servicenow

# No-Code (Citizen Developer) Guide for the Now Platform

servicenow

# Table of Contents

# Introduction

A key initiative in many organizations is to reduce IT workload by empowering "citizen developers" with the tools and processes needed to build their own applications on the Now Platform. These citizen developers can have limited to no coding experience, which means the tools provided to them must be easy to use but also provide enough functional depth to accommodate their business needs.

ServiceNow provides these citizen developers, along with more experienced developers, with the Now Platform. The Now Platform provides a single mobile and web application development platform to quickly build business applications that power your digital transformation.

Within this guide, you will learn how anyone can automate, extend, and build digital workflow applications (i.e., apps) across the organization using the Now Platform.

## Steps for building an application

Building an app on the Now Platform can be done by following this four-step process:


- Planning: think through your application before you start building it
- Data: the data you want to gather, display, process and report
- Design: how you want others to interact with your application
- Logic: the way your application will work

The remainder of this guide will provide guidance for each of these steps so you can easily build a ServiceNow application.

## Who can build an application in ServiceNow?

If you want to build an application in ServiceNow, you don't need to be a professional developer, know how to write code nor even need to have built an app before. All you need to understand is the business process you want to automate and have a willingness to learn something new.

If you are an expert in a certain business process, but are not a professional programmer, then this guide is for you.

# Planning

Paramount in building any app is the planning process. Being *intentional* about the planning step has both immediate and long-term benefits for the app you want to build.

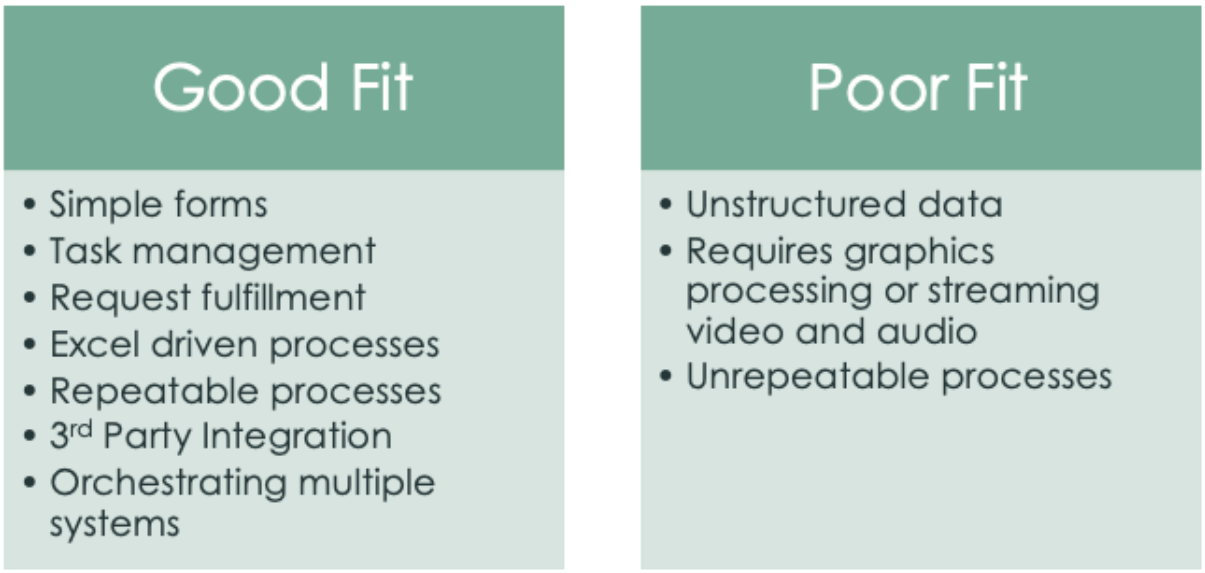
Your answers to the following questions will help to determine how best to utilize Now Platform features to build an app that maximizes business value to your organization:

1. What are the goals, objectives, and outputs of your application? (i.e., what business problem are you trying to solve?)
    a. Without a specific business objective, you will have difficulty measuring the success of your application or justifying its continued use within the organization
2. Are you taking a spreadsheet and turning it into an application in ServiceNow, or does the application exist somewhere else?
    a. This decision impacts your approach for building the application as there are different tools within the Now Platform to support your efforts
3. What people will be using your application?
    a. Identifying your target audience has a direct impact on the features your application will provide, the data it will capture and the interface you need to provide for your application
4. Do you want everyone to have the same ability to see and edit fields, or will some people need more or less access than others?
    a. Security is an increasing concern in most organizations, so identifying who has access to what during the planning stage is a critical step in application development
5. What do the users do with the application? (i.e., provide information, collect information, route information, request something, look up information, collaborate on information?)
    a. Identifying these actions establishing the features and functions you will need to build into your application
6. Will all of the data in your application be entered by people, does it already exist in ServiceNow (like user data), or will you need to import data from an external source?
    a. You will want to leverage available data sources as much as possible to avoid duplication of data, as well as ensure your application has the data it needs to meet its business objectives
7. How will people interact with your application? On their computer or mobile device?
    a. Understanding how people access your application impacts how your application will function (swipe of a finger or click of a mouse or both)
8. How will your stakeholders need to report in your application?
    a. If your application is meeting a business purpose, you will want to be able to provide reports showing usage, adoption and key business objectives associated with our application

Throughout the remainder of this document, we will revisit these questions and explain why they are important to answer during the planning stage of application development.

## Candidates for a good ServiceNow app

Not every application idea makes for a good fit on the Now Platform. As the criteria below shows, some apps are better suited for the Now Platform than others:




### Planning Question: What are the goals, objectives, and outputs of your application?

Before you start building, you should begin with the end in mind. Understanding and visualizing (virtually or on a whiteboard) your desired solution helps determine the remaining steps in building your app. Often, the outputs are the drivers for the inputs. If you are attempting to speed up a process, then knowing your output metrics can help make clear what to measure. If you are managing assets, then perhaps cost and location are more important than the minute details of each item. Identifying your goals and objectives will ensure you can manage conversations with key stakeholders to ensure your application is specifically addressing the desired business outcomes.

## Permanent Considerations

There are some actions that you will take when building an app that are irreversible, so it's important to be aware of these so you can plan in advance.

- Scoped vs Global Application
  - When you create an application, you can choose to create it in a private scope (scoped application) or in the global scope (global application). Scoped applications get extra functionality for managing development, deploying the application, and data security. By default, all applications are created in a private scope and ServiceNow's recommendation is that citizen developers work with scoped applications.
- The instance where you start building your application matters
  - Proof of Concept (PoC) app builds can be built in a Personal Developer Instance that you get from the developer portal [developer.servicenow.com].

        - These instances will be named something like dev12345.service-now.com.
        - You can rebuild these PoC apps in your personal instance, but do not import them into that instance
    - Applications that your organization will actually use (i.e., production apps) should be created in your organization's dev instance so they can follow your organization's testing and deployment process. See your ServiceNow System Administrator for more details about which instance to use for an app that will eventually be deployed to your organization's production instance.
- The name you give your application matters
    - Based on your application's display name, ServiceNow suggests an internal name called the application scope. It will be in the form of x_[company code]_[app_name] like **x_acme_legal_request**.
    - Everything you create within your app will inherit that scope name so it's important to think about what that will be when you start. The display name of the application can always be changed.
- The names you give your tables and fields matter
    - Once your app is created you will most likely be creating new tables and fields for your app. Tables and fields have internal database names which should only be edited at creation time.

## Prerequisites for building an app

Before developing your app, you will need:

- A ServiceNow instance
- An admin or delegated developer role in that ServiceNow instance. A delegated developer role is a role with fewer privileges than the admin role but still allows for application development.

You should also get familiar with the ServiceNow interface through this **SELF-PACED TRAINING:** Build My First Application

# Data

Now that you've done your planning and prerequisites it's time to build your data model. This means that you'll be creating one or more tables with fields on them, possibly loading it/them with data, and making sure the appropriate people have the correct level of access to that data.

## Create an app

The first action to take is to either create or open your application record:

- **Create** - using ServiceNow Studio, you can create an app a number of ways, but it is recommended that you use the Create a custom application option, which allows you to create a table, user role, and application menu as the scoped application is created.
- **Open** – depending on how your organization manages access to the app development process, you may need your organization's ServiceNow System Administrator to create the application for you and grant you a delegated development role

**Expert tip**
The internal names of any application and table you create cannot be changed once they are created. However, you can change the display name of the application and tables at any time. The users of your application will only see the internal names in the URL, but you should make sure they are concise and accurate.

## Create the Tables and Fields

Once the application is created, you'll need to add fields to the table created when you created the application as well as create any additional tables needed.

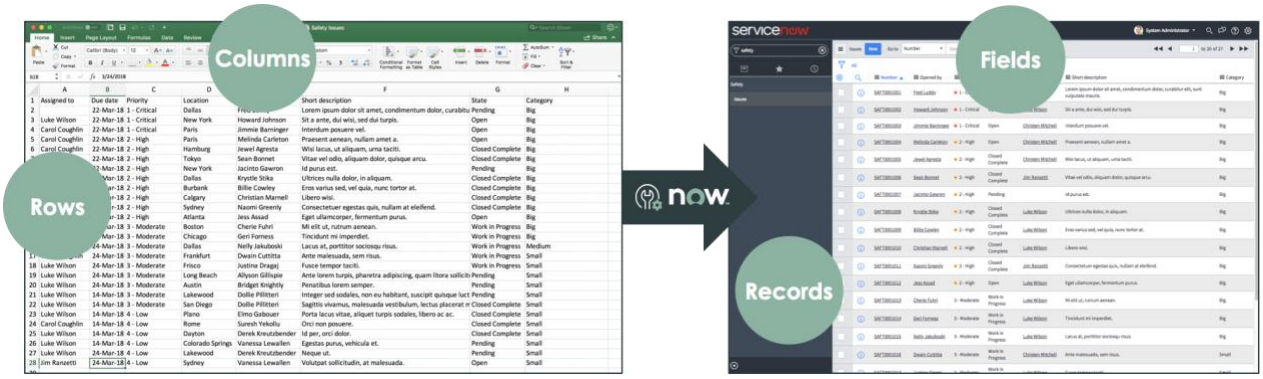
NOTE: There are 6 fields auto-created for every table in ServiceNow. They contain auto-populated information about the table like when it was created, when it was last updated and by whom, as well as a unique identifier for the table. These fields cannot be manipulated.

| Field name | Database Name | Description |
|---|---|---|
| Created by | sys_created_by | The user that created the record |
| Created | sys_created_on | The date/time which the record was created |
| Updated by | sys_created_by | The user who last updated the record |
| Updated | sys_created_on | The time at which the last record was updated |
| Sys ID | sys_id | The unique identifier for the record. This is auto-assigned and unique throughout the instance |

| | | |
|---|---|---|
| Updates | sys_mod_count | A numeric field that counts the number of updates for this record since record creation |

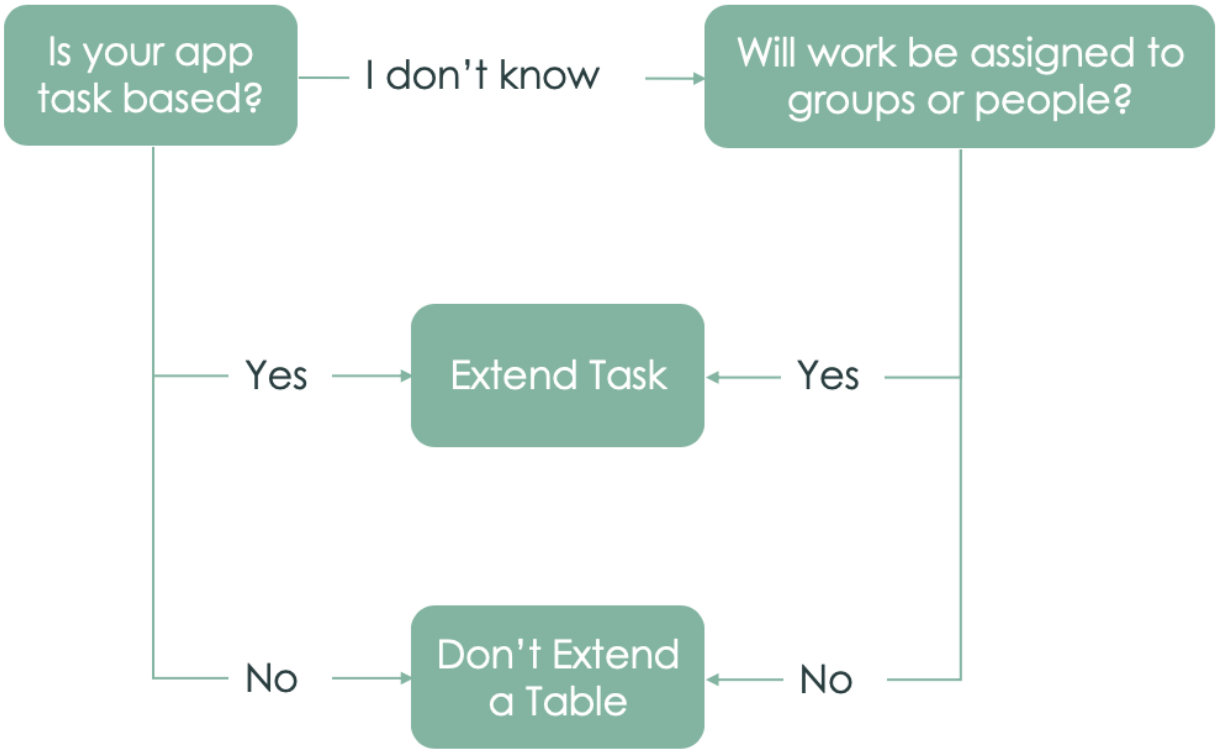
### Planning Question: Are you turning a spreadsheet into an application?

If you're creating an application based on a spreadsheet, each worksheet is likely to map to a table in ServiceNow, each column may become a field on that table, and each row a data record in the table.




When creating a table, you have the option to **extend** a table, which means that your new table inherits all of the fields and functionality from the table you're extending, saving you time. By far the most common table to extend in ServiceNow is the **task** table.

To determine if you want to extend a table, use this decision tree:




NOTE: contact your ServiceNow System Administrator if you think there is a table other than Task that you want to extend within your app.

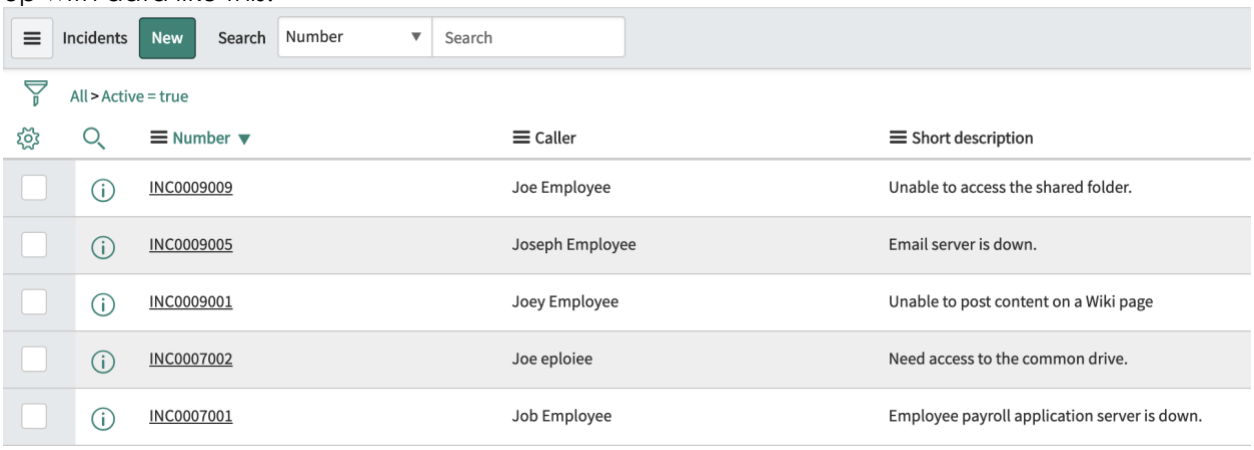
**Documentation**: Create a Table

## Fields

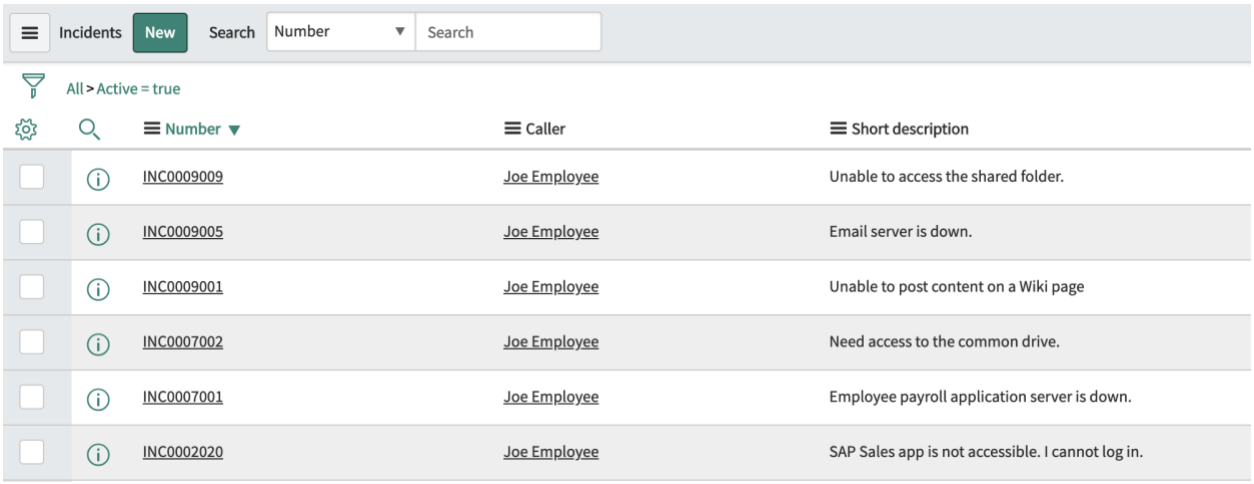
Once you have created a table, you will need to add fields. ServiceNow has many different field types with built-in validation and you should choose the one that best fits that field's data type. It's easy to make plain text (string) fields where people can enter anything, but it can result in bad and inconsistent data that is difficult to use.

**EXAMPLE**:

You have a field on your table for someone's name. You use a plain text (string) field and end up with data like this:




However, if you use a reference field type instead of a plain text field, you will get data that looks like this:




You can use reference fields to normalize (i.e., make consistent) your data by referencing an existing table in ServiceNow. As ServiceNow has 2000+ tables at your disposal, Appendix B lists some commonly used tables for building an app.

While a reference field can normalize your data, there are other fields that can be used for specific types of data. The complete list of Field Types can be found in the ServiceNow Documentation site, but here are some common field types:

| Field Type | Notes |
|---|---|
| Integer | Use this if this value will always be a number and you may be using it in calculations |

| | |
|---|---|
| Currency | Holds a currency value and will show the currency of the logged in user |
| Phone number | Includes validation and formatting for E164-compliant phone numbers |
| Reference | This displays a record from another table and helps to normalize your data |
| Choice | Choice lists are another way to normalize data through a preset list of choices displayed as a select box. This should be a shorter list of less than 10 choices. |
| Date | A date picker. Use this if you don't need a specific time. |
| Date/Time | A date and time picker. Use this if you're comparing specific times or the exact time is important. |
| String | This is a freeform text field and should be used if no other field type fits your purposes |

NOTE: once you have created a field it is difficult to switch to a different type.

**Expert tip**
If you have extended a table, you will have a number of fields to choose from rather than having to create new fields. Before creating a new field, check to see if there is an existing field that might meet your purposes simply by changing the field's label. Note that the purpose of the field should be similar to the purpose of the field in the base table.

## Secure Your Data

During the table creation process, security rules are created for the table where only people with the application's role can access the table to read, create, write, and delete. Data security is one of the most important and overlooked aspects of creating an app.

### Planning Question: What kind of people will be using the application?

You need to identify If you have people with different jobs using your app. For example, if you will have some people making requests and other fulfilling those requests, you'll need a couple of personas within your app as you don't want the requestors to be able to see or edit the same fields that the fulfillers can.

To accommodate these different personas, you should create a new role for each of your personas, and then configure the security on the individual fields for those roles. You can also use Access Controls to apply security to all fields on a table and grant access to individual fields as exceptions. If you think you may have this type of security model it is recommended that you go through the self-paced security training linked at the end of this section and contact your ServiceNow System Administrator.

### Planning Question: Is there any sensitive data in your applications and do you want everyone to have the same ability to see and edit your data?

If your app contains sensitive information like people's names or financial information, you should determine which personas need to access this information. It is important to examine the data that will exist in your app and make sure that you don't have a persona able to access what they shouldn't.

As shown in the table below, some data can be access by any type of user, while access to more sensitive data should be controlled.

| Data point | App User | App Fulfiller | App Admin |
|---|---|---|---|
| Task data | Read/Write | Read/Write | Read/Write/Delete |
| User records | Read | Read/Write | Read/Write |
| User Personally Identifiable Information (PII) fields | No access | Read | Read/Write |

In addition to securing data in tables, records, and fields by role, you can also secure it contextually (based on the records other values) and with a script. Contact your ServiceNow System Administrator for help if you have a more complex security requirement.

**Expert tip**
It is important that the security is set up in this step before you configure any interfaces or business logic. Since security can affect the data available to interfaces and business logic, waiting until the end of your application build process may cause rework and issues. Also, consider making any auto-populated fields read only.

**SELF-PACED TRAINING**: Securing Applications

## Import Your Data

Now that you have your data model (tables and fields) created and security set up, you may need to add data into your application's table(s).

**Planning Question: Will all of the data in your application be entered by people, already exist in ServiceNow (like user data) or will you need to import data from an external source?**

If data needs to be imported, the easiest way to import data into your application's tables is to use the Easy Import functionality. This functionality allows you to create a spreadsheet that you can populate and easily import into ServiceNow. Before importing data make sure that the data does not already exist in ServiceNow. If data already exists in ServiceNow that fits your purpose you can reference that data from your table with a reference field.

**Expert tip**
Break up large amounts of data into smaller sets for faster imports. Consider 100,000 rows your baseline and break up anything larger than that into sets of 100,000. For example, importing 10 sets of 100,000 will finish quicker than 1 set of 1 million records.

SELF-PACED TRAINING: Importing Data

## Recap

For this part of building your app, you have learned about Data, creating your app, creating tables & fields, securing your data and importing your data. As shown in the table below, all of these activities can be accomplished using the Studio feature within the Now Platform.

| Function | Data | Design | Logic | Tooling |
|---|---|---|---|---|
| Application Creation | X | | | Studio |
| Tables and Fields | X | | | Studio |
| Secure Your Data | X | | | Studio |
| Import Your Data | X | | | Studio |

# Design

Now that your application's data model is created, secured, and where necessary, populated with data, you now need to create the design elements to access that data.

### Planning Question: How will people interact with your application?

Will your app be accessed via desktop or mobile? Is your target audience already using an application on ServiceNow where they're comfortable using forms and lists, or will they need a self-service type of interface? These are the design considerations when building an app for specific audiences.

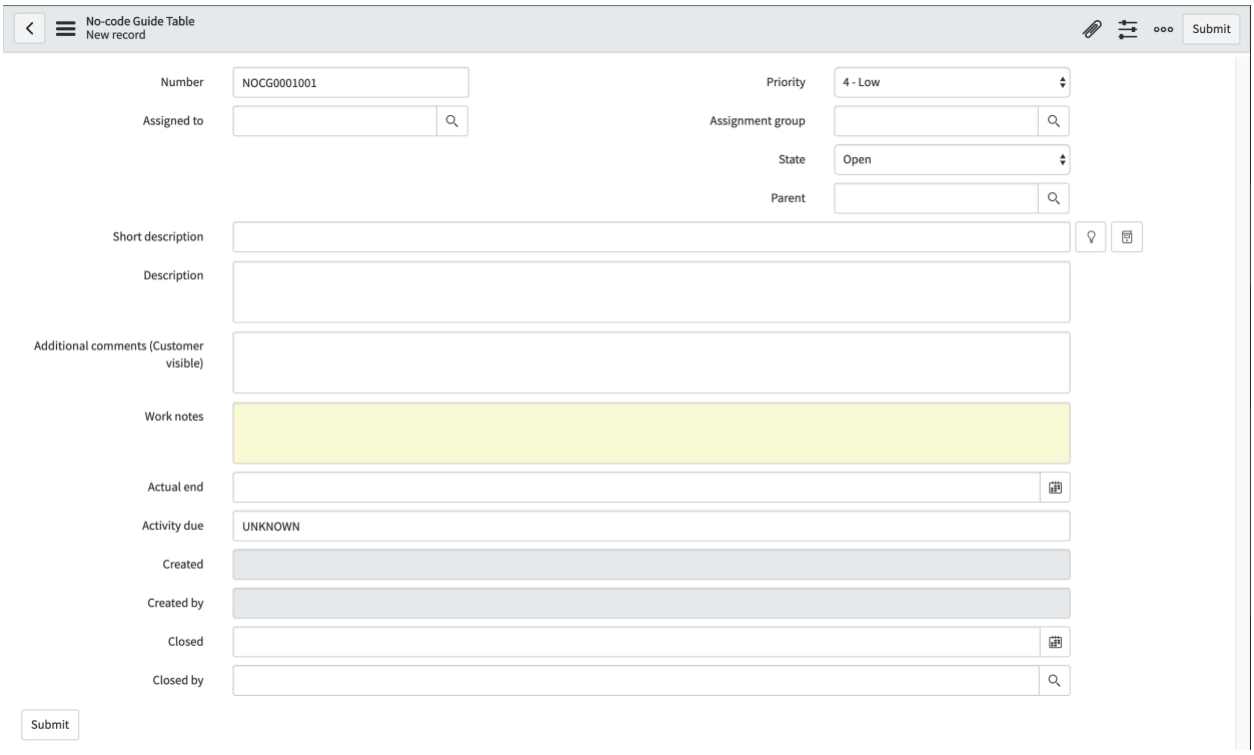
## Forms and Lists

The standard method of accessing data in ServiceNow is through the default forms and lists. A form displays information from one record in a data table, and a list displays a set of records from a table. When configuring forms and lists there are a few guidelines you should follow.

1. Keep the number of fields on a form to a minimum. The more fields you have on a form the longer it will take to load. You can use form views to create different sets of fields for different situations.
2. Use form sections to logically group fields together and keep users from having to scroll. The top section of the form should contain the fields that are always needed or used, while the other form sections contain less frequently utilized fields.
3. Make sure fields appear in the right order. For example, the start date field should always come right before an end date field.
4. Use 7 or fewer columns in a default list. People can add more if they want to by personalizing their lists.
5. Avoid using a reference field as the first item in the list view as it is shown as hyperlinked text. Clicking on the reference field will redirect the user to the referenced record instead of the list record and results in a poor user experience.

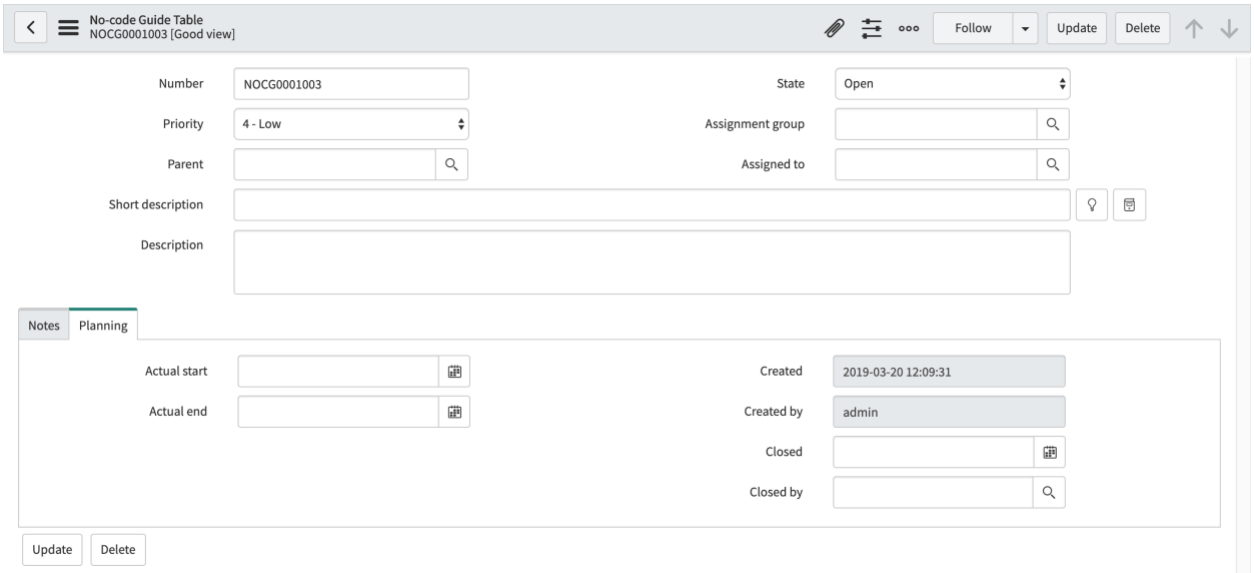
The figure below shows an example of a poorly designed form as it contains the following characteristics:

- There are no sections – it is one long form
- Similar fields are not grouped together (example: "assignment group" and "assigned to" are on different sides of the form




Below is a well-designed form as it contains the following characteristics:

- Fields are grouped together logically, like "assignment group" and "assigned to"
- The form has been broken into sections for easier viewing and data entry



## Mobile

If users will interact with the app on their mobile devices and will need native iOS or Android functionality like geolocation or offline access to your data, you can use the ServiceNow Agent app for this. The development is done by creating a mobile application in Studio inside of the application you've been building.

Once your mobile application is created you will create applets. An applet shows up as a tile within an application and consist of a master screen, such as a map or a list, and various details screens, such as an activity stream or related lists. Applets should have focused experiences.

**Expert tip**
Individual applets can be secured by role as well as made available in offline mode.

SELF-PACED TRAINING: Mobile

## Service Portal

If during the planning phase, you decided that your application has a Requestor or Self-Service user, you may want to use Service Portal to provide a friendly web experience.

In order to give self-service users the ability to easily create records from your company's existing Service Portal, create a record producer from the table record. A record producer provides a better end-user experience instead of using the regular form. Talk to your ServiceNow Administrator about the appropriate catalog and categorization so the record producer is accessible through your Service Portal.

Alternatively, you may need to create a new Service Portal for your app if the following are true:

- You need to provide different branding, navigation, or user experience than your organization's current Service Portal.

**OR**

- Your organization does not have an existing Service Portal,

**AND**

- You need to provide more functionality than the portals included by default provide

**AND**

- You need a more customized user experience than the default forms and lists can provide,

**OR**

- You need more control over branding and themes than the default interface provides

**Expert tip**
Do not try to reuse any existing service portal pages in your app. Create a new portal and new pages, and then reuse components in your pages like widgets and headers.

SELF-PACED TRAINING: Service Portal

## Reporting & Dashboards

Most applications will have some level of reporting requirements. Reports should be created with actions in mind and be built to drive change. The reporting tool in ServiceNow is very powerful, so here are some guidelines to follow:

- Be careful when reporting on large tables as it could have performance impact on your ServiceNow instance. Make sure you're filtering by a date range or other limiting criteria rather than showing all records on the table.
- When grouping records in a report, try to avoid grouping by fields that contain many possible values as it could impact performance.
- If running your report gives you a "Long running transaction timer" message and takes a long time to run, you should consider adding more data filters to reduce the report run time.
- If someone needs a report daily or weekly, consider scheduling it to be sent via email.

You can also use dashboards to show multiple reports on one page. Be careful with the number of reports you add to a dashboard. If you have too many reports on a dashboard and multiple users are using that dashboard it could affect overall instance performance.

SELF-PACED TRAINING: Reporting and Analytics

## Recap

For this part of building your app, you have learned about Design, including forms and lists, mobile, portal, virtual agent, reporting & dashboards. As shown in the table below, all of these activities can be accomplished using the Studio, Service Portal Designer and Virtual Agent features within the Now Platform.

| Function | Data | Design | Logic | Tooling |
|---|---|---|---|---|
| Application Creation | X | | | Studio |
| Tables and Fields | X | | | Studio |
| Secure Your Data | X | | | Studio |
| Import Your Data | X | | | Studio |
| Forms and Lists | | X | | Studio |
| Mobile | | X | | Studio |
| Portal | | X | | Service Portal Designer |
| Reporting & Dashboards | | X | | Studio |

# Logic

Now that you have created your application's data model and provided your users a way to access the data, it's time to add some logic. Logic is what makes your app a useful tool and can come in many forms, ranging from form logic (what people can and cannot see/use on a form), business logic (rules that govern what happens to data when it is entered), and notifications (making users aware of conditions and events within the app).

## Form Logic

Controlling what users see when they visit a form can greatly increase productivity and responsiveness. For example, users should only see fields that are useful to them and may need to see different fields based on what they have selected so far. Several options exist for controlling what is visible, read-only, and mandatory on a form as well as showing conditional messaging.

The following question will help direct you to the right decision for when to control user access to information: Is this a suggestion or enforcement? A suggestion makes the form easier to complete whereas enforcement forces the user to do something in order to complete the form.

UI policies are useful for conditional *suggestions* like showing and hiding fields or adding field messages based on another field's value, while data policies and business rules are better suited for doing conditional *enforcement* like making a field mandatory.

The best user experience is to utilize both suggestion and enforcement together.

SELF-PACED TRAINING: UI Policy Section

## Business Rules

Business rules are logic and validation rules that run when a record is created or updated. They are good for building simple conditional logic to run after the form is submitted.

Trigger: **if** this happens on a record,
Action: **then** set this value or show this message.

More complex logic with multiple steps can be done via Flow Designer.

DOCUMENTATION: Business Rules

## Flow Designer

Flows are a powerful tool for building business workflows. When designing a flow, keep the following tips in mind:

- Single Purpose – each flow should have a singular goal
- Reusability – Design for reusable sub-flows (approval is a great example)
- Clarity – The language and layout of your flow should be made so that it is very clear each action's purpose.

Start with a white board design of your business flow. Then build the flow action by action to align with your process. You might need more than one flow, for a single process to keep to the principles above.

SELF-PACED TRAINING: Flow Designer

### IntegrationHub

One powerful tool available when building flows is utilizing pre-built integration actions from IntegrationHub. Select from available IntegrationHub spokes while building a flow to easily allow ServiceNow to interact with other systems in your application landscape. For example, if your team uses Slack for collaboration, you can have your flow send a notification automatically to a Slack channel. For the available spokes in in your organization, contact your ServiceNow System Administrator.

## Notifications

Most applications will need some sort of email notifications configured. Some examples of that are:

- When a task is assigned to a user or group.
- When a request is opened or closed on behalf of someone.
- When an approval is needed from someone.

There are many examples already in the notification table you can use if you need to create your own notifications and it's very easy to copy those and change the copies to suit your purposes.

Here are some things to keep in mind when setting up notifications:

- Consider keeping the Send to event creator checkbox unchecked. In this case ServiceNow will not send an email to the person who took the action causing the email to be sent. For example, if I assigned a task to myself, I don't need to be notified about it.
- Use email templates if you think you will be sending out multiple notifications containing the same subject and/or body. For example, if you're sending an email when a task is assigned to a group, it will probably contain the same text as an email getting sent to the assigned user, even though the conditions and recipients will be different for both notifications.
- Rather than specifying a specific user or group (also known as 'hardcoding') in a notification, consider using the *Users/groups in fields* field on the notification record to automatically send to the person or groups in a user or group reference field on your record. This is also one more reason to use a reference field over a string field in your table.

SELF-PACED TRAINING: Notifications

## Recap

You've now made it through the Data, Design, and Logic sections and completed your application build.

| Function | Data | Design | Logic | Tooling |
|---|---|---|---|---|
| Application Creation | X | | | Studio |
| Tables and Fields | X | | | Studio |
| Secure Your Data | X | | | Studio |
| Import Your Data | X | | | Studio |
| Forms and Lists | | X | | Studio |
| Mobile | | X | | Studio |
| Portal | | X | | Service Portal Designer |
| Dashboards and Reporting | | X | | Studio |
| Form Logic | | | X | Studio |
| Business Rules | | | X | Studio |
| Flow Designer | | | X | Flow Designer |
| Notifications | | | X | Studio |

# App Completion

So, you've finished building your app. Now what should you do?

You should have been building your app in your organization's development instance. Before you make your app available to others, you will need to move it to a testing instance (usually called *test* or *QA*) so other people in your organization can test it.

Once your app passes testing, it will move into your organization's production instance where users will be able to interact with your application. Congratulations, your app is now solving the business problem it was designed to address.

Check with your ServiceNow System Administrator for help with your organization's testing and deployment process.

Once your application has been moved to production, you will want to start utilizing versioning. This is done using the publish function inside of Studio. Versioning allows you to publish changes to your app or roll back those changes if they have adverse effects.

## What to do next

Now that your application is being used in a production environment, it's time to think about how to improve and enhance it. Here are some suggestions for determining where to go next:

- The people who are using your application day-to-day will be the best source of feedback. Talk to them about what new features or changes they would like to see.
- Is there another interface you could support, like Service Portal, Mobile, Virtual Agent, or Visual Task Boards that you did not include in the first version of your app?
- Are there additional process flows you can support through Flow Designer?

## Feedback

If you would like to leave any feedback regarding this document, please email platformenablement@service-now.com and let us know what you think.

# APPENDIX A – ServiceNow Tools and Resources

## Studio

ServiceNow Studio is a tool that offers ServiceNow application builders a centralized interface where they can create and manage their applications. Studio will be the primary interface for creating and updating parts of the application.

- How to access studio
- Practice with Studio

## Flow Designer

Flow Designer is a Now Platform® feature that enables rich process automation capabilities in a consolidated design environment. It enables process owners to use natural language to automate approvals, tasks, notifications, and record operations without having to code.

- Getting Started with Flows
- Practice with Flow Designer

## Service Portal

- Service Portal Configuration
- Practice with Service Portal

## Virtual Agent

If you have a Service Portal for end users of your application, consider adding a conversational bot that can chat with your users and take actions based on these chats. You can specify inputs and take action on those inputs like creating or looking up records. Find out more about Virtual Agent in the ServiceNow docs site.

- Getting Started with Virtual Agent

## Resources

Here are some resources you should bookmark for help.

- Product Documentation – Documentation on all of ServiceNow's features and functionality
- Developer Portal – Portal where developers can get a developer instance and train
- Developer Training – Specific, constantly updated self-paced training classes
- Community – Active community forum where you can ask questions
- Customer Success – Best practice information
- Training Courses – ServiceNow's virtual or in-person training classes

---

**Expert tip**

One of the best resources for your application is existing applications. You can look in your instance and see how others have built and structured their application and learn from their example. Or look through the ServiceNow share site to find applications others have posted.

---

# APPENDIX B – Common Tables within ServiceNow

| Label | Name | Description |
|---|---|---|
| User | sys_user | All ServiceNow instance users |
| Location | cmn_location | List of all user locations. Users are typically associated with a location. |
| Group | sys_user_group | List of all of the groups. Users are typically associated with groups and inherit any security roles associated with the groups. |
| Company | core_company | List of companies that interact with your organization. |
| Role | sys_user_role | List of security roles in the instance. Some will be default roles and some will be created by your organization. |
| Task | task | This is the common base table that gets extended. Task has fields and functionality related to assigning work across teams and individuals, managing the state or the task, and other functions. |