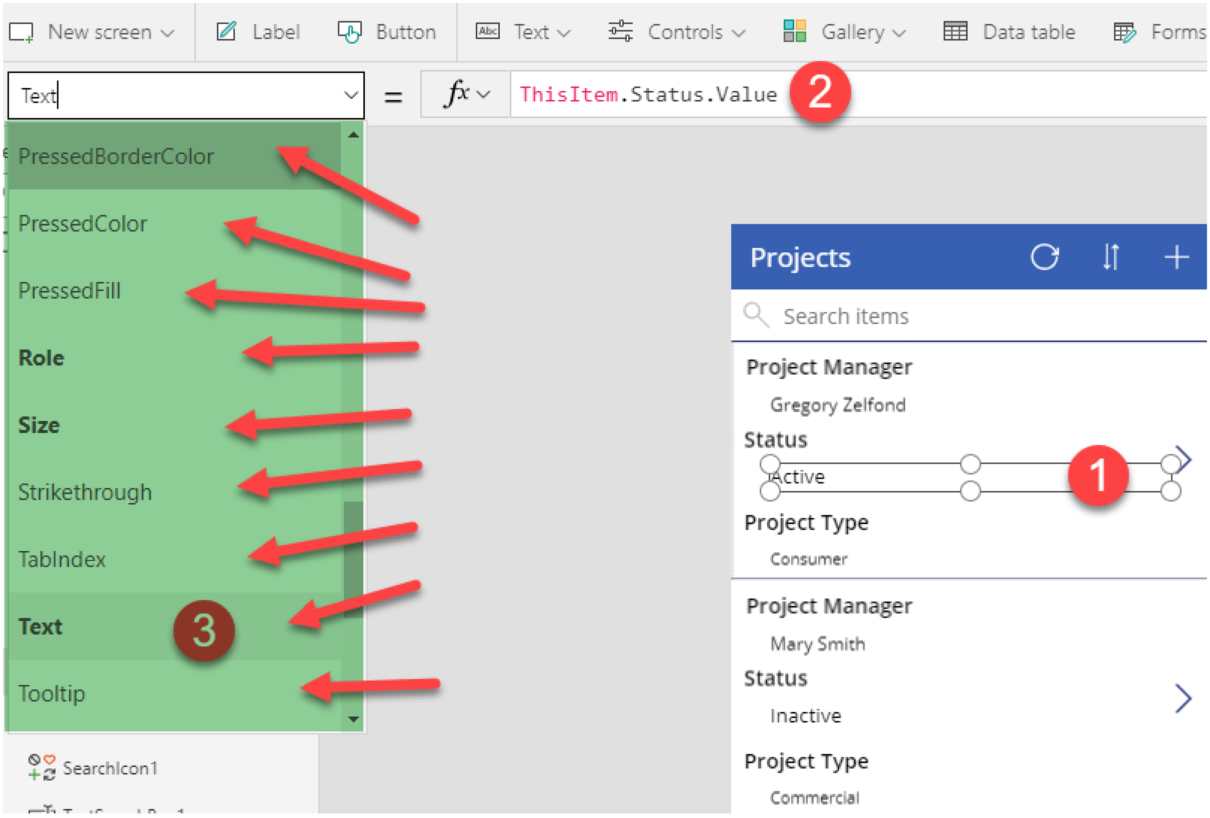
4. CONTROL

Controls is what allows you to visualize and interact with your records. Using the above **Status** field example, the **Drop-Down** to choose Project Status **is a control**. Depending on the type of your field, you can have different types of controls. For example, instead of a drop-down, I could have a radio button or toggle switch to make selection a bit more elegant on the phone app.

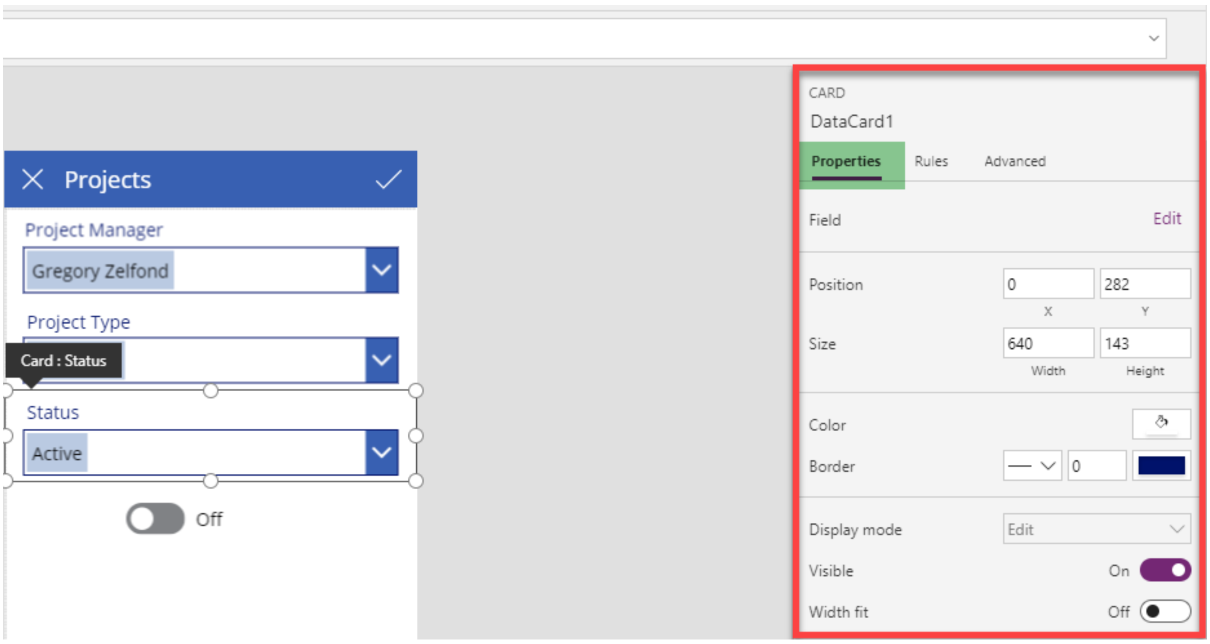


5. PROPERTY

Each control has **properties**. For example, a text entry box has a property for font size, text color, text box fill color and so on. Properties can be accessed and changed from the Properties drop-down on the left-hand-side of the screen as well as on the panel on the right-hand-side once the property is selected.



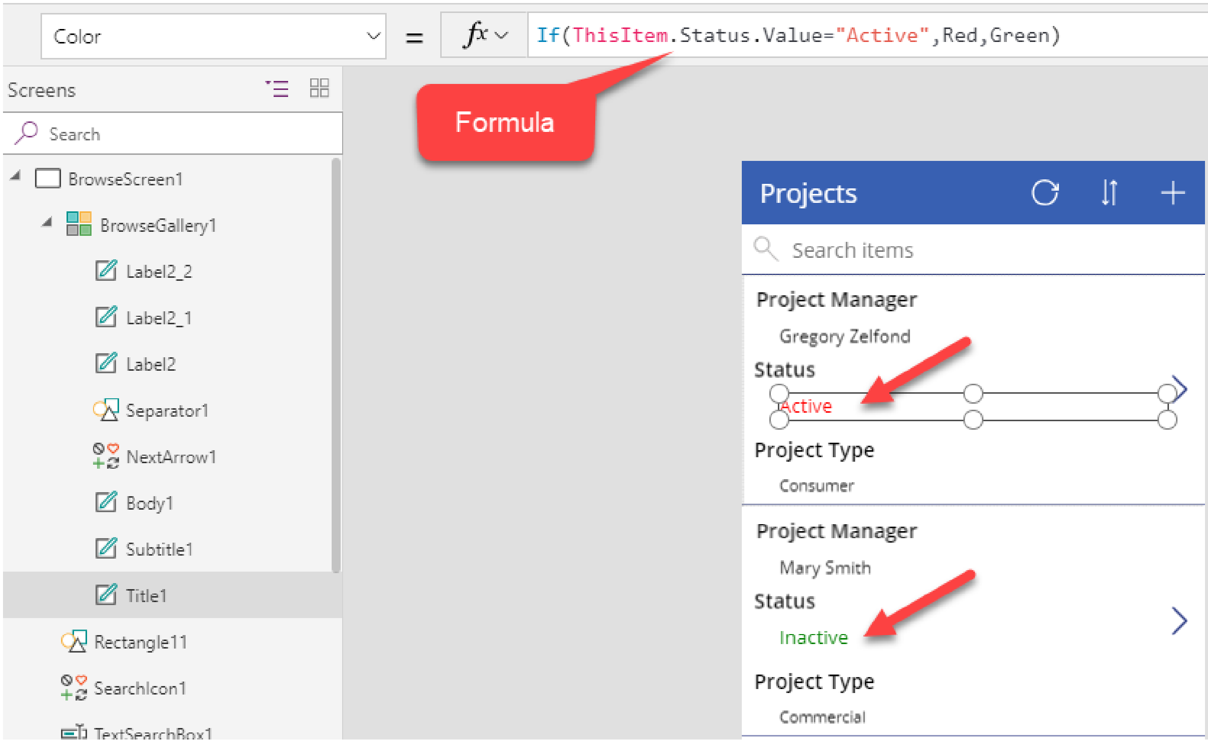
All the arrows point to properties of a certain control



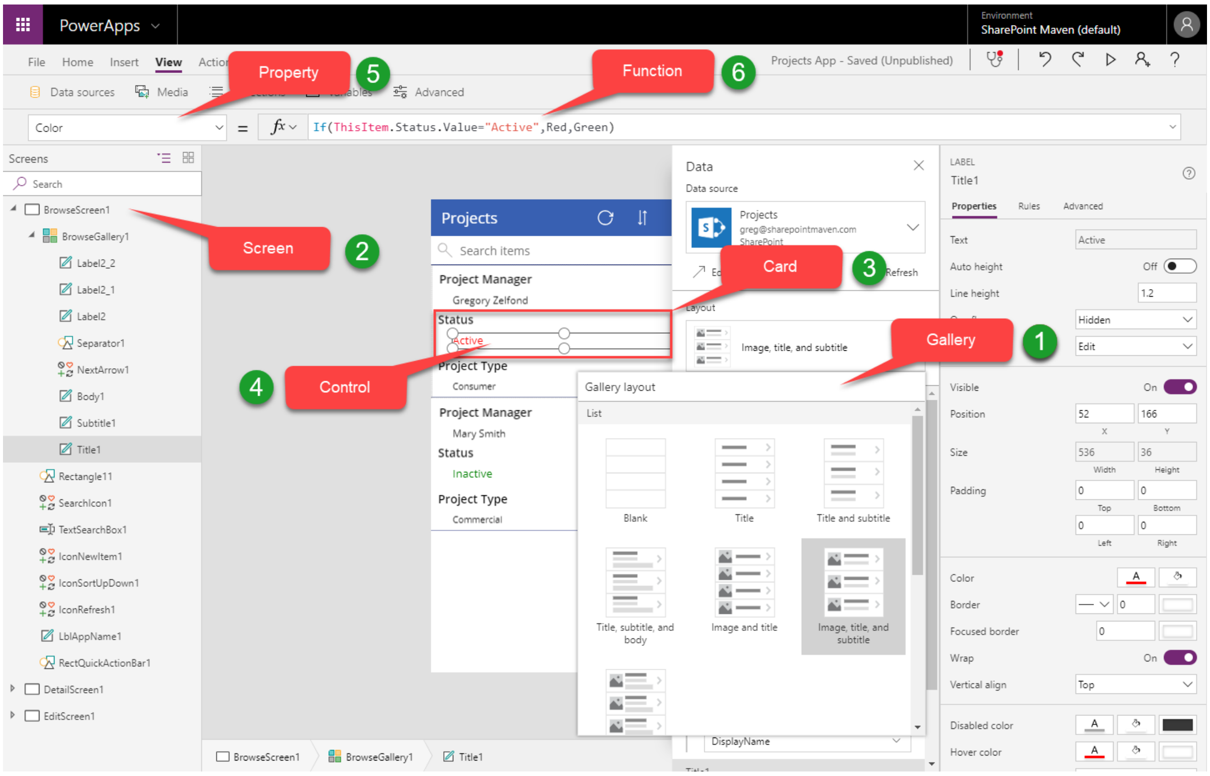
Another way to edit properties for the card

6. FUNCTION

Functions is how you interact with and change the properties. If you are familiar with Excel, you are in luck as the syntax for PowerApps functions is kind of similar to Excel. Below is an example where I use the **Formula** (If Statement) to change **Property** (Color) of a **Control** (Text) depending on the value of the field.



And now, putting it all together:



Tweet

<u>**Custom Components in PowerApps**</u>

PowerApps can be a powerful tool for creating fast prototypes, small business apps, and proof of concepts. There are a lot of built-in components that are useful but often require a lot of configuration if you want more than just the default functions and style.

Enter PowerApps components. Components let you create your own reusable components which can be exported and imported to different projects, and make maintenance a lot easier for developers.

Components are useful in building larger apps that have similar control patterns. If you update a component definition, all instances in the app reflect your changes. You can also improve performance by using one or more components because you don’t copy and paste controls, which duplicates overhead.

Components are isolated instances that are decoupled from your app. This means that a custom component can’t access variables declared in your app unless they are parsed as properties into the component.

custom connectors in Power Platform

Power Platform has become one the most set of tools that many organizations are leveraging. Power Platform has the robust power of **Power Apps, Power Automate** and **Power BI** into one powerful business application platform that helps citizen developers as well as pro developers in providing quick and easy app building and data insights.

Microsoft Power Platform is low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications a shown below

Microsoft Power Platform is low-code platform that spans Office 365, Azure, Dynamics 365, and standalone applications a shown below

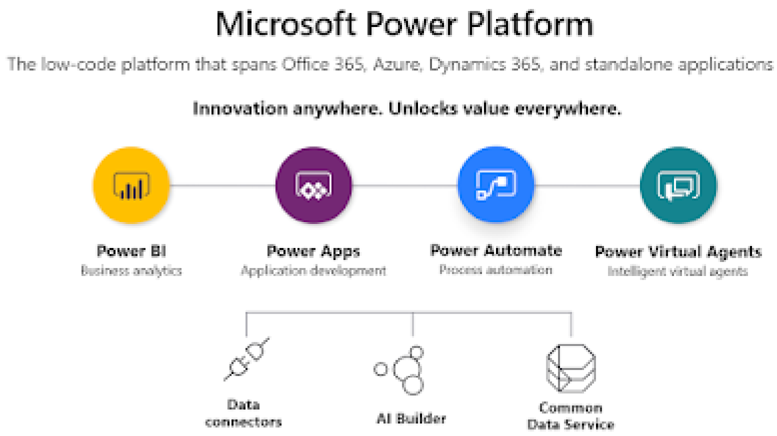


Figure 1 Credit: Microsoft

The best thing about Microsoft Power Platform is that it's catered not only for **citizen developers** but also for **professional developers** as shown:

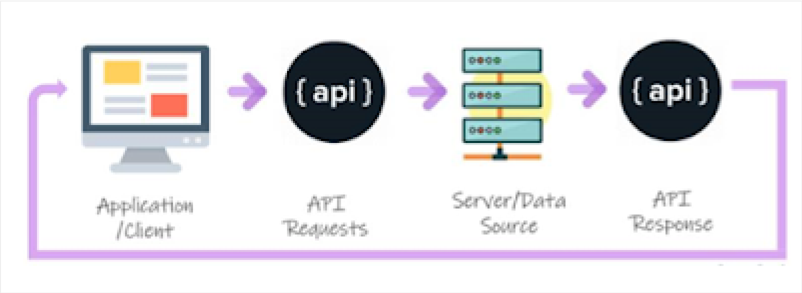
Why needs an API in Power Platform?

Power Platform leverages Microsoft services to connect to other services such as SharePoint, OneDrive as 3rd party services such as **MailChimp**, Adobe Sign etc.

What is an API?

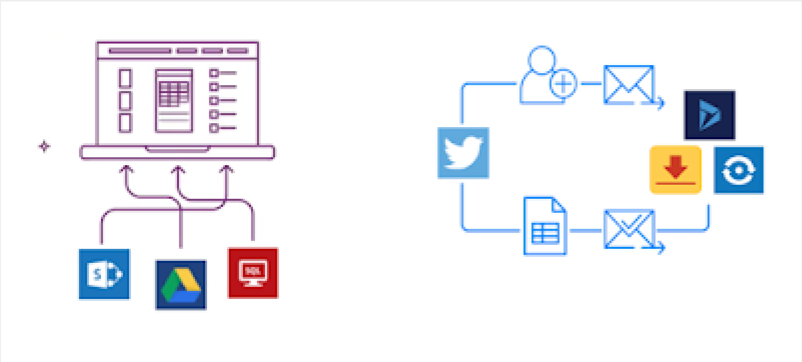
API (Application Programming Interface) that allows **two applications** to talk to **each other**.

for e.g. A user initiates **Power Apps** (Application) to get a signature **leveraging Adobe Sign**, this will be the API call from Application (Power Apps) and **Adobe Sign as a data source** as shown below.



What is a connector?

A connector is a **wrapper around an API** that allows the underlying service to talk **to Power Automate** and **Power Apps.**

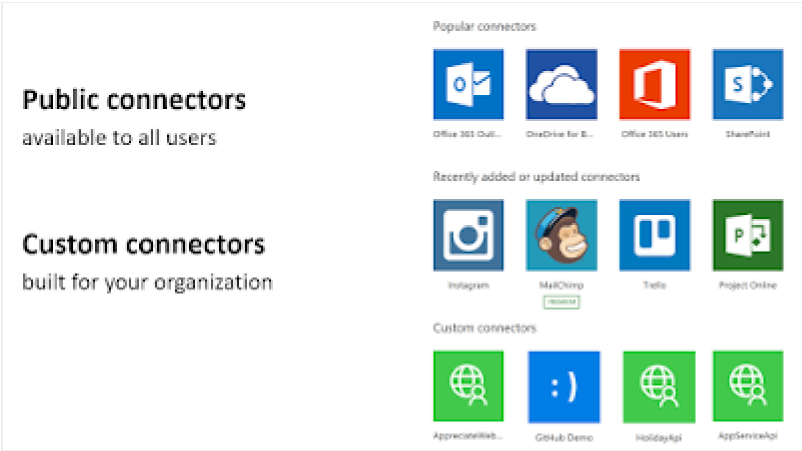


Type of connectors

You have 2 types of connectors.

1. **Public connectors (available to all)**

2. **Custom connectors (built for your organization)**



Introduction of Variables

With the introduction of PowerApps, more and more users are moving into PowerApps development for easy and no-code development tools. However, users from a coding background found it difficult, as there were no variables in PowerApps. Not now, with the introduction of variables in PowerApps, it has become a Powerful no-code tool. So let's understand PowerApps Variables and how they works.

Types of Variables

Just as we have different scopes of variables in the programming language in PowerApps, we also have different scoped variables the three different types of variables which are available in PowerApps, as mentioned below:

Global variable	App	This is similar to a global variable in programming language and it can hold a number, text string, Boolean, record, table, etc as a data value	Set
Context variables	Screen	This is similar to the parameter which we pass to methods or procedure and this variable can be referenced from only one screen	UpdateContext Navigate

Collections	App	This variable can store table which can be referenced across anywhere in the app	Collect ClearCollect

Creating and Removing Variables

In all programming languages, we declare a variable before using the variable but such is not the case with PowerApps. So, in PowerApps we never declare variables explicitly. In terms of typing, PowerApps has implicit typing. So, we do not require to specify the type of variable for example int a = 0 in programming language indicated the variable is of type integer but in PowerApps we do not require to specify the type of variable.

To create a variable, we just need to run the function:

- Set(global Variable, "Hello World")
- UpdateContext({context Variable:"Hello World"})
- Collect(collection Variable, "Hello World")

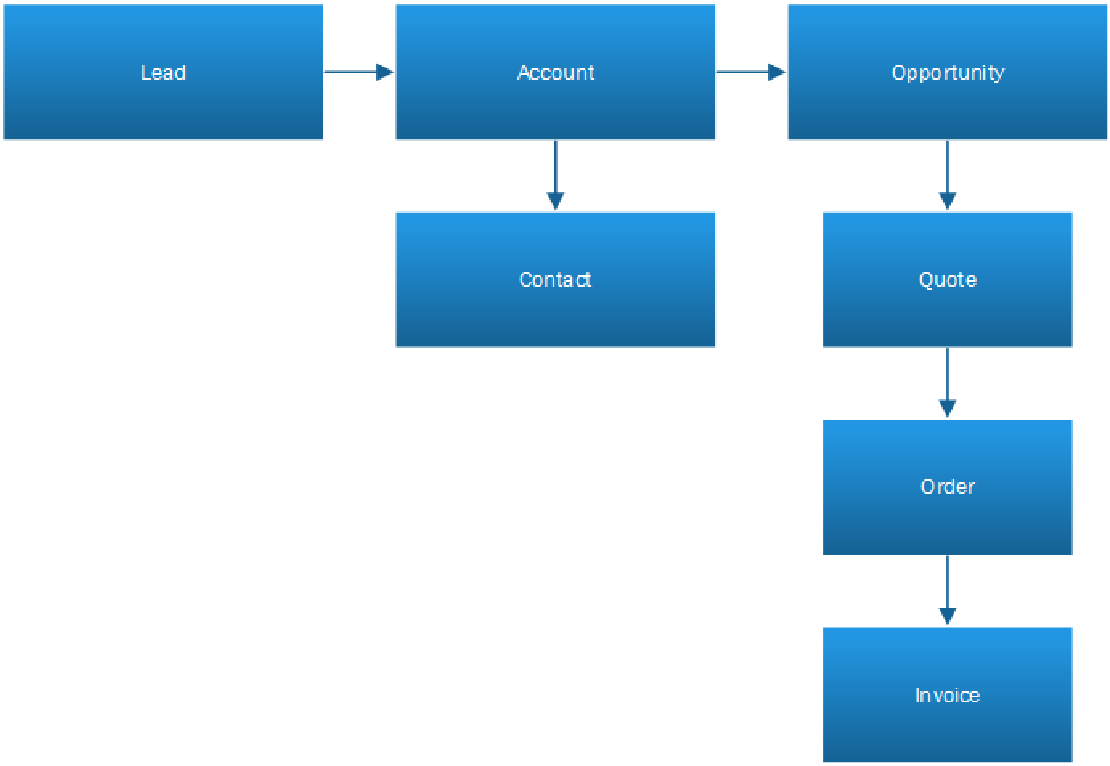
Where X is the variable name and Hello World is the value of that variable.

For removing the variable, we are required to remove all Set, UpdateContext, Navigate, Collect, or ClearCollect functions that implicitly establish the variable. If these functions are not present then there is no variable. We must even remove the variable reference if used in other places.

3rd Week Content

Sales life cycle

Generally, the sales life cycle starts with Lead generation. A lead is your potential customer. When an individual or a company shows any interest in your product/service and provides some details for further communication, then it is considered a lead:



The salesperson either manually enters the details of the lead into the CRM or captures the details through mails, websites, phone calls, campaigns, and so on. It requires a lead name (first name and last name) and a topic that describes the lead.

After lead creation, the salesperson does multiple activities to convert it to a customer. This can be done in form of appointments, …

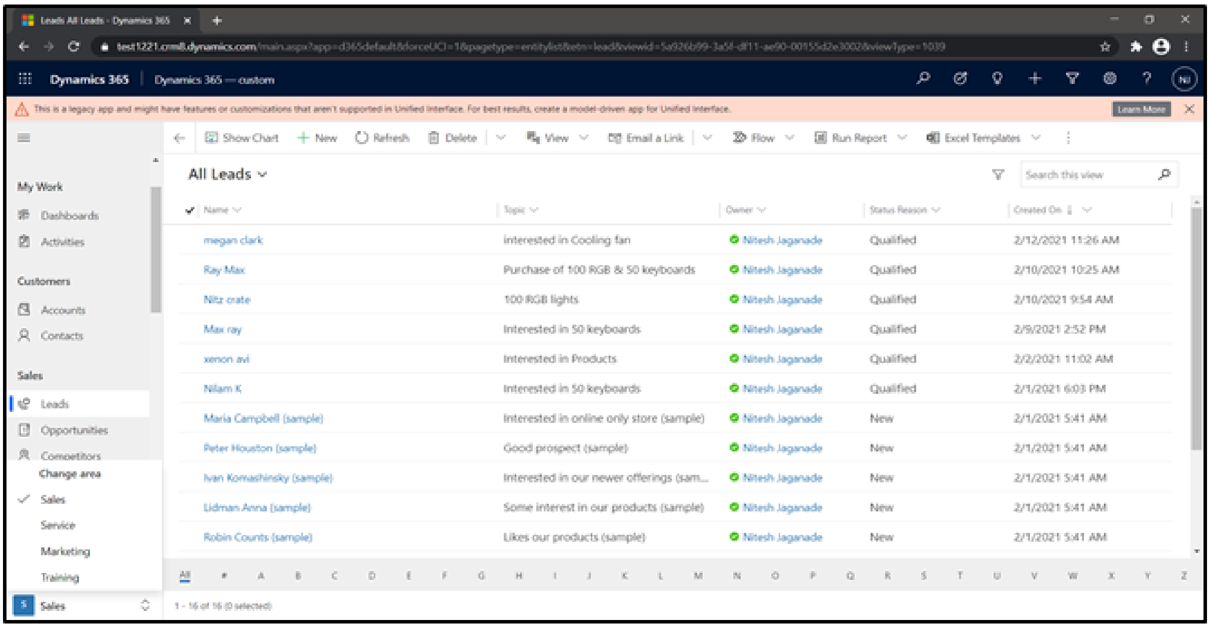
The Accounts & Contact:

Accounts are entities with which your organization has a relationship. In Dynamics CRM, this is where all account information is stored in the database. Accounts can include prospects, vendors, business partners, and more.

Contacts are individuals with whom your organization has a relationship; these are generally customers or contacts of customers. Contacts are often related to an account, but certain organizations and businesses may serve or sell to individual consumers as well.

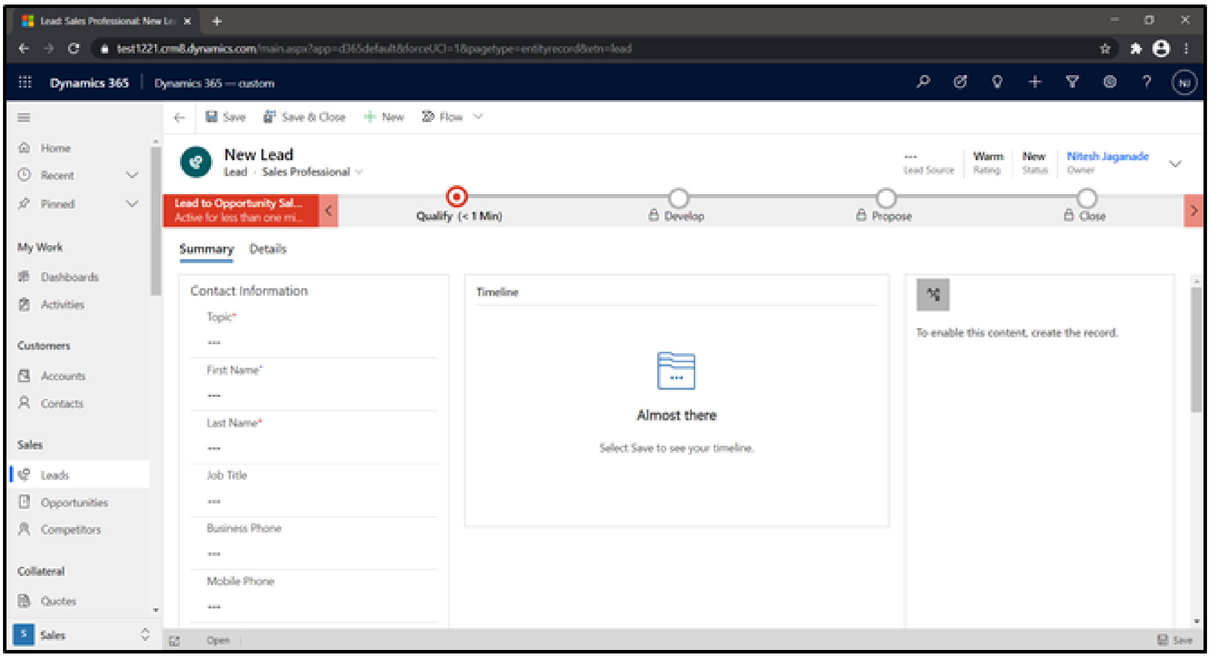
Navigation steps for [Sales Process Life cycle](#):

First Select Area -> Sales, from left bottom (change area)



Navigate to Leads -> click on new from the top

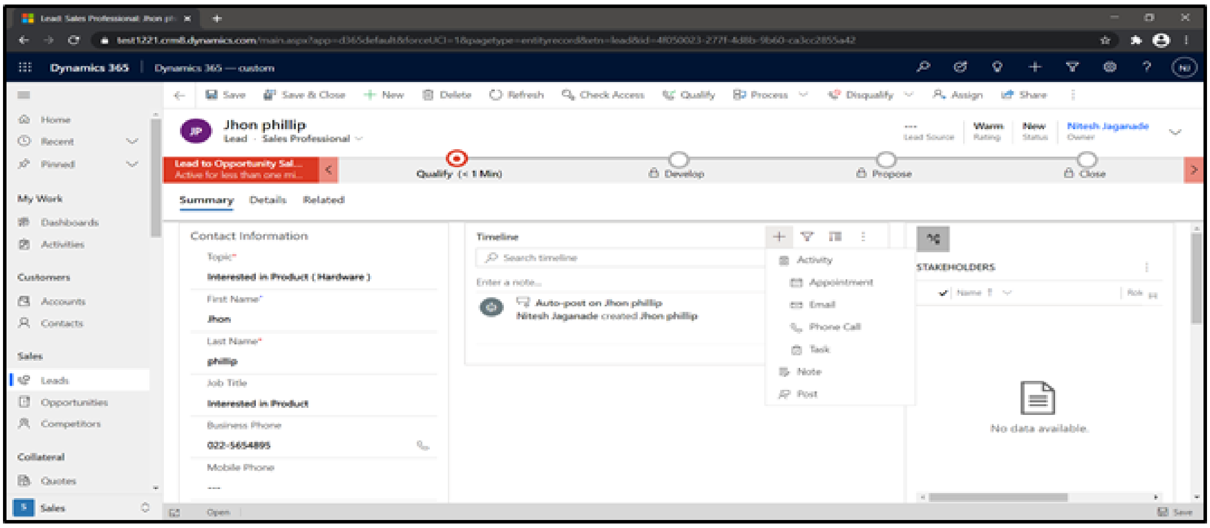
(form will appear regarding lead information)



Fill all the details of a Lead then click on Save button.

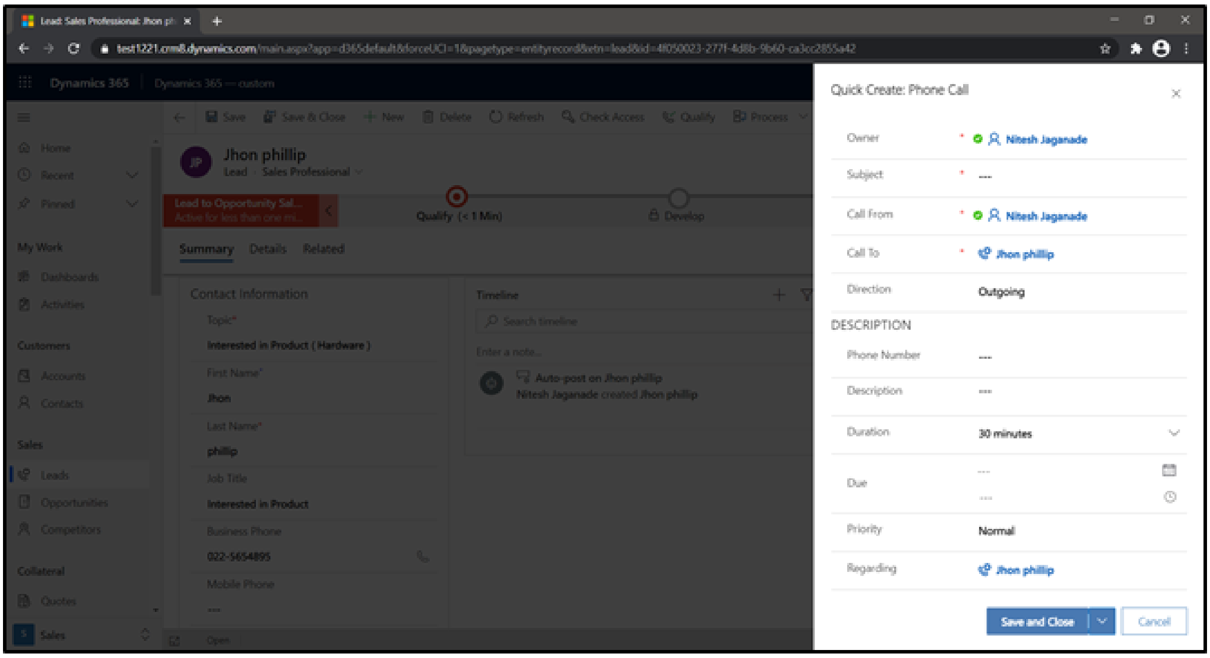
After saving the details of lead it will get converted into An Opportunity based on approach.

Using the Activity options from Timeline we can set call, email, appointment & task.

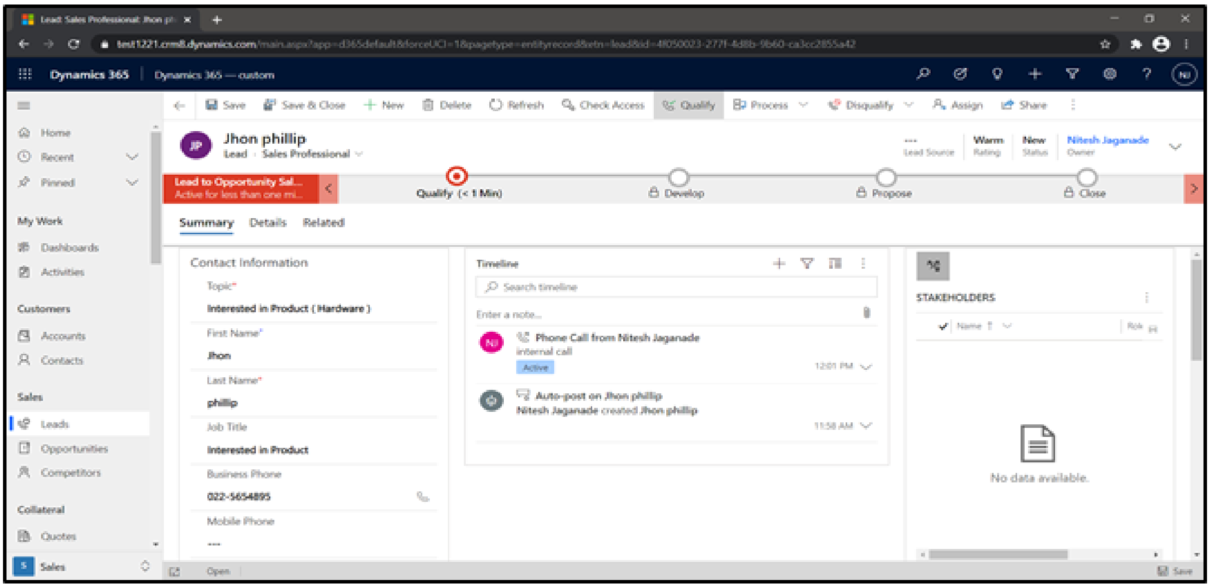


For example, if we try to create a phone call with the lead.

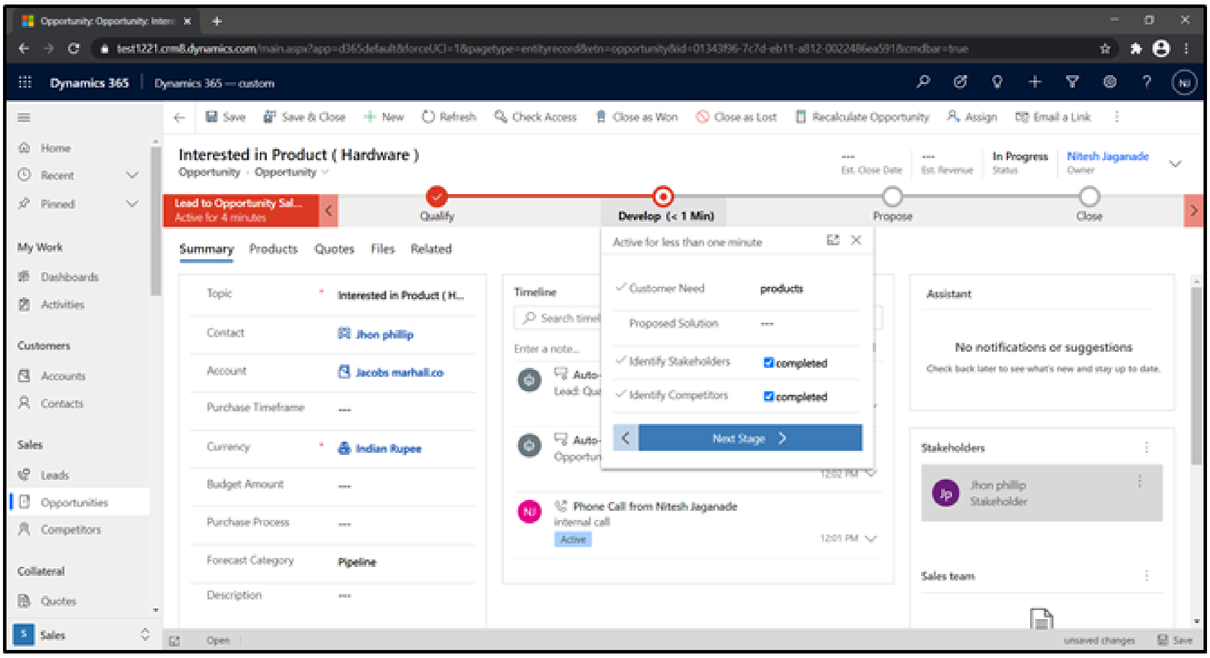
Activity->Phone call (Quick create form will open)



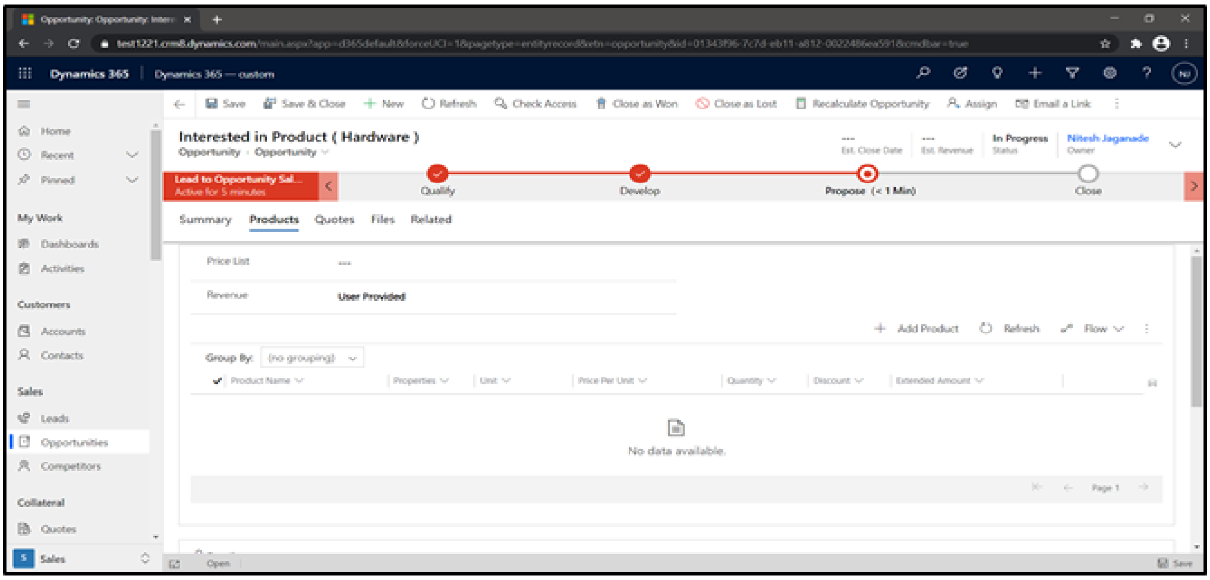
After entering all the call details ->Save and close. All the updated tasks will now appear in Timeline. (as shown in image)



After confirming, select Qualify from top tab ->click on Develop from BPF & complete all requirements -> next stage.

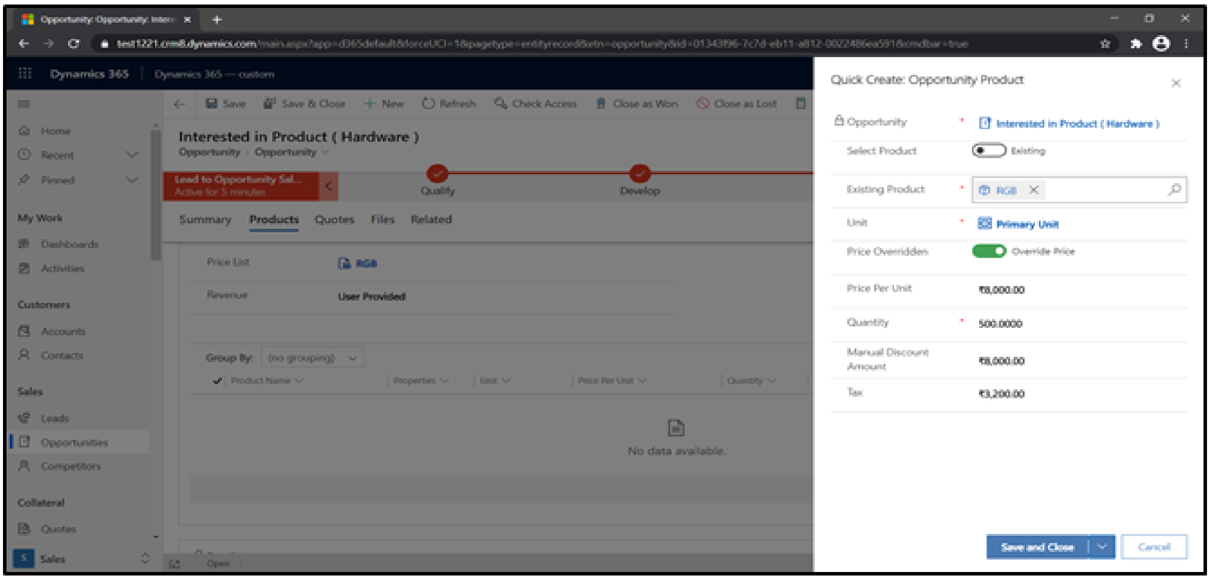


Now the Product Tab will appear beside summary, select Products.



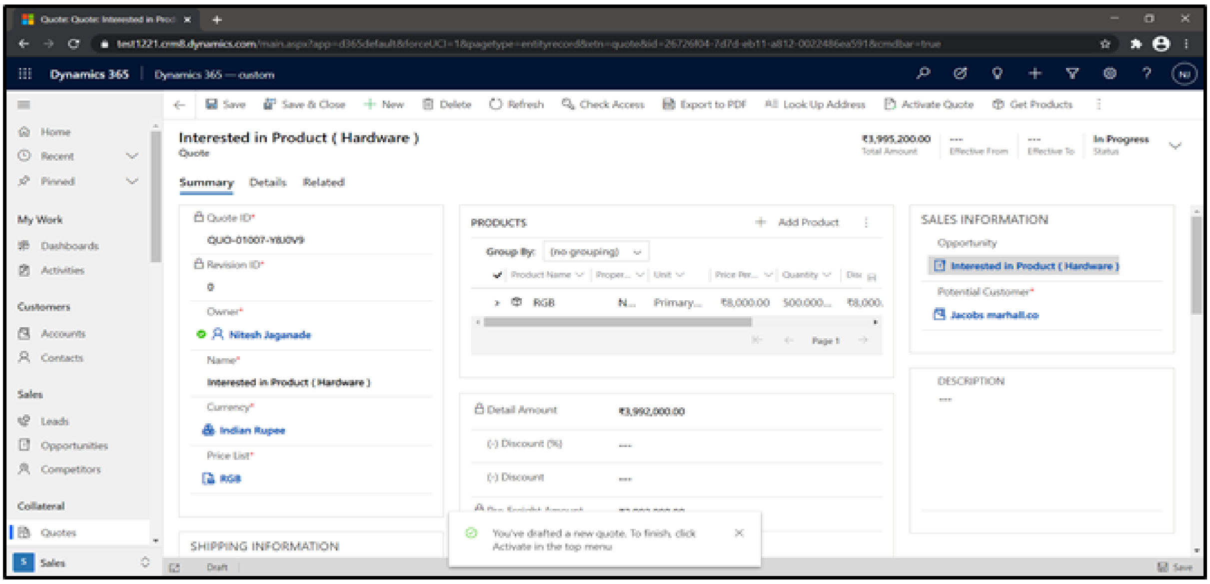
Set price list->Revenue ->add Product.

After selecting add Product, the Quick create form will appear (as shown in image).



Fill in all the required details of Product & Quantity-> Save and Close.

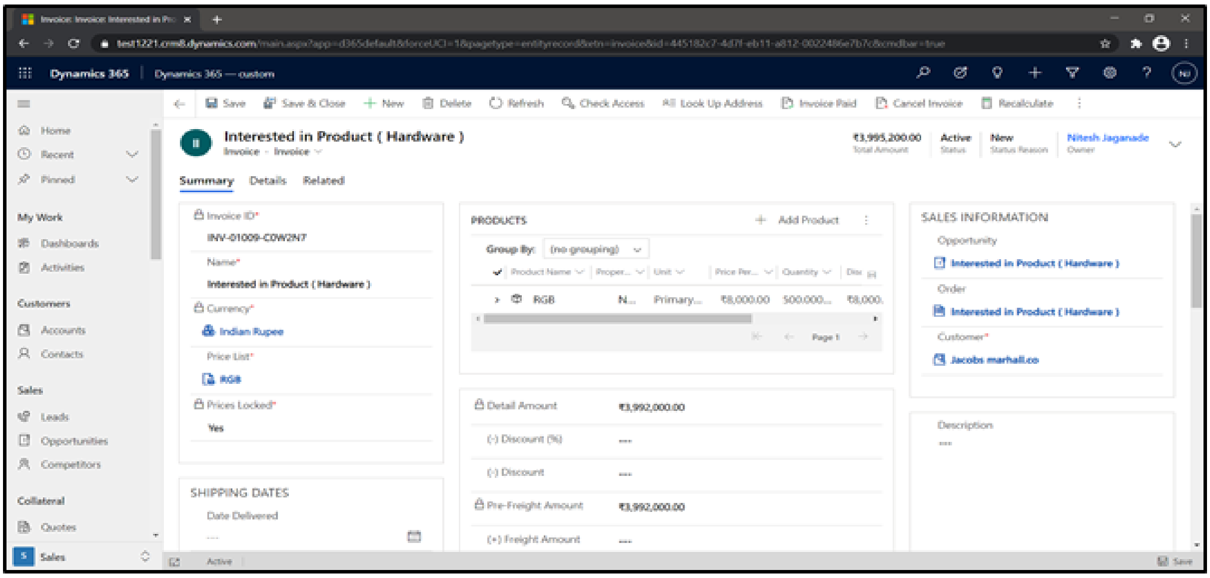
Quote will appear (as shown in image).



Select Activate Quote from top-> Propose the Quote after getting approval for it -> click on Create order.

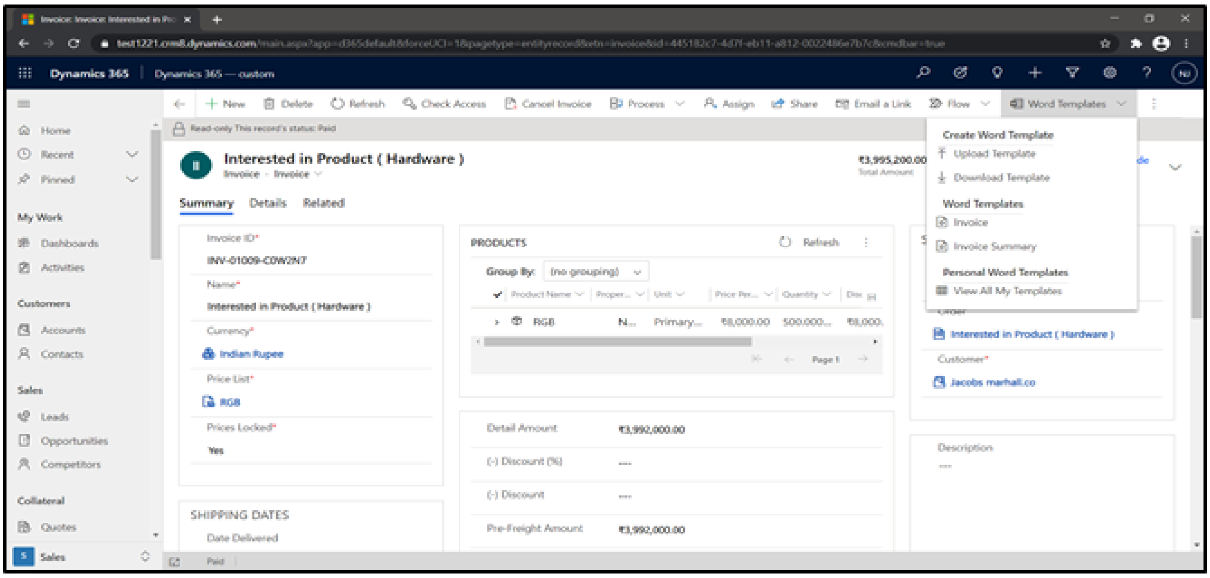
Create order -> Full fill order (after fulfilling order)

Select Create Invoice, it will get created (as shown in image)

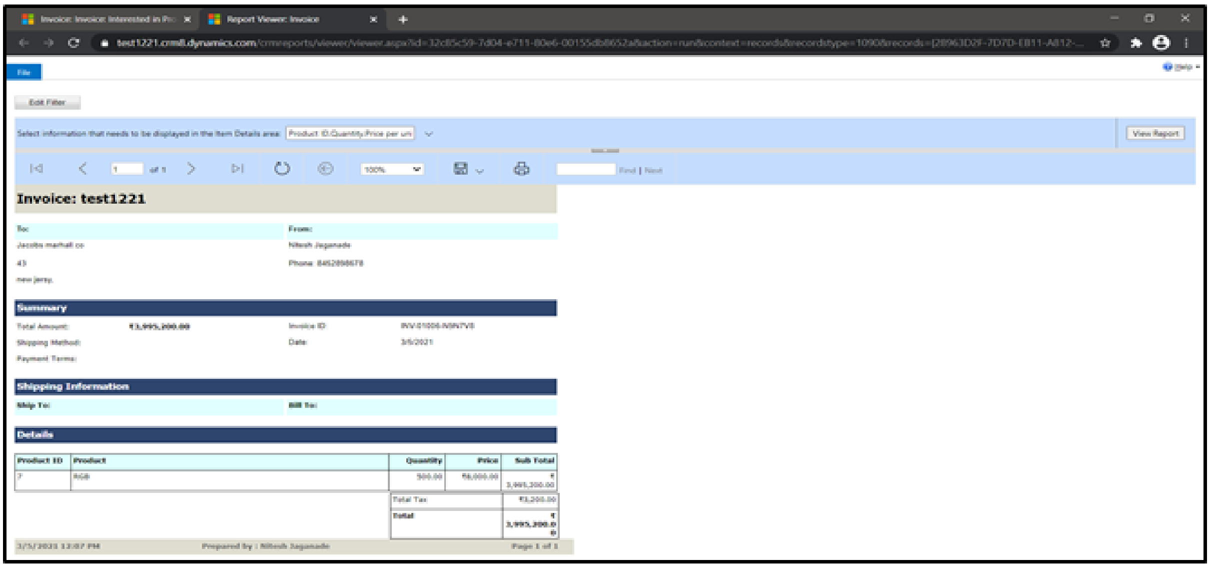


After getting paid for certain order, select Invoice Paid from the top (also the amount paid status can be set from there).

From the word template, we can get all the templates.



After selecting an invoice from the list, the invoice will appear, you can view it in the default viewer or save it to the system.



The Sales Process Flow gets complete.

Microsoft Dynamics 365 with outlook integration

In Microsoft Dynamics 365, Users can associate E-mails with their Dynamics Instance.

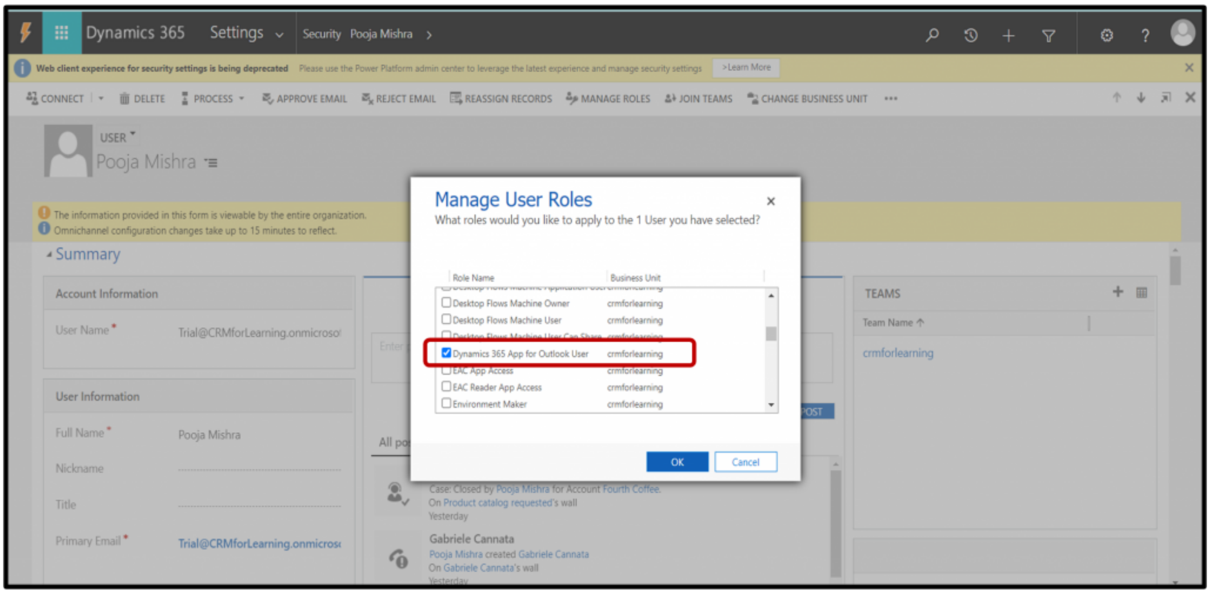
E-mailing is a preferred customer channel in Dynamics 365, and it is also an effective channel for lead nurturing and other sales activities. Switching back and forth between a CRM and a contact centre or email client is time consuming and can reduce productivity, but many CRM solutions integrate with email and bring additional benefits.

<u>Approach to achieve this requirement</u>:

- Security Role
- E-Mail Configuration
- Add Dynamic 365 App to Outlook
- Add an account with Outlook

1. <u>Security Role</u>:

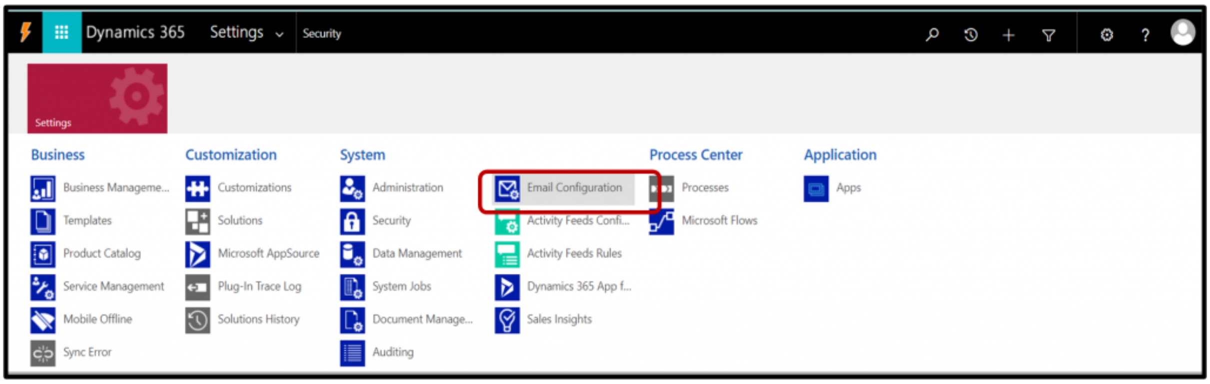
   NAVIGATION STEPS:
   Step 1: Navigate to Advanced Settings -> Security -> Users.
   Step 2: Select the User.
   Step 3: Navigate to Manage Roles.
   Step 4: Provide user a Security Role – Dynamics 365 App for Outlook User in your Dynamics.



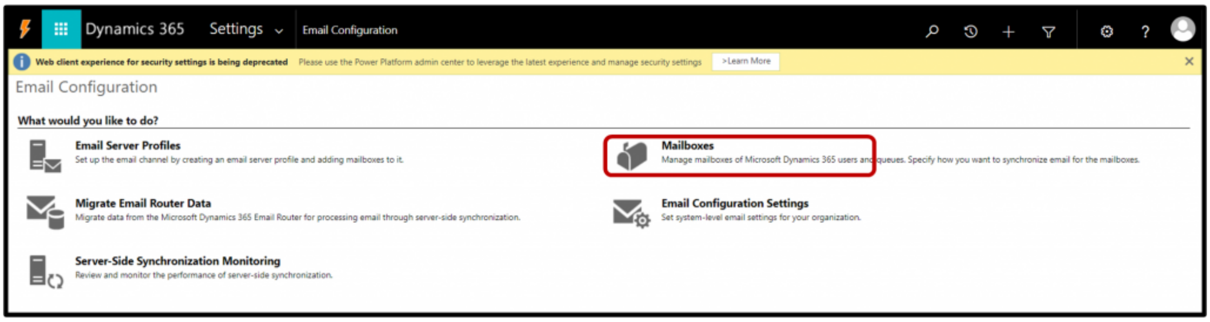
2. E-Mail Configuration:

NAVIGATION STEPS:

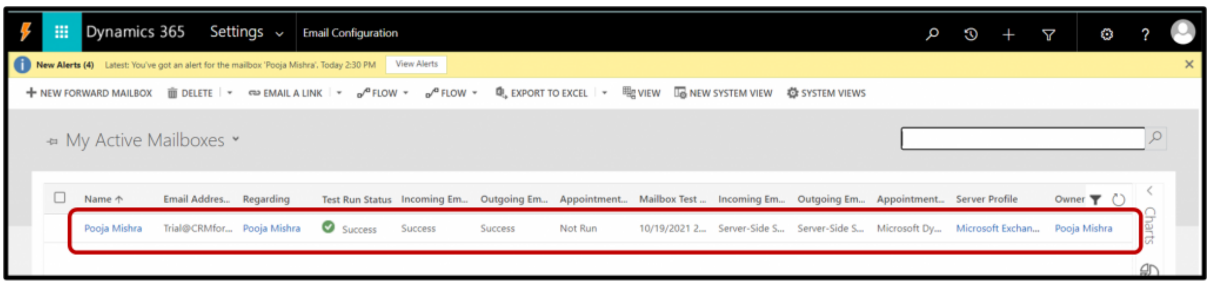
Step 1: Navigate to Advanced Settings -> Email Configuration.



Step 2: Navigate to Mailbox and open it.



Step 3: Select the required Mailbox and open it.



Step 4: Synchronization Method à Appointment, Contacts and Tasks.

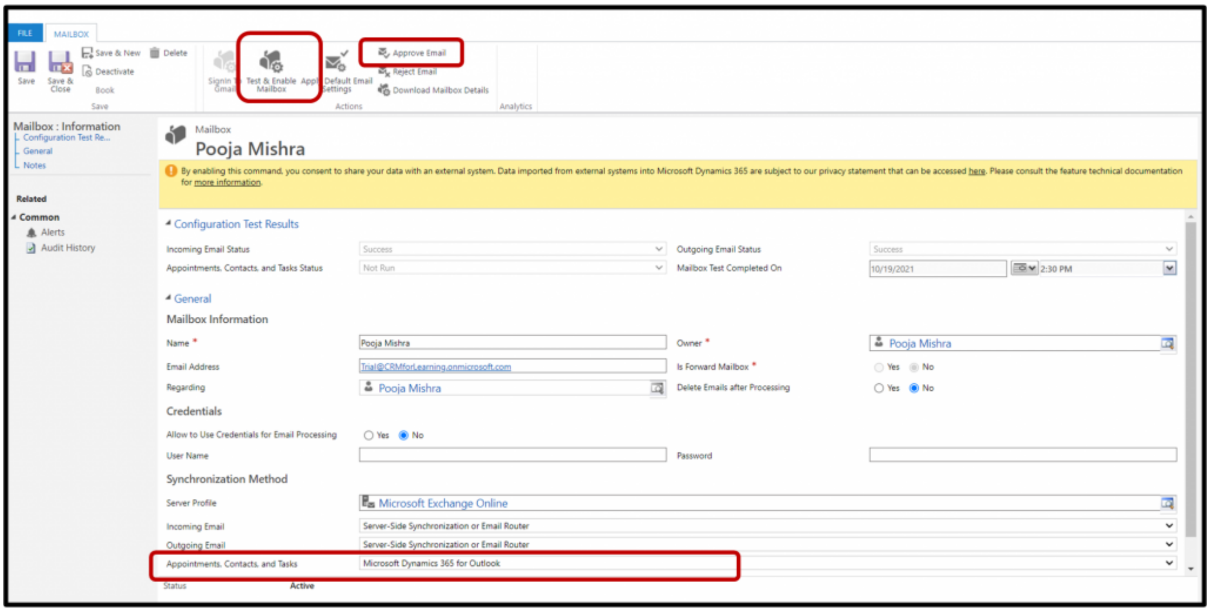
Step 5: Select MicrosoftDynamics 365 App for Outlook.

Step 6: Save the Form.

Step 7: Click on Approve Email option. A pop-up message Approve Primary Emailwill appear then click on OK.

Step 8: Click on Test & Enable Mailbox. A note will reflect, check the note, and click on OK.

Step 9: Save & Close the form.

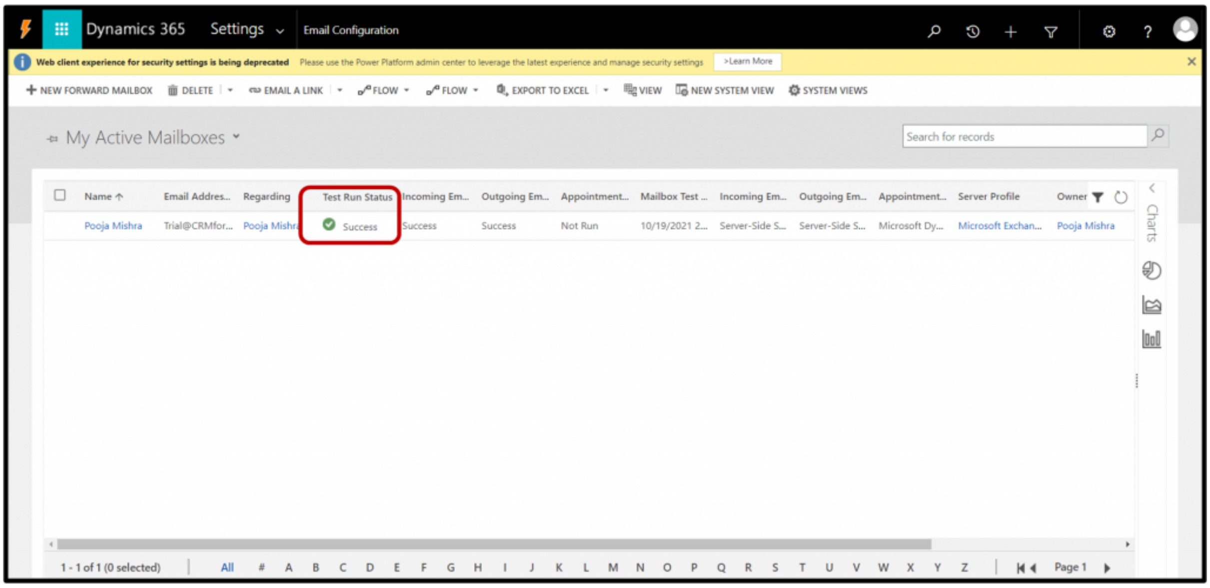


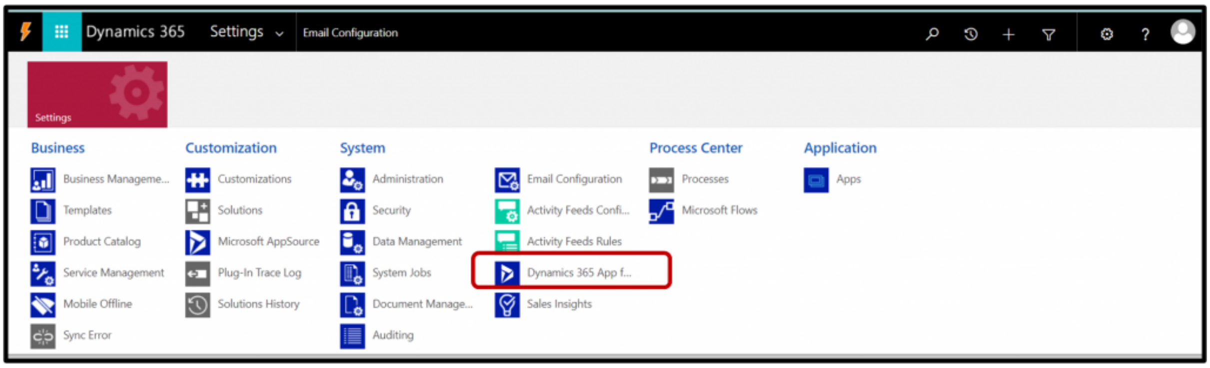
Step 10: Refresh the page and your Test Run Status will be successful.

3. Add Dynamic 365 App to Outlook:

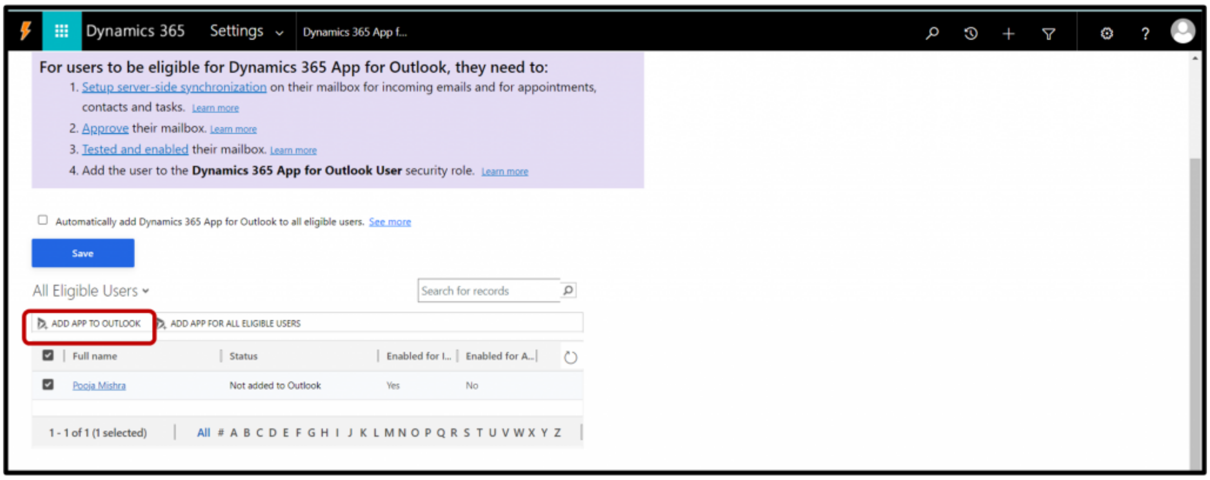
NAVIGATION STEPS:

Step 1: Navigate to Settings -> Dynamics 365 App for Outlook.





Step 2: Select the user and then click on ADD APP TO OUTLOOK.

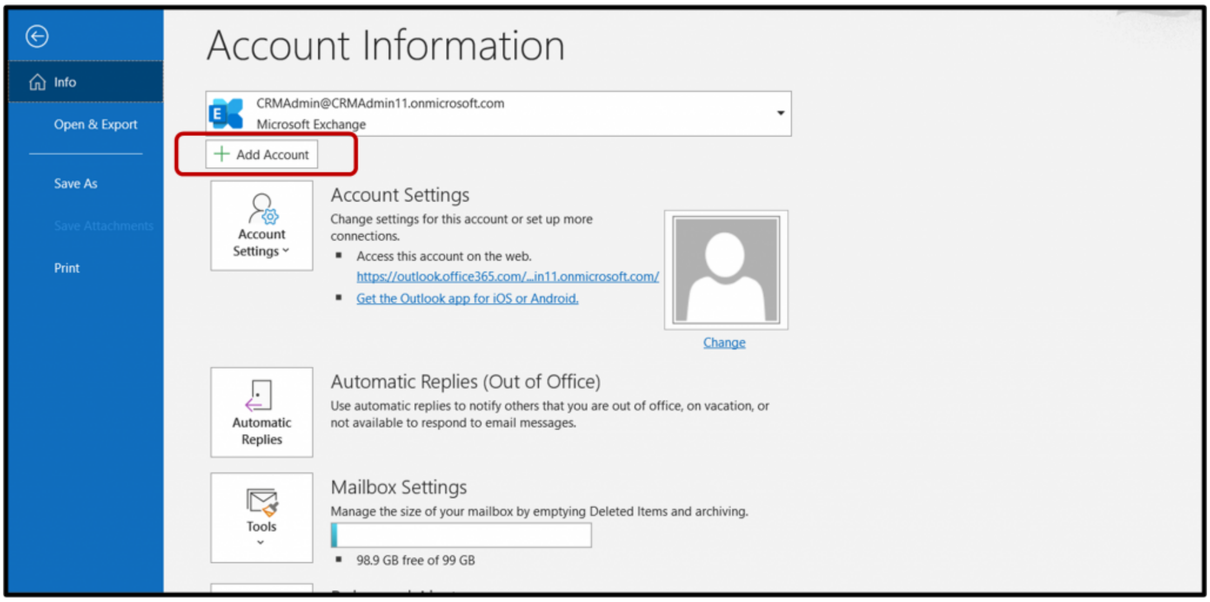


4. Add an account with Outlook:

NAVIGATION STEPS:

Step 1: Open your Outlook.

Step 2: Navigate to File à Add Account.



Step 3: Add your new Microsoft Dynamics 365 account by providing Username and password.

Step 4: Refresh Outlook and your connection is ready.

Microsoft flow OR Power Platform

Microsoft flow helps you to simplify your day to day **repetitive tasks** to flows, that can run or execute automatically without much manual intervention.
But Flow is not just meant for SharePoint, Microsoft flow allows you to **connect and communicate with other services** like Google, Gmail, Facebook, Twitter, and many more business applications.

By using flow, you can **send automatic notifications**, push notifications or emails, ask for **approvals** from designated persons, and many more things you can do.

<u>What is Power Automate?</u>

What is Power Automate? **Power Automate is nothing new, Microsoft just renamed Microsoft Flow to Power Automate.**

Why use Microsoft Flow?

Why use Microsoft Flow? **We can use flow to automate lots of business processes by adding various flow actions with stages and steps. Now only this we can also use flow to send just email notifications or push notifications.**

Apart from this, we can also use Microsoft flow to **start simple to complex approvals**, even we can start various multilevel approval processes.

Not only this we can also use other services to **capture information**. For example, we can use Microsoft flow to capture data from Twitter or Facebook on a specific hashtag.

Even we can **store email attachments** from your email to SharePoint document library by using flow. We can create a flow that will trigger whenever you will receive an email with an attachment, this will store the attachment in SharePoint.

Even we can also **copy files between one drive for business and SharePoint** or even we can sync new OneDrive files to Google drive files. And, we can also upload photos to OneDrive for Business on a button flow.

Office 365 uses Microsoft Teams. We can even use flow to communicate with the Microsoft team from other applications. Like, we can Microsoft flow to **post messages to Microsoft teams when a new task is created in Planner**. Or we can **alert the team when a specific documents are uploaded**. Or even we can **post message to Microsoft Teams when an email arrives in Office 365 Outlook**.

But any time you can customize the flow according to your business needs.

Few flow templates are:

- Send myself a reminder in 10 minutes
- Start approval when a new item is added
- Send a customized email when a new SharePoint list item is added
- Save Gmail attachments to your Google Drive
- Save tweets to a Google sheet
- Copy files from Google Drive to Onedrive on a daily basis

- Block out my Office 365 calendar for an hour
- Start approval for new documents and notify via Teams, etc.

Microsoft flow connectors

Microsoft provides flow connectors to connect with various services like Twitter, RSS, SQl Server, Google Calendar, Google Drive, Yammer, Slack, GitHub, Google Tasks, Asana, Twilio, etc.

There are nearly 250 connectors are available to work with various other services in Microsoft flow or Power Automate.

Standard flow connectors

These are the connectors that are available with your flow license to use. Examples of a few **standard flow connectors** are Twitter, Linkedin, Google Calendar, OneDrive for Business, Mail, Gmail, Dropbox, Trello, Instagram, Box, Buffer, Campfire, etc.

Microsoft flow Premium Connectors

Flow premium connectors require an additional license to use those connections. All premium connectors marked as Premium so that you can separate it from standard connectors. A few premium flow connectors are Azure DevOps, Dynamics 365, Salesforce, Agilite, airSlate, Ally, Azure VM, Chainpoint, Chatter, etc.

Custom flow connectors

You can also create your own custom flow connectors and can access various applications.

Type of Flows

We can create different types of flows for your business processes.

- **Automated flows**: We can easily create an automated flow that will trigger automatically. For example, you can create a flow that will trigger automatically whenever someone inserted an item to a SharePoint Online list.

- **Instant Flows**: Microsoft Instant flows will trigger on a button click. On a button click you can send a notification to your manager regarding work from home email.
- **Scheduled Flows**: Microsoft scheduled flows will trigger automatically on a specific scheduled time. For example, you can copy items from one SharePoint library to OneDrive for business on monthly basis.
- **Business process flows**: You can create a flow for your organization and ensure that everyone in the company follows the same steps. IT Support ticket systems.
- **UI flows**: In a UI flow, you can record clicks and keyboard strokes from your desktop and web applications and then you can automate the playback of manual steps on legacy software.

WEEK - 4 Content

Model Driven App Versus Canvas App

Model-Driven Apps is an AppModule enabling users to design and develop component-focused apps to implement an end-to-end business process. Canvas App is an AppModule enabling users to design and develop task-specific apps with designing flexibility.

What is model Drive app?

Model-driven apps **provides a no-code or low-code component approach to app development**. ... Model-driven apps start with your data model – building up from the shape of your core business data and processes in the Common Data Service to model forms, views, and other components.

Does model-driven apps require common data service?

The basis of a model-driven app is **always a data model**, which can be pre-existing in Dynamics 365 or the Common Data Service (CDS) or be built from scratch. Model-driven apps are based on CDS and executed in Dynamics 365. ... Users have less control over layout and functionality than with Canvas apps.

Business rules

You can create business rules and recommendations to apply logic and validations without writing code or creating plug-ins. Business rules provide a simple interface to implement and maintain fast-changing and commonly used rules.

Define *business rules* for a table that apply to all the table forms and at the server level. Business rules defined for a table apply to both *canvas apps* and *model-driven apps* if the table is used in the app.

Business rules provide a simple interface to implement and maintain fast-changing and commonly used rules. They can be applied to main and quick create forms. Business rules work in model-driven apps, legacy web apps, Dynamics 365 for tablets, and Dynamics 365 for Outlook (online or offline mode).

By combining conditions and actions, the following actions are possible with business rules:

- Set column values
- Clear column values
- Set column requirement levels
- Show or hide columns
- Enable or disable columns
- Validate data and show error messages
- Create business recommendations based on business intelligence.

Microsoft Dataverse

Dataverse lets you securely store and manage data that's used by business applications. Data within Dataverse is stored within a set of tables. A *table* is a set of rows (formerly referred to as records) and columns (formerly referred to as fields/attributes). Each column in the table is designed to store a certain type of data, for example, name, age, salary, and so on. Dataverse includes a base set of standard tables that cover typical scenarios, but you can also create custom tables specific to your organization and populate them with data by using Power Query. App makers can then use Power Apps to build rich applications that use this data.

Why use Dataverse?

Standard and custom tables within Dataverse provide a secure and cloud-based storage option for your data. Tables let you create a business-focused definition of your organization's data for use within apps. If you're not sure whether tables are your best option, consider these benefits:

- Easy to manage – Both the metadata and data are stored in the cloud. You don't need to worry about the details of how they're stored.
- Easy to secure – Data is securely stored so that users can see it only if you grant them access. Role-based security allows you to control access to tables for different users within your organization.
- Access your Dynamics 365 Data – Data from your Dynamics 365 applications is also stored within Dataverse, allowing you to quickly build apps that use your Dynamics 365 data and extend your apps with Power Apps.
- Rich metadata – Data types and relationships are used directly within Power Apps.
- Logic and validation – Define calculated columns, business rules, workflows, and business process flows to ensure data quality and drive business processes.
- Productivity tools – Tables are available within the add-ins for Microsoft Excel to increase productivity and ensure data accessibility.

(Github Link for details reference)

business logic using client scripting in model-driven apps using JavaScript

Apply business logic using client scripting in model-driven apps (Github Link for details reference)

Client-side scripting using JavaScript is one of the ways to apply custom business process logic for displaying data on a form in a model-driven app.

[!IMPORTANT] All the client scripting concepts and APIs explained in this documentation also apply to Dynamics 365 Customer Engagement (on-premises) users.

Client scripting shouldn't be your first choice though for applying custom business process logic in model-driven app forms. *Business rules* provide a way for someone, who does not know JavaScript and is not a developer, to apply business process logic in a form. More information: Create business rules to apply logic. You will find the business rule designer within the Data > Tables > [table_name] area on make.powerapps.com. When you view table, look for the Business rules tab.

However, if your business requirement can't be achieved using a business rule, you will find that client-scripting using the client API object model provides a powerful way to extend the behavior of the application and enable automation in the client.

Use client scripting in model-driven apps

Forms in model-driven apps help display data to the user. A form in model-driven apps can contain items such as columns, a quick form, or a grid. An event occurs in model-driven apps forms whenever:

- A form loads.
- Data is changed in a column or an item within the form.
- Data is saved in a form.

You can attach your JavaScript code to "react" to these events so that your code gets executed when the event occurs on the form. You attach your JavaScript code (scripts) to these events by using a Script web resource in model-driven apps.

Model-driven apps provides you a rich set of client APIs to interact with form objects and events to control what and when to display on a form.

[!NOTE] Some client APIs are deprecated in the current release of model-driven apps. Ensure that you are aware of these APIs as you write your client-side code for model-driven apps. More information: Deprecated client APIs

Events in forms and grids in model-driven apps

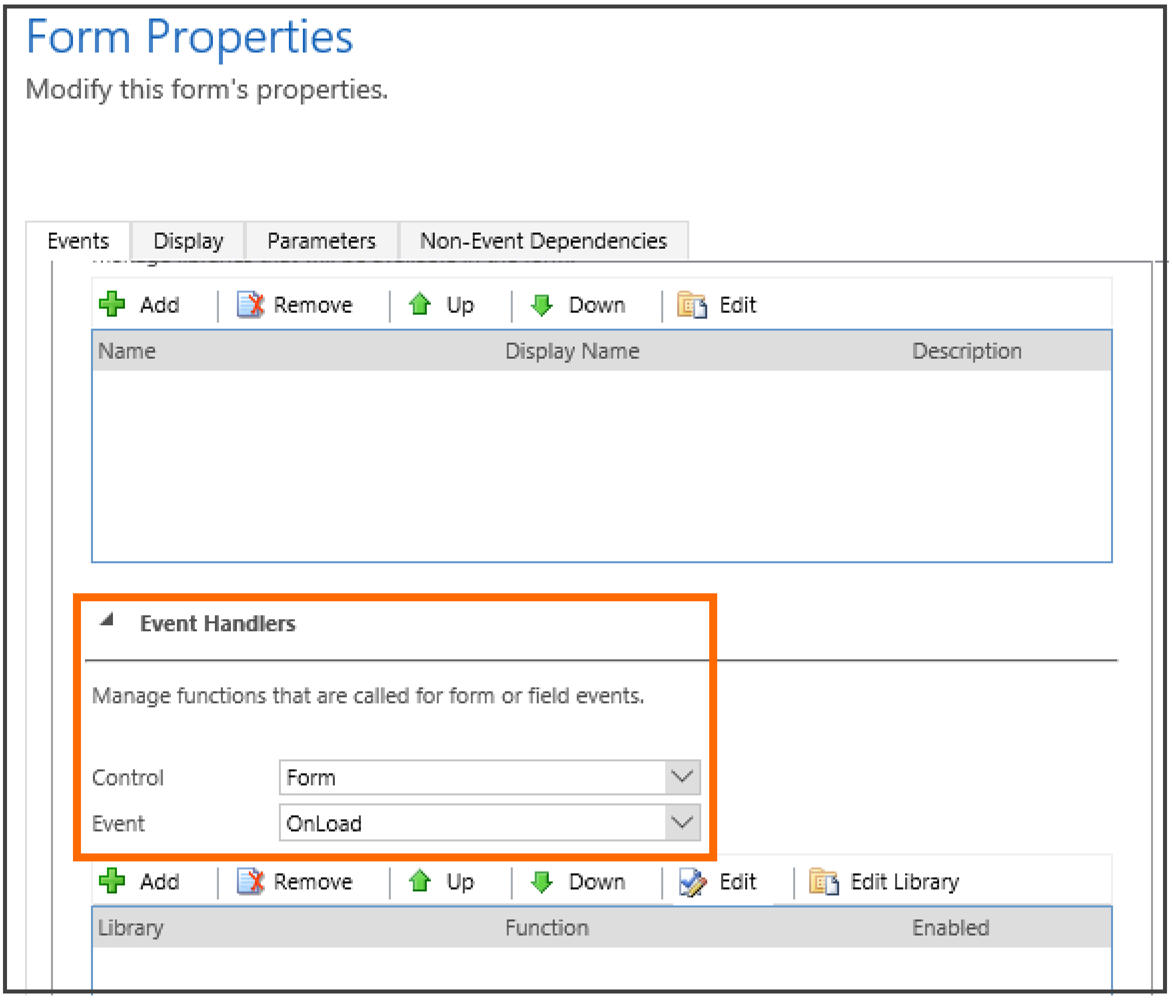
All client-side code is initiated by events. In model-driven apps, you associate a specific function in a JavaScript library (Script web resource) to be executed when an event occurs. This function is called an *event handler*. Each event handler specifies a single function within a JavaScript library and any parameters that can be passed to the function.

You can associate event handlers to only some events using the UI. For events that are not available to be associated through UI, Client API provides methods that can be used to attach event handlers to such events.

Add or remove event handler function to event using UI

Legacy

Use the Event Handlers section of the Form Properties dialog box to associate your script with an event for forms and columns.

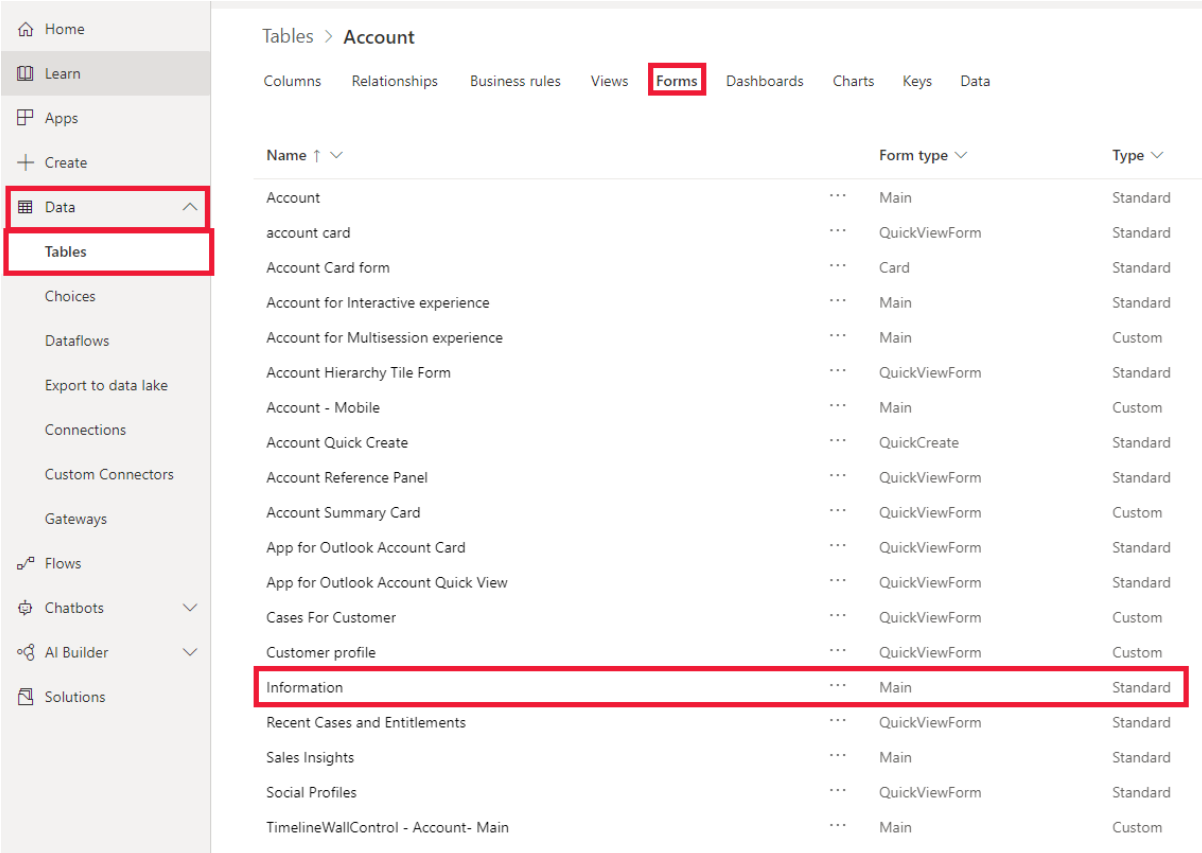


Unified Interface

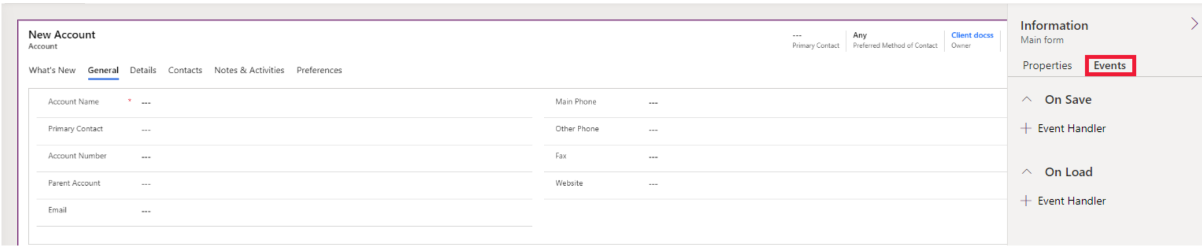
To add or remove event handler functions to events in the new Unified Interface:

1. Go to Power Apps.
2. In the left navigation pane, select Data and then select Tables.
3. From the list of tables, select the table where you want to add the event handlers.

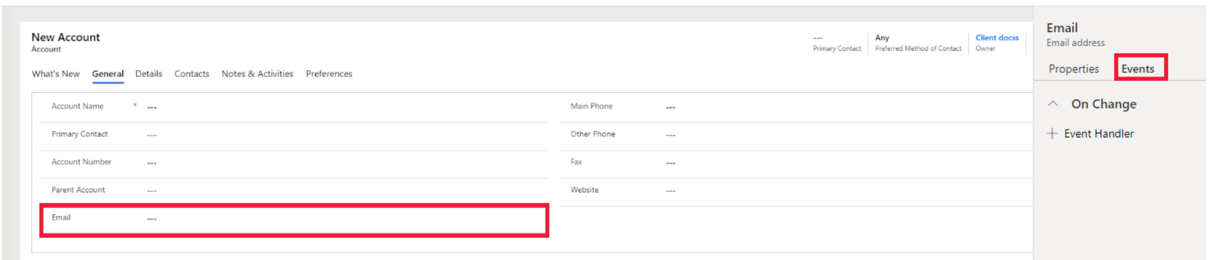
4. Select Forms tab from the command bar and then select the form where you want to add.



5. Select Events tab. You'll notice that both the On Save and On Load event handlers.



6. If you wish to add the event handler for the on change event, select the column and then select Event tab.



Add or remove event handler function to event using code

Using the following methods to add and remove event handler for events that cannot be associated through UI:

Events	Event handler
Attribute OnChange	addOnChange and removeOnChange methods
Form OnLoad	formContext.ui addOnLoad and removeOnLoad methods
Form data OnLoad	formContext.data addOnLoad and removeOnLoad methods
Form OnSave	addOnSave and removeOnSave methods
Lookup control PreSearch	addPreSearch and removePreSearch methods
kbsearch control OnResultOpened	addOnResultOpened and removeOnResultOpened methods
kbsearch control OnSelection	addOnSelection and removeOnSelection methods
kbsearch control PostSearch	addOnPostSearch and removeOnPostSearch methods

[!IMPORTANT] The execution context is automatically passed as the first parameter to functions that are set using the code. More information: Client API execution context

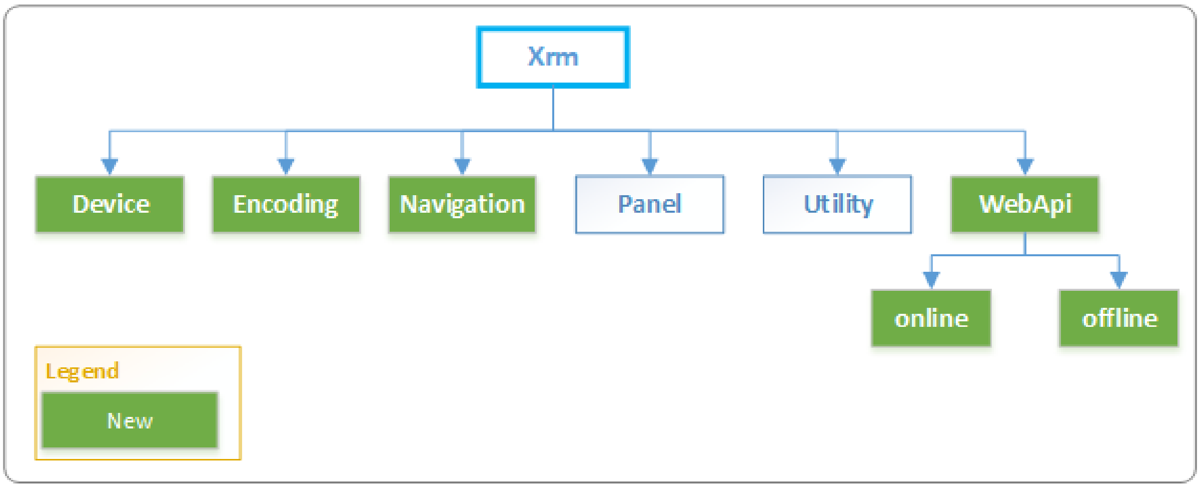
Client API Xrm object

-

The Xrm object is globally available to use in your code without having to use the execution context in Client API.

Xrm object model

The following illustration displays the Xrm object model:



Here is the information about each of the namespaces in the Xrm object:

Namespace	Description
Xrm.Device	Provides methods to use native device capabilities.
Xrm.Encoding	Provides methods to encode strings.
Xrm.Navigation	Provides methods for navigating forms and items in model-driven apps.

Xrm.Panel	Provides a method to display a web page in the side pane of model-driven apps form.
Xrm.Utility	Provides a container for useful methods.
Xrm.WebApi	Provides methods to use Web API to create and manage records and execute Web API actions and functions. Xrm.WebApi.offline: Provides methods to create and manage records in the model-driven apps mobile clients while working in the *offline* mode. Xrm.WebApi.online: Provides methods to use Web API to create and manage records and execute Web API actions and functions when connected to the server.

Client API global context

Use the Xrm.Utility.getGlobalContext method in forms to retrieve information specific to an organization, a user, or the client where script is run without going through the form execution context. This is a change from previous versions where you had to use the form context to retrieve global context by using Xrm.Page.context.

Link for Reference

https://docs.microsoft.com/en-us/powerapps/developer/model-driven-apps/clientapi/walkthrough-write-your-first-client-script?tabs=unified-interface%2Cunified-add-web-resource