

# INTERNAL: Setup for Process Builder Training

1. Start the [Automation Nomination](#) process in the Catalytic team for each participant. Each should submit at least one process to build in advance of the first day of training. All processes should be evaluated by the trainer.
2. All participants should have a physical folder containing the [action/integration list](#), the [Mapping a Catalytic Process](#) document, **Builder**, **Designer**, and **Pushbot** stickers.
3. Share contents of the [Process Builder Materials](#) folder located in the Customer Success > Process Builder Certification folder. If participants are internal, share folder in Catalytic Team Drive. If external, send .pdf files via email.
4. Slide [Catalytic Certification](#) should be updated with the team name and date.
5. Slide [Overview of the Week](#) should be customized for the week's schedule.
6. Slide [Processes in Use by Your Organization](#) should be customized with successful use cases for the org.
7. Slide [Catalytic Team Setup](#) should be customized with the appropriate team name.
8. Slide [Bosch Process Naming Conventions](#) should be unhidden for Bosch trainings.
9. Slide [Record invoice submission date](#) should be updated with the [Time zone identifier](#) where the training is occurring.
10. Slide [Application Limits](#) should be customized with that team's help site.
11. Slide [Resources](#) should be customized with that team's help site.
12. Send invitations to access the certification team in advance. This access should be terminated promptly after training is complete. Catalytic employees attending the training should do their building in their test teams.
13. You should have an active process running to illustrate basic navigation concepts.



# Catalytic Certification

Bosch  
June 4-6



Catalytic

# Introductions

# Times They Are a-Changin'

Today, 71% of task hours are performed by humans. In 2025, humans will perform less than 50% of all tasks.

Automation and AI will lead to displacement of 75M jobs. But at the same time 133M new jobs are created.

# 45% of work can be automated using current technology.

McKinsey, "Four Fundamentals of Workplace Automation"

## Best for automation

Alerts • Email Triage • Data Processing • System Updates • Document Analysis • Calculation • Follow-Ups • Status Updates • Data Entry • Auto-Responses • Reporting • Data Extraction • Merging Data • Anomaly Detection • Orchestration • Gathering Data • Routing Tickets • Updating Documents • Mass Emails • Transactional Decisions • Reminders



## Best for people

Strategy • Negotiation • Relationships • Creativity • Design • Presenting • Writing • Inventing • Managing People • Defining Vision • Critical Decisions • Mentoring • Inspiring • Copywriting • Brainstorming • UX • Conversations • Software Development • Coaching • Empathy • High-Touch Support • Teaching • Ethical Decisions • Managing Change • Aligning Staff • Strategic Planning

# Catalytic Intelligent Automation Platform



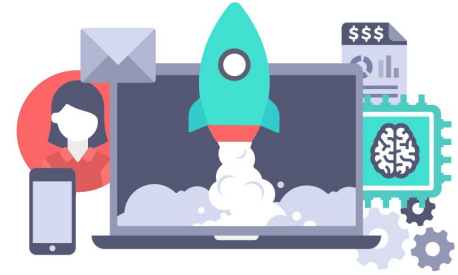
Modern, secure  
cloud SaaS

---



Business user  
friendly

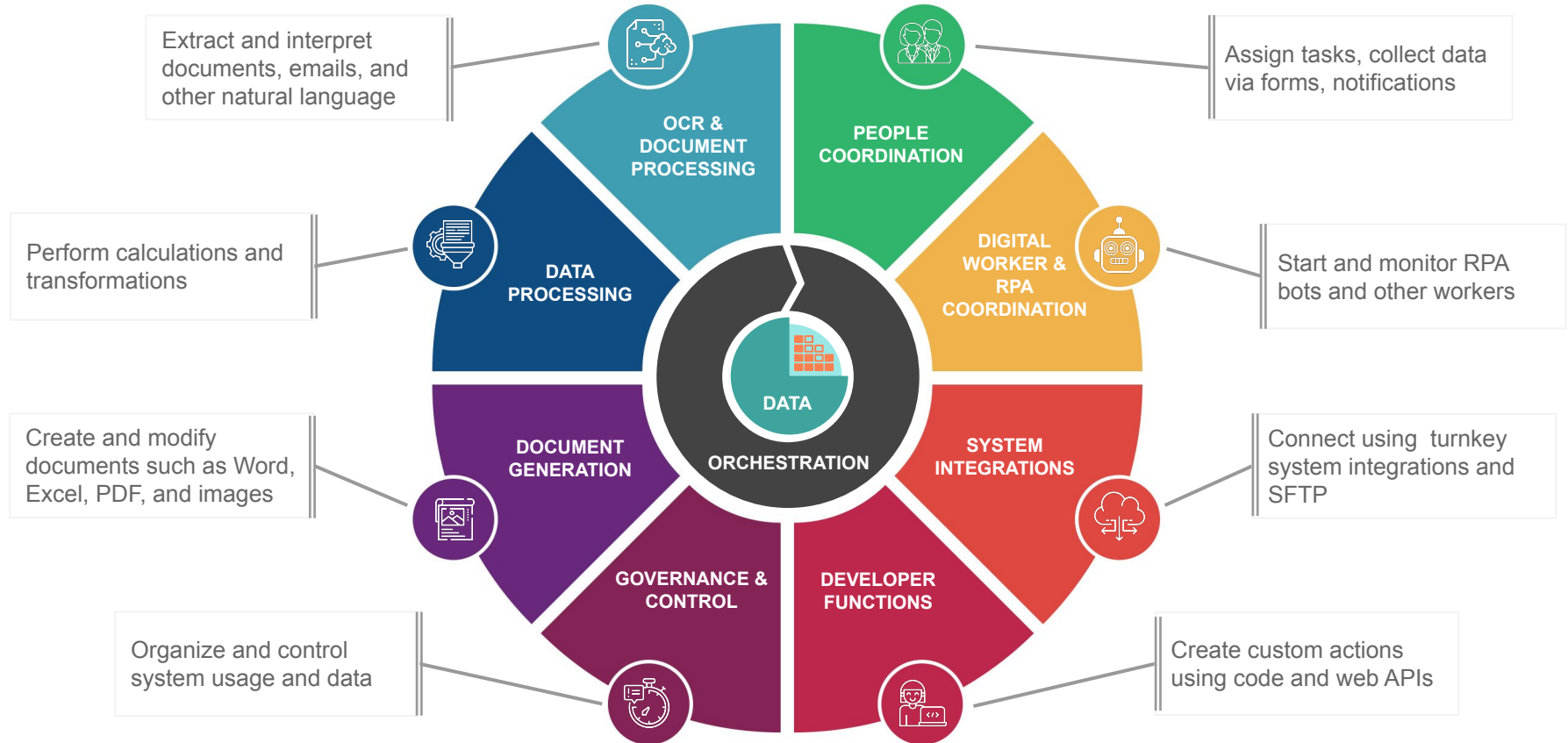
---



Hundreds of  
integrated capabilities

---

# Catalytic has over 210 built-in actions.



# Platform for the Autonomous Enterprise



## AI & Automation Built-in Services



Amazon SageMaker



Google Vision OCR



Rossum Invoice Processing



FullContact Data Enrichment



Twilio SMS



HelloSign E-Signatures



Alexa Web Ranking



SmartyStreets Address Parsing



Mailgun Email



Google Natural Language



ConvertAPI File Conversion



Sauce Labs Browser RPA



PERL



AutoHotKey



UiPath



Automation Anywhere



Blue Prism



AutoIT



PowerShell

## RPA Integration & Desktop Scripting



stripe

ORACLE Cloud



CISCO



DocuSign



workday



## Turnkey Integrations



box

okta





# Bosch Success Cases

## 2 Examples out of 100+

### Supplier Compliance

Saved **200+ days** of effort



Generate 3600+ supplier reports



Email 3600+ suppliers



Send reminders



Track completion

### Contract Term Extraction

Saved **1500+ days** of effort



Import 69000+ contracts



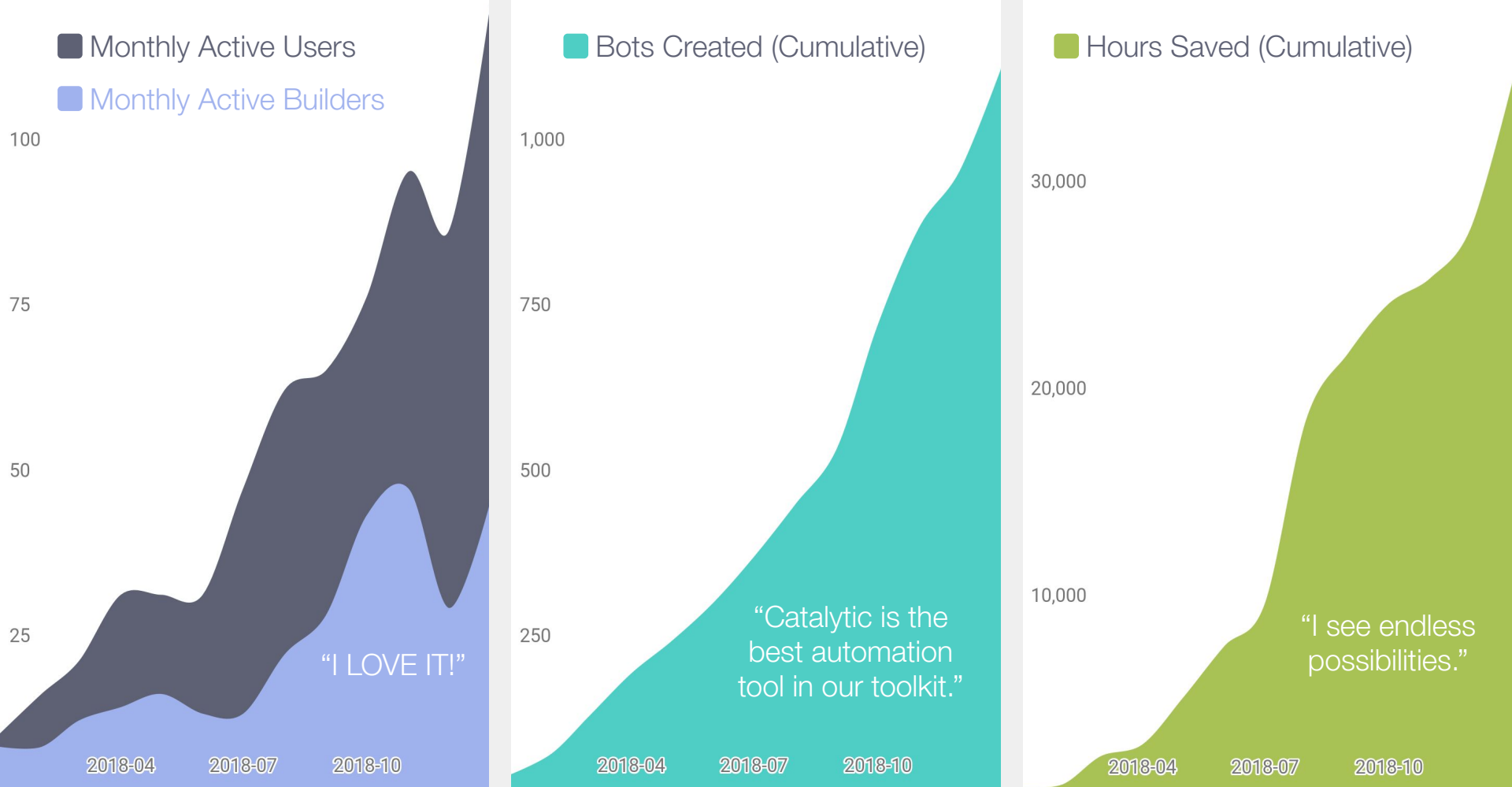
Scan contracts using OCR



Extract & structure key terms

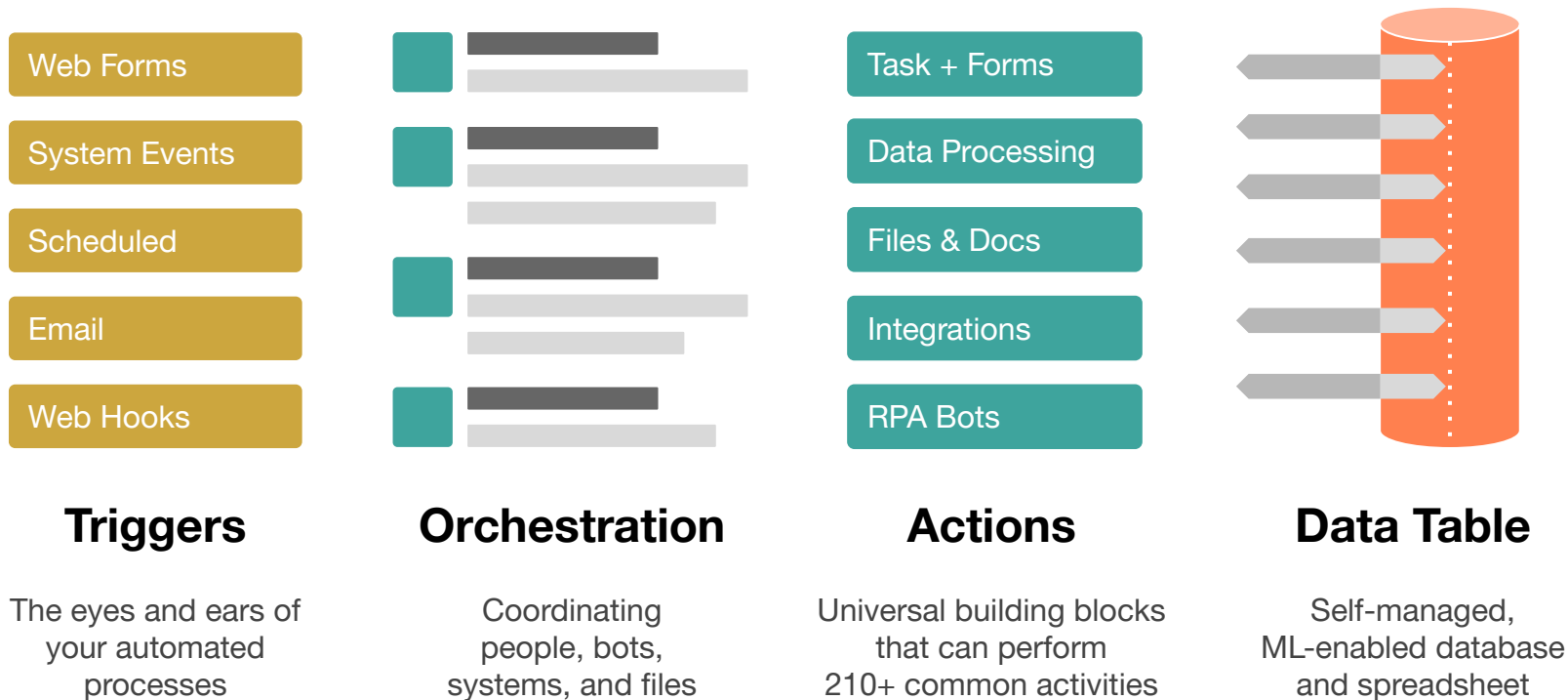


Webform for supervised learning



**1 Year of Catalytic at Bosch**

# Catalytic is an automation cloud.



# Most successful Bosch use cases

SRM  
checkBot  
(Worldwide)

Gratuities  
approval  
(Mexico)

Tender  
Process  
(Hungary)

N2580  
Update  
(Worldwide)

PILUM  
checkBot  
(Rexroth worldwide)

Extracting  
Payment  
Terms  
(Germany)

Create  
Corporate  
Agreement  
(Rexroth)

Monthly  
Time Stamp  
Upload  
(Germany)

Minority  
Business  
Enterprise  
(USA)

Maintenance  
PUR  
Responsible  
(Rexroth)

Mentor  
ButlerBot  
(Worldwide)

SSL  
Certificate  
Renewal  
(USA)

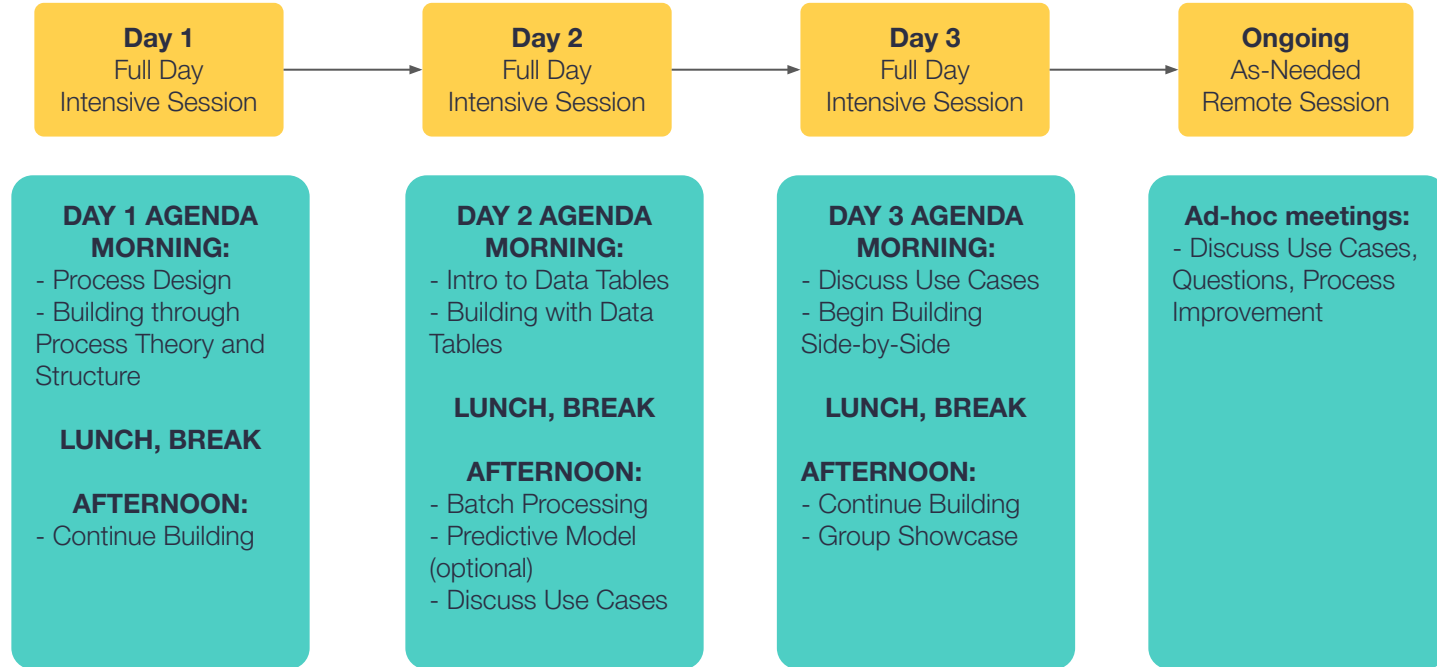
IATF 16949  
Survey  
(Worldwide)

SupplyOn  
Coverage  
Report  
(Germany)

Billing  
Report  
Creation  
(China)

# Demo

# Overview of the Week



# Catalytic Certification Roles

## Process Designer

- Identify opportunities for Catalytic and automation, funnel to Process Builders
- Measure return on investment of implemented processes
- Act as automation evangelists within organization
- Lead workshops, process discovery sessions

## Process Owner

- Monitor processes to ensure they run regularly, correctly
- Track the status of process runs, assign tasks
- Lead basic issue and error troubleshooting
- Report on process usage and outcomes
- Make minor modifications to process templates

## Process Builder

- Configure all commonly used Catalytic actions
- Create complex processes involving a parent process and subprocesses
- Work with a Catalytic Customer Enablement Manager to identify areas for process expansion, optimization

# Process Design







# What Is a Business Process?

# What Is a Business Process?

A business process is a series of tasks in a specific sequence, that delivers value for an internal or external customer. It typically has six characteristics:

- **Definability:** Clear beginning and end: inputs and outputs.
- **Order:** Actions are ordered sequentially.
- **Customer:** There must be a recipient of the output, a customer.
- **Value-Adding:** Data are transformed to add some value for the customer.
- **Embeddedness:** Within an organizational structure.
- **Cross-Functionality:** A process can (but does not always) span several functions.

# What Is a Process Designer?

Process Designers evaluate current processes for automation suitability. They should be able to:

- **Vet** candidate processes for automation within the Catalytic platform
- **Lead** process discovery sessions to ask questions, and guide peers on what's possible within Catalytic
- **Map** process flows, including: inputs, tasks, automation steps, and outputs
- **Partner** with a Process Builder to scope, design, test, and deploy
- **Facilitate** and lead organizational change management
- **Measure** return on investment of implemented processes
- Act as automation evangelists and thought leaders within their organizations

# Processes in Use by Your Organization



# Identifying Opportunities for Catalytic

# Catalytic | Visualizing Automations

## Inputs

### People



### Files



### Systems



## Automation



**Cognitive Automation**



**Data Processing & Transformation**



**Document Automation**



**Integration, RPA & Microservices**

## Outputs

### Communicate



### Create Documents



### Update Systems



**Workflow**



**Data Store**



**Analytics**

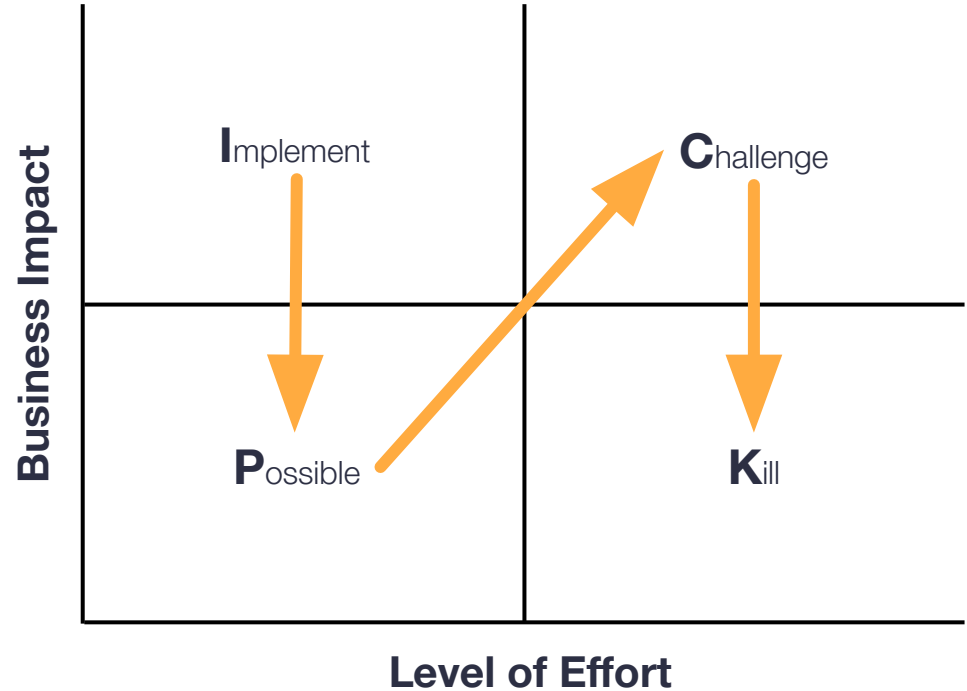
# Strategy Considerations & Use Cases

# Define Automation Strategy

Catalytic allows for automation of many tasks. Your initiatives should align with your business's broader automation strategy. Most organizations fall into one or a combination of the following:

- **Cost Savings**
- **Risk Mitigation**
- **Employee Productivity**
- **Data Fidelity**

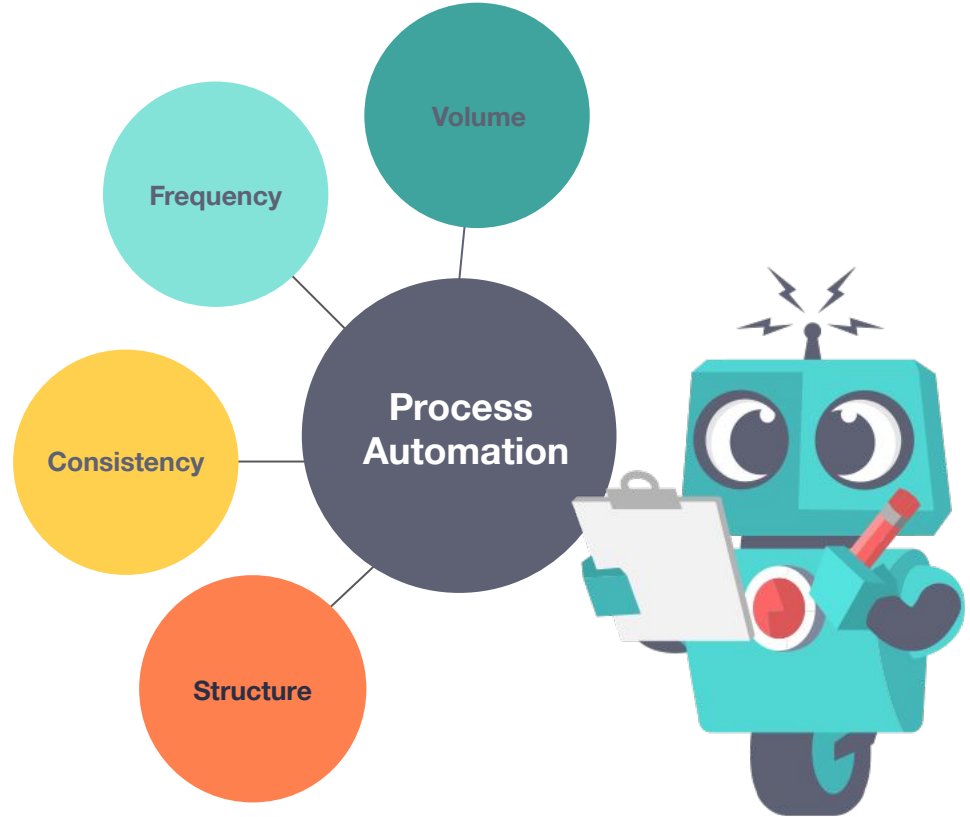
Determine which processes will have the highest impact, and are easy to implement.





# Process Selection Characteristics

- **Volume**  
*Of data processed*
- **Frequency**  
*Of process runs*
- **Consistency**  
*Of process throughout org*
- **Structure**  
*How uniform/logical are data within the process?*



# Defining Return on Investment

**Hard ROI:** Measurements that relate to cost savings or time.

- Time savings per task
- Process cost savings
- Time spent on scheduling/documentation
- Time spent correcting errors
- Savings on operational costs

**Soft ROI:** General efficiency gains or employee/customer satisfaction

- Decrease in risk
- Increased employee productivity
- Improved company morale
- Improvement in team collaboration
- Decrease in the number of bottlenecks
- Improvement in operational visibility
- Reduction in the amount of lost documents

# “Return on Automation” Calculator

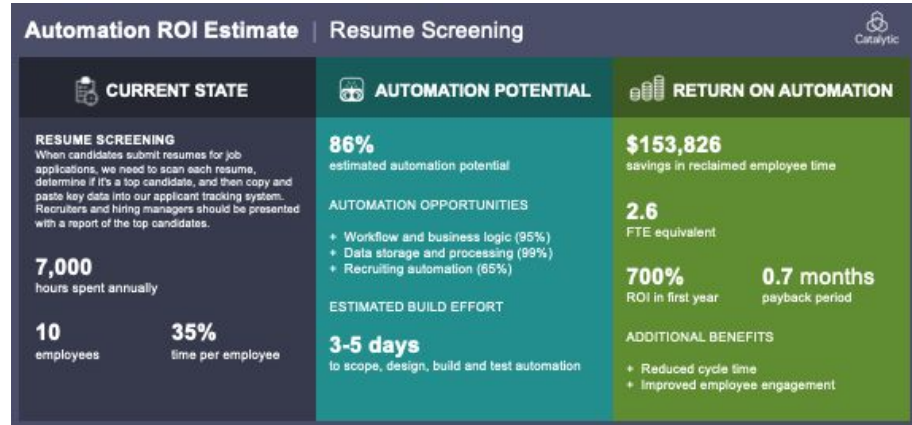
- Catalytic built a return on automation calculator, needing a few fields:
  - Process name
  - Brief description
  - Number of employees involved
  - Percentage of employee time
- The calculator will send you a PowerPoint slide with an estimated return on investment calculation for your process

## **YOUR TURN**

Go to <https://catalytic.pushbot.com/form/roi-calculator> and complete the form to see how your candidate process is scored.

# “Return on Automation” Calculator

- Below is an example of the output of this process.
- Not intended to be comprehensive, just to estimate:
  - Automation potential
  - Estimated build effort
  - Time and cost savings
  - ROI and payback period



# ROA Calculator: How Does It Work?

- Process begins with a **public web form**
- The **description is compared to a database**, and “fuzzy matching” **identifies Catalytic automation features** likely to be used.
- An **average of automation potential is calculated** based on the above.
- **Basic math and formulas** do these calculations.
- Human **tasks are assigned** if figures are outside of pre-identified range.
- Values are **inserted in a PowerPoint template**, which is **sent by email**.

## MORE

Automations like this one are within reach for the average business user, with some baseline data and a bit of creativity. Read more at our blog: [catalytic.com/blog](https://catalytic.com/blog).

Document the Process

# Guiding Questions

- Does the process exist in a standardized format, or is it a theoretical problem for which no good solution exists?
- What is the process at its core?
- Are the data for this process already structured, and reliable?
- What actions need to be completed as part of the workflow?
- What is the volume of this process?
- What is the frequency?
- Can the process be broken down into smaller components?
  - How are the components connected?

# Process Building



Catalytic



# Session 1: Process Builder Certification Overview



# Process Builder Responsibilities

Process Builders are tasked with the following:

- Scoping, building Catalytic processes at the request of the Center of Excellence (CoE), or leadership within one's line of business
- Leading user acceptance testing, deployment of processes
- Working with Designers and Owners to measure return on investment of implemented processes
- Ongoing collaboration with Catalytic Customer Enablement team to identify areas for process optimization, and expansion

# What We'll Cover

## Over the course of the Process Builder Certification you will...

- Build three (3) working processes for the following use cases:
  - Invoice Approval
  - Vendor Contact Information Update
  - Past Due Invoice Reminders
- Incorporate 24 different action types\*
- Use a Catalytic process to maintain a dataset
- Learn to test and troubleshoot process templates
- Understand configurations for commonly used Catalytic actions
- Create a complex process involving a parent process and subprocesses
- Incorporate a predictive model using machine learning on a dataset\*
- Become an expert on your team on Catalytic and its functionality

*\*If optional predictive model module covered.*

# Catalytic Team Setup

1. Accept the email invitation to join the **[team]** team.
2. Complete the account setup for the **[team]** team.
3. When you login you are taken to a landing page. Click around to familiarize yourself with navigation. Try the *Complete your first task*.
4. Go to My Account > Settings and **Enable Beta Features**.
5. Bookmark **[team].pushbot.com** to easily sign in later.

# Application Navigation

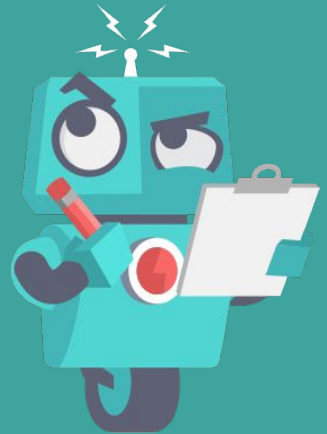
To use Catalytic at the most basic level, you should be able to:

- Access and complete tasks
- Find an active run
- Start a process
- Find data tables

Go to your team page and follow along as I navigate through these concepts.



Questions?



# Session 2: Building through Process Theory and Structure





# Process Theory



# What Is a Process?

As we discussed, nearly all processes have the following characteristics:

- Definition
- Order
- Participants
- Value
- Embeddedness
- Cross-functionality

Think about process in terms of **inputs** and **outputs**: after a sequenced automation, inputs are transformed into some valuable output(s).

# Mapping a Process

1. How does the process start?
2. What are the inputs?
3. What are the desired outputs?
4. What is the order in which steps occur?
5. Who is involved?
6. What role does each participant play?
7. When does the process occur? At what interval?
8. What files or data are involved in the process?
9. What other systems, software or websites are involved?
10. Are there different variations of the process?

## **TIP**

One human action will not necessarily equate to a single machine action. You may need to use multiple actions in sequence to suit your need.



# Process Structure/Building

# Framework: Process Templates

- Actions are combined in order to create a **Pushbot**, also called a **process template**.
- Once built one can run a Pushbot infinitely, using unique inputs (data). This will yield a unique output(s) each time.
- Each time a process is started, that pass through is called a *process instance* or *run*.

## YOUR TURN

1. Create a new process template: **All Pushbots >> Create a Pushbot**
2. Name the template “**Your Name** Invoice Approval Process.”

# Bosch Process Naming Conventions

- Bosch processes should be named with the following:
  - [STATUS] -
    - SANDBOX
    - DEVELOP
    - TESTING
    - PRODUCTIVE
  - Use Case Name -
  - Bot Name -
  - Position in Process Suite
    - Starter
    - Finisher
    - Backup
  - (Department)

## Examples:

*[DEVELOP] - N2580 - Data Collection/Tracker - Starter (DC/LOG)*

*[DEVELOP] - N2580 - Data Collection/Tracker - Finisher (DC/LOG)*

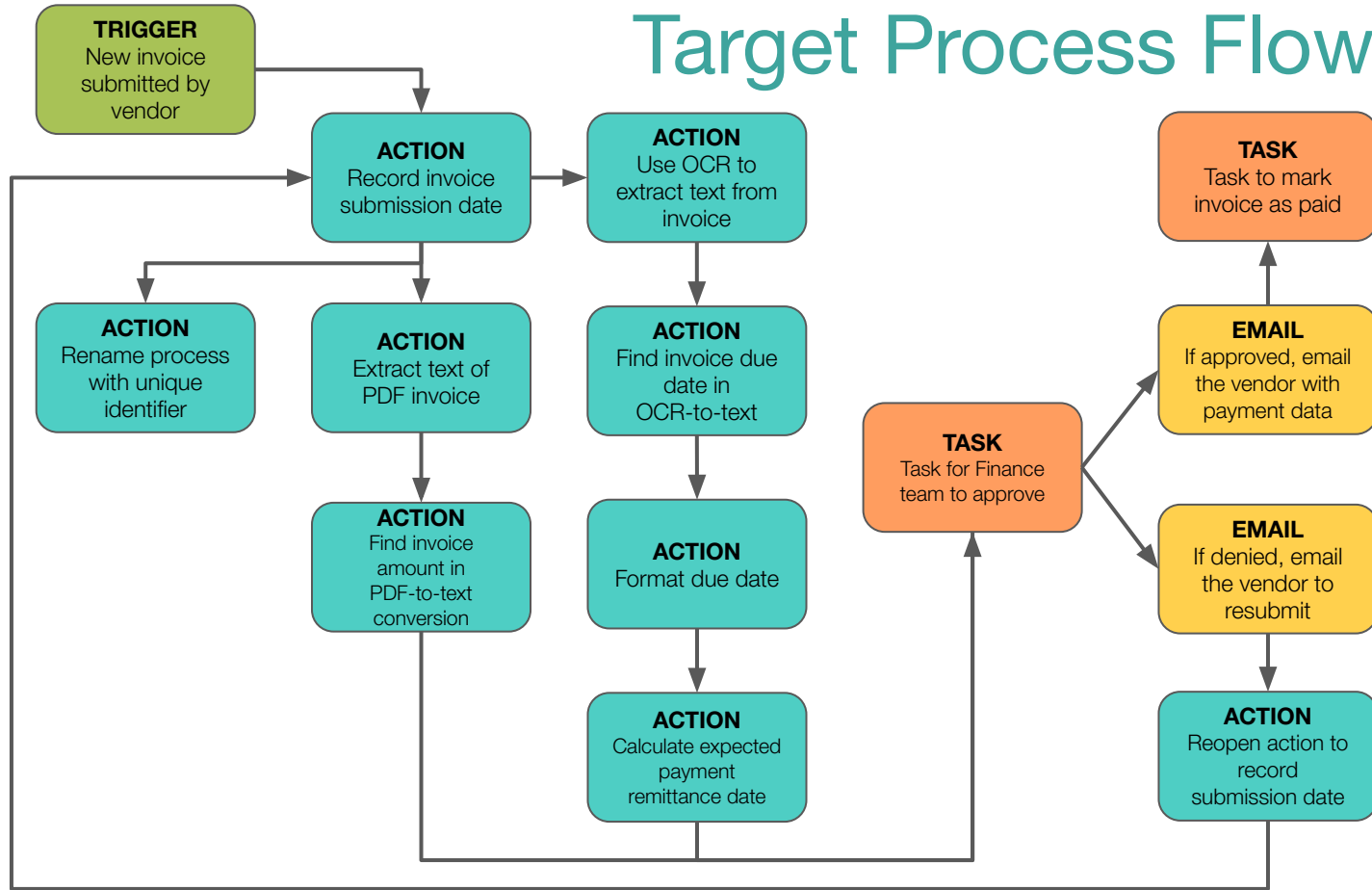
*[DEVELOP] - N2580 - Backup (DC/LOG)*

# Target Process

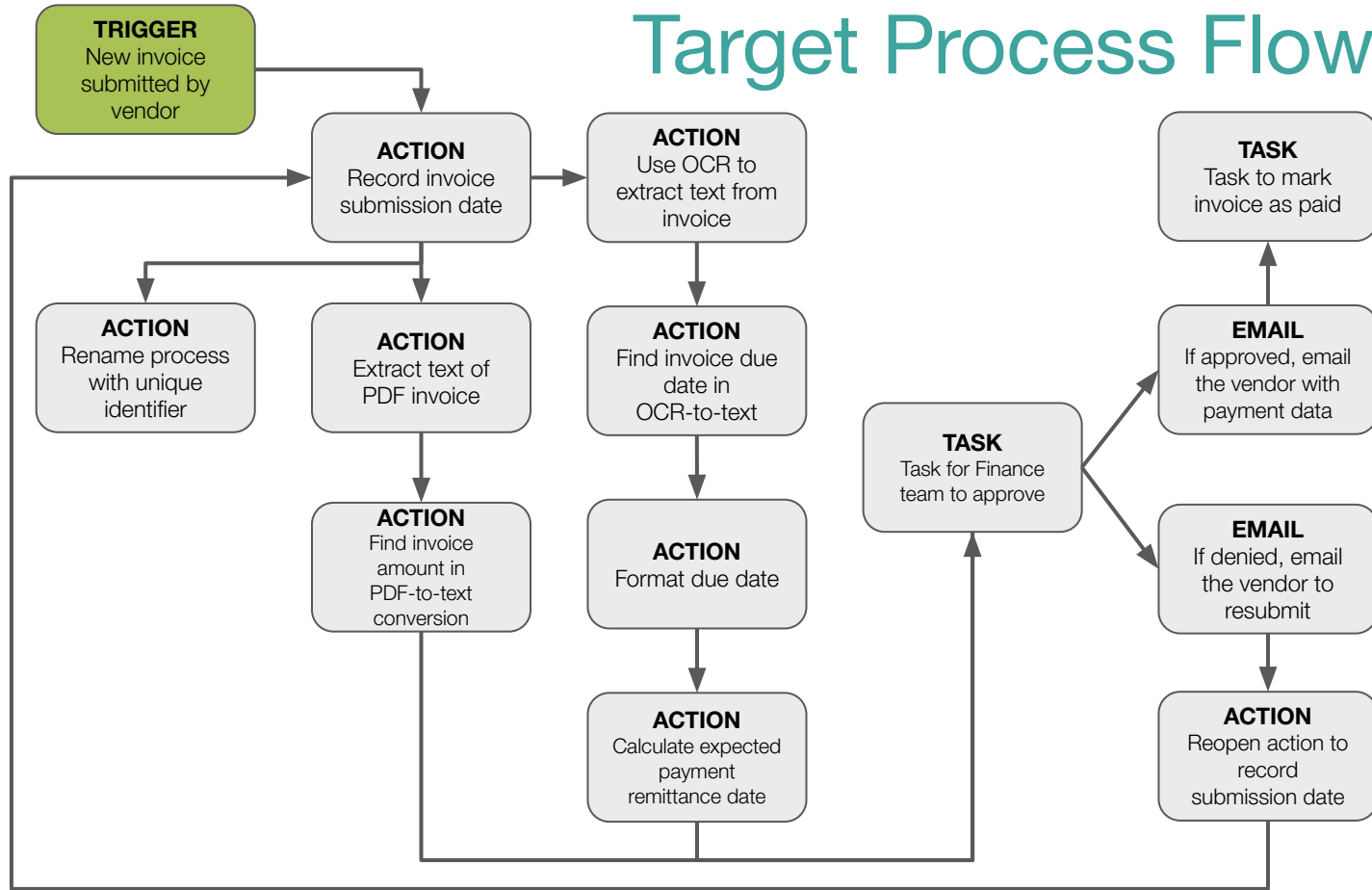
As a group, we're going to build an invoice approval process that will:

1. Start via webform trigger when a vendor submits a new invoice
2. Calculate a recommended payment remittance date based on the due date
3. Send the invoice to the finance team for approval
4. If the finance team denies the invoice, sends the invoice back to the vendor for modifications

# Target Process Flow



# Target Process Flow





# Catalytic Triggers

A *trigger* is the method of starting a process template, creating an instance/run. Processes can be started manually or through an automated trigger. Triggers include:

- **Webform:** Instance starts when a public webform is submitted. Webforms require **template-level fields** to gather data from people.
- **Scheduled:** Instance starts at a specific date and time, or at regular intervals.
- **Email:** Instance starts when someone emails a designated email address.
- **Integration:** Instance starts through Catalytic integration, e.g. Slack or Salesforce.
- **Webhook:** Starts when a POST request is made to a Webhook URL.
- **Manual:** Instance starts ad hoc by clicking **Start an Instance** on the process details page.

# Creating a Web Form Trigger

1. This process uses a web form trigger. To create one, navigate to the **Pushbot Settings** page and select **Edit**, then **Add a Trigger** in the *Triggers* section.
2. If needed, click the **+** at the bottom, right corner and select the **Web Form** trigger type.
3. Complete the required configuration parameters:

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Trigger name</i>	The name the trigger will be referred to as in the full triggers list	*Your Name* Invoice Approval Trigger
<i>Form title</i>	The name that appears at the top of the form end users will see	New Invoice Submission
<i>Form instructions</i>	Instructions for the end user that will appear at the top of the form	Complete below webform to submit invoice.
<i>URL</i>	Designate the end of the URL at which the form will be found	invoicesubmission*yourname*
<i>This trigger will start</i>	The process that will be started when the webform is submitted [*May not display, depending on path to trigger page.]	Select your Invoice Approval Process

**Once created, save the URL for your web form trigger.**

# Template-Level Fields

- **Fields** can collect information from people at the **template level**, or the **action level**.
  - When starting a process, **template-level fields** are entered before any actions start.
  - These fields are completed through a webform, or will appear when you use the **Start an Instance** button.
  - **Action-level fields** are collected as individual actions run.
    - We will return to this concept later on in this build.

# Field Types

- All fields gathered from people can be assigned a **field type**, allowing one to gather a specific type of information. Catalytic field types include:
  - **Text**
  - **Integer**
  - **Decimal**
  - **Date**
  - **Date and Time**
  - **True or False**
  - **Choose One**
  - **Choose Multiple**
  - **Coordinates**
  - **JSON**
  - **File**
  - **Instructions**
  - **Table**
- To change field type, click the field, then select the **Type** configuration
- Fields in human tasks and web forms always default to the **Text** type when first added.
- Automated actions output specific field(s); their type is fixed.

# Field Configuration Options

There are several parameters builders can set when creating fields to be completed by people. For example:

- If a field is **required**, users cannot submit the webform (or task) without completing the field.
- The field **description** is text that displays between the field name and the field box.
- The field **default** assigns a standard value for the field.

# Creating the Webform Trigger

1. To access the template-level fields page, click **Edit**, **Add an Action**, then **:**, and finally **Fields**.
2. Add the following *required* fields (the field type is in parentheses):
  - **Name** (Text)
  - **Email Address** (Text)
  - **Organization** (Text)
  - **Invoice** (File)

## TIP

To change field requirement or type, click the **:** button to the right of the field to see the editing options. You can also click the field to open up the full edit page.

# Field Permissions

- Be aware of the **default field permission** at the template-level, and on individual fields
  - If intaking sensitive data, or if automated tasks result in sensitive data, consider **confidential** or **highly confidential** permission for field
  - Information classification and handling is the builder's responsibility

# Pushbot Permissions

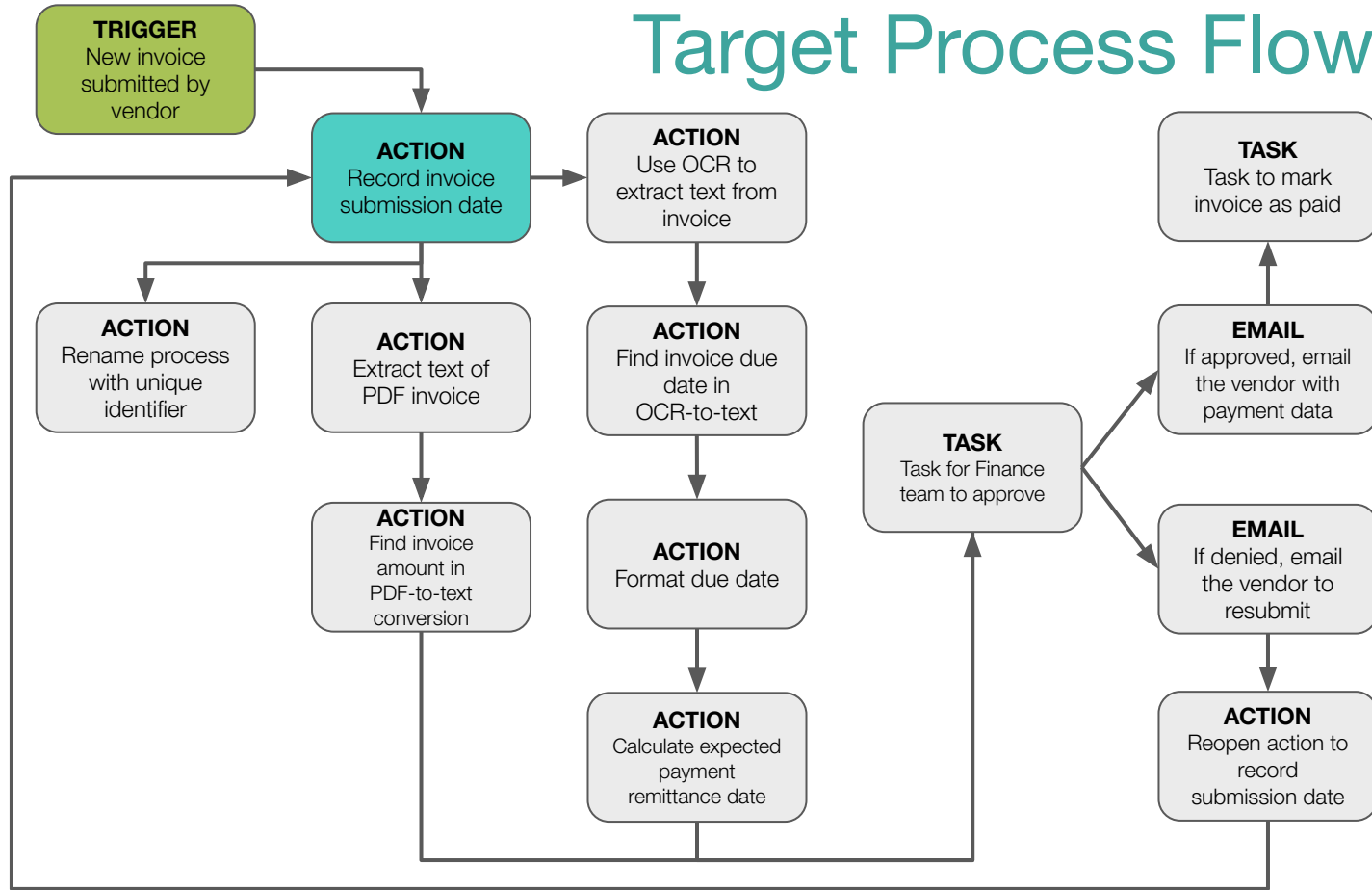
- Pushbots have several permission layers. Process owner can:
  - Control who can **edit** the Pushbot
    - Builder permissions also required
  - Control who can **find and run** the Pushbot
  - Control **default field visibility**, applied:
    - To all new fields added in human tasks, email forms
    - To fields generated automatically when the process runs

## YOUR TURN

Take a moment to update your process so that it is visible only to you.



# Target Process Flow



# Actions and Action Types

- Each step in a process is an **action**. Actions can be:
  - **Automated** (completed by Catalytic)
  - **Manual** (completed by a human)
- Catalytic has over 200 **action types**. When we add a new action to our process, we select the action type to tell the action what to do.
  - Action configuration is similar across action types, but each has different parameters.
- Take a look at your actions/integrations list in your folder. Some commonly used actions include:
  - Email: Send an email
  - Excel: Apply filters
  - Dates: Get current date
  - Word: Create a Word document
  - Tables: Add a row to a data table

# Actions: Naming Conventions

- When adding an action to your process, choose a name that is both **succinct** and **descriptive** in terms of that action's purpose.
- For example, if using the **Email: Send an email** action:
  - ✗ “Send email” (To whom? Why?)
  - ✗ “Email” (To whom? Why?)
  - ✗ “Email finance team to let them know that the invoice has been received” (Too long.)
  - ✓ “Send confirmation email to finance team for approval” (Just right.)

# Action 1: “Record invoice submission date”

## PURPOSE

Record the date the webform was submitted and the process run was started.

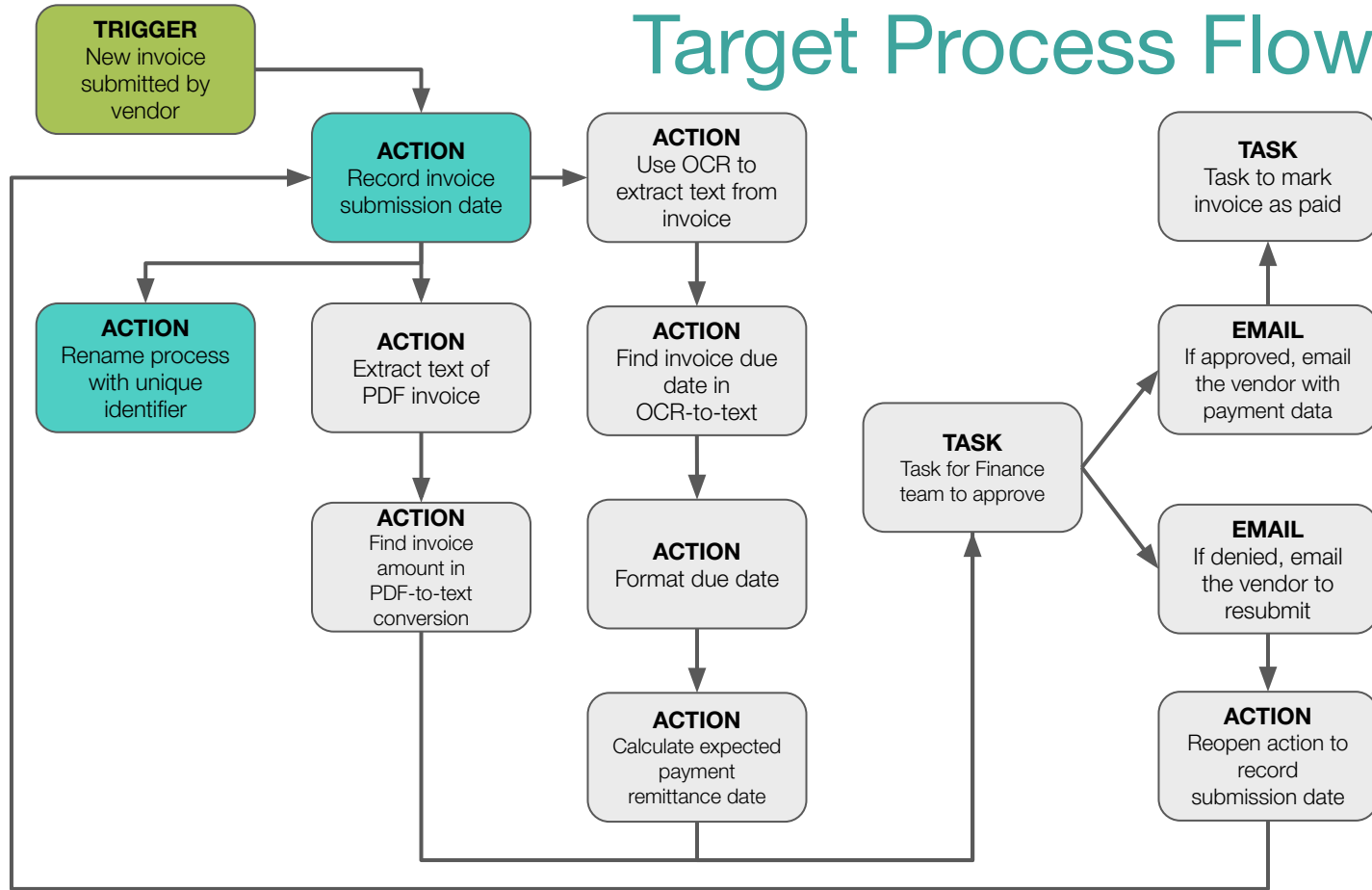
## ACTION TYPE

Date: Get current date

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Timezone identifier</i>	Use the timezone you'd like the date to be recorded for. Written in the format “America/Chicago.”	Europe/Berlin
<i>Return Field Name</i>	This is a common configuration field. This is the name that the result of the action will be saved as.	submission-date

DEPENDENCIES	-
CONDITIONS	-

# Target Process Flow



# Referencing Fields

- When a field is collected in a task or form, or generated by an automated action, it can be referenced as needed.
  - Putting a field name between **{{handlebars}}** indicates you want to reference that field's value
  - Use the handlebar picker **{{}}** to select fields you wish to reference
  - If manually referencing fields, all letters should be lowercase
    - Handful of exceptions for globally available fields (process metadata)
  - Sequential whitespaces and non-alphanumeric characters are replaced with hyphens (-)
  - For example, we could reference the field “First Name” list this:

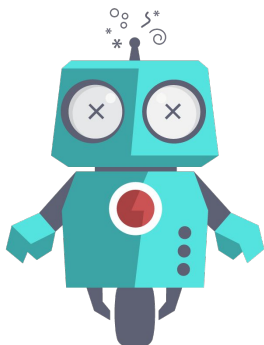
Field: **First Name** —————> Reference: **{{first-name}}**

## YOUR TURN

How might you reference a field called **Need Assistance?**

# Referencing Fields

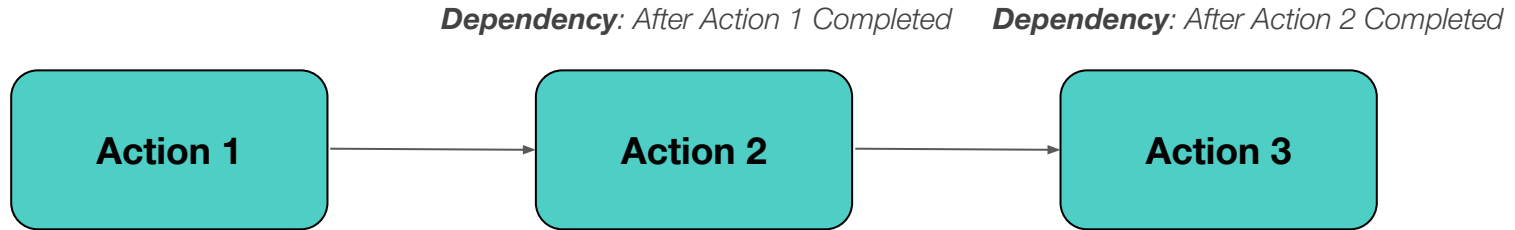
## CAUTION



- Editing the name of a **human** field changes the *display name*, but not the *field reference* name:
  - Web Form Trigger
  - Assign task to a person
  - Email: Send a form
- When you later want to refer to that field in `{{handlebars}}`, you would use the original *field reference* in handlebars.
- Best practice is usually to *delete* human fields instead of renaming them.

# Dependencies

- **Dependency** relationships between actions set process chronology. They determine the order in which actions should start.
- An action should be made dependent upon another action where:
  - The action requires fields collected, generated, or transformed by the other action
  - In workflows, human tasks need to happen in a specific sequence
- If you have three actions, and the **second should happen after the first, and the third after the first and the second**, :



- Note that the difficulty of troubleshooting process errors is closely related to complexity of dependency relationships.



# Actions: Dependencies

- There are three dependency options:
  - **“Start after all of these tasks are completed...”**
    - All of the selected tasks must be *completed* before this task will activate. If any of the selected tasks are skipped, then this task will also be skipped.
  - **“Start after any of these tasks are completed...”**
    - Any one of the selected tasks must be completed before this task will activate. If all of the selected tasks are skipped, then this task will also be skipped.
  - **“Start after all of these tasks are either completed or skipped...”**
    - All of the selected tasks must be resolved in some way. This is useful for tasks that are dependent on two different conditional branches of the process.

## TIP

We can use the **Flowchart** view in Catalytic to verify we've set our dependencies correctly


# Action 2: “Rename process with unique identifier”

## PURPOSE

Name the process run using field data. This allows users to easily search for that run in the Instances list.

## ACTION TYPE

Pushbot: Rename this Pushbot


CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>New name</i>	The name each process run will have. Should include fields in <code>{{handlebars}}</code> to make each name unique.	<code>{{organization}}</code> Invoice, Submitted: <code>{{submission-date}}</code> 

DEPENDENCIES	<b>COMPLETED:</b> Record invoice submission date
CONDITIONS	

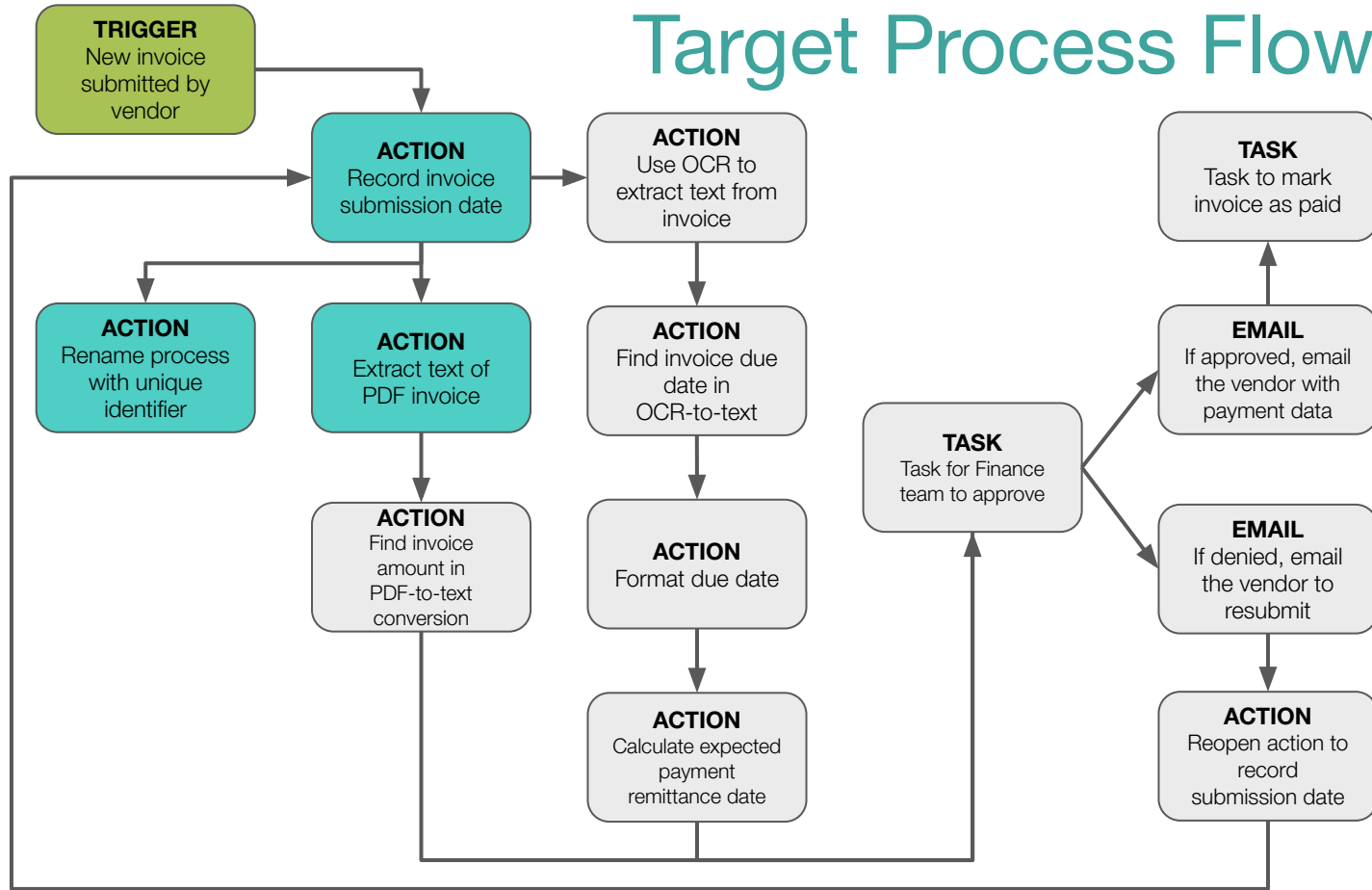
### QUESTION FOR DISCUSSION

Why do we need a dependency on this action but not on the previous one?

### TIP

Use the  icon to select fields collected earlier in the process. Anywhere you see this icon in this presentation indicates when this might be helpful.

# Target Process Flow




# Action 3: “Extract text of PDF invoice”

## PURPOSE

Extract the text of the invoice so that we can find the pieces of information we need later.

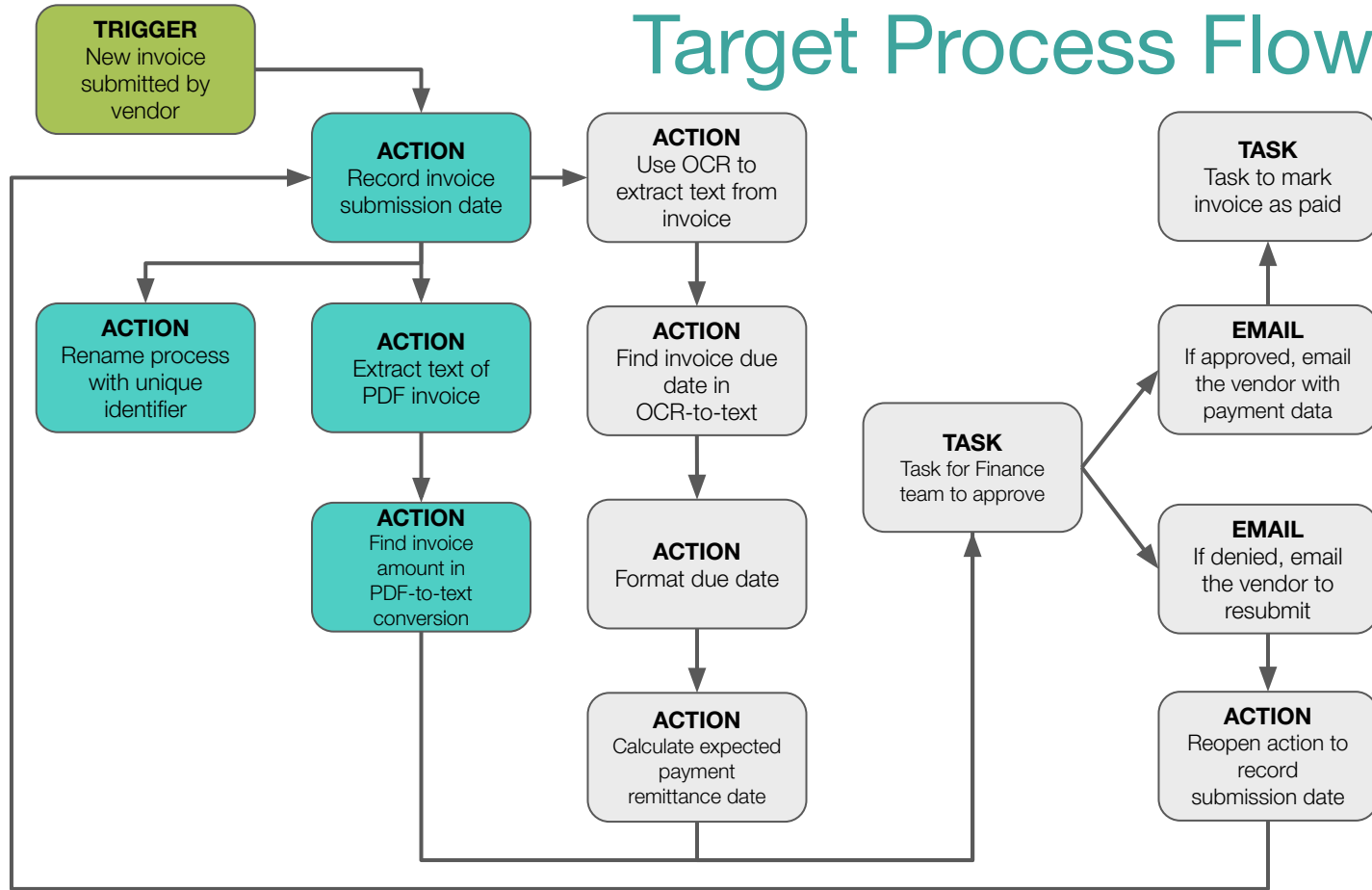
## ACTION TYPE

PDF: Extract text to a field

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>PDF File</i>	The file from which the text will be extracted. Usually will reference a file previously uploaded and saved to a field.	{{invoice}} 
<i>Output field name</i>	Similar to “Return Field Name.” This is the field to which the text will be saved. We’ll reference that field when we want to do something with the text later on.	invoice-text

DEPENDENCIES	COMPLETED: Record invoice submission date
CONDITIONS	

# Target Process Flow



# Action Inputs, Outputs

- All actions require an **input** that is transformed into an **output**.
  - In most instances a **sequence of actions** is required for this transformation
  - The required **input type varies by action type**:

Input	Text	Date	Number(s)	Table
<b>Example Action</b>	<i>Text: Find text next to other text</i>	<i>Date-time: Format a date time</i>	<i>Numbers: Perform basic math</i>	<i>Tables: Apply filter</i>
<b>Output</b>	Specific text value within a body of text	Date expressed in preferred format	Mathematical calculation	A new, filtered table

- If a Catalytic action yields a single output (field), you may be required to designate a *return field*, or *output field name*.
  - This can later be referenced in handlebars **{{ }}**.
- Some actions yield multiple outputs, requiring an *output field prefix*.
  - This is added at the front of the resulting fields, e.g. **search-terms--first-match**
  - There are two hyphens (--) between the output field prefix and the field name.

# Example Outputs

Action type: **Tables: Look up data in a column**

Output field prefix: **lookup**

Note the double  
hyphens (--) in all  
field names

Note all fields  
begin with the  
same prefix



test action

null

lookup--Number of matching rows  
1

lookup--Customer Number  
67,891,234

lookup--PO Number  
526,878

lookup--Invoice Amount  
47,000

lookup--Invoice Date  
5/19/17

lookup--Invoice Number  
128

lookup--Customer Name  
Customer 6

lookup--Business Owner Email  
Stuart+1@catalytic.com

lookup--Business Owner Name  
Ben

[View this test task in a new tab](#)


# Action 4: “Find invoice amount in PDF-to-text conversion”

## PURPOSE

Find the invoice amount in the text extracted from the PDF invoice and save it to a field.

## ACTION TYPE

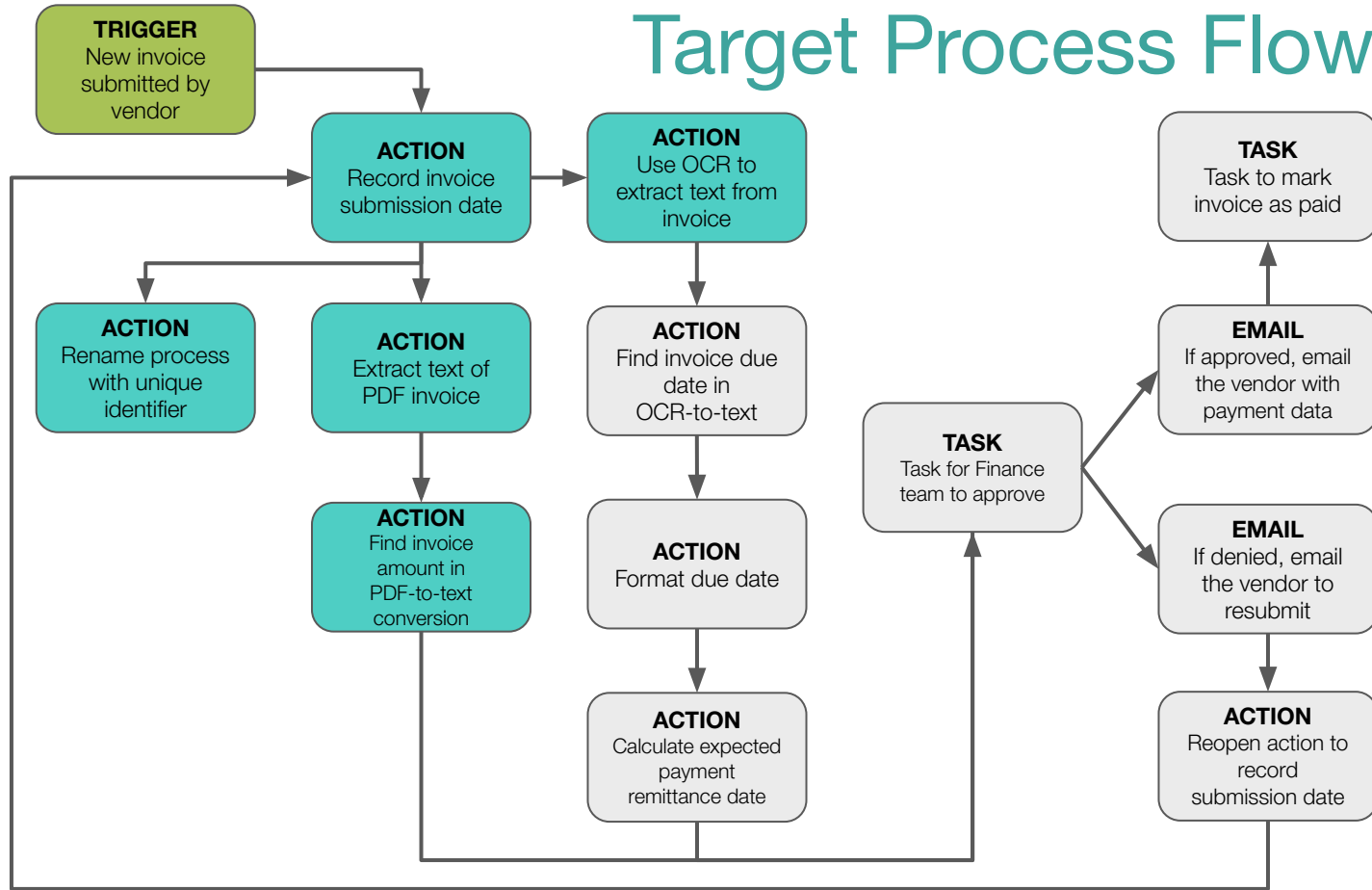
Text: Find text next to other text

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Text to search</i>	The text in which we will look for certain text next to other text	{{invoice-text}} 
<i>Start type</i>	Designates where you'll be looking in the text.	after-text
<i>Start value</i>	What Catalytic should look for at the start of the text we're searching.	Amount Due \$
<i>End type</i>	Designates where to stop looking in the text.	word-end
<i>End value</i>	What Catalytic should look for at the end of the text we're searching	
<i>Case sensitive</i>	Designates whether you are searching for an exact match	FALSE
<i>Output field prefix</i>	This addends a prefix to the field outputs from the action. The first matching value from this step can be referenced as follows: {{amount-due--first-match}}	amount-due

DEPENDENCIES	COMPLETED: Extract text of PDF invoice
CONDITIONS	



# Target Process Flow




# Action 5: “Use OCR to extract text from invoice”

## PURPOSE

Extract the text of the invoice so that we can then find the pieces of information we need later.

## ACTION TYPE

Images: Optical character recognition (OCR)

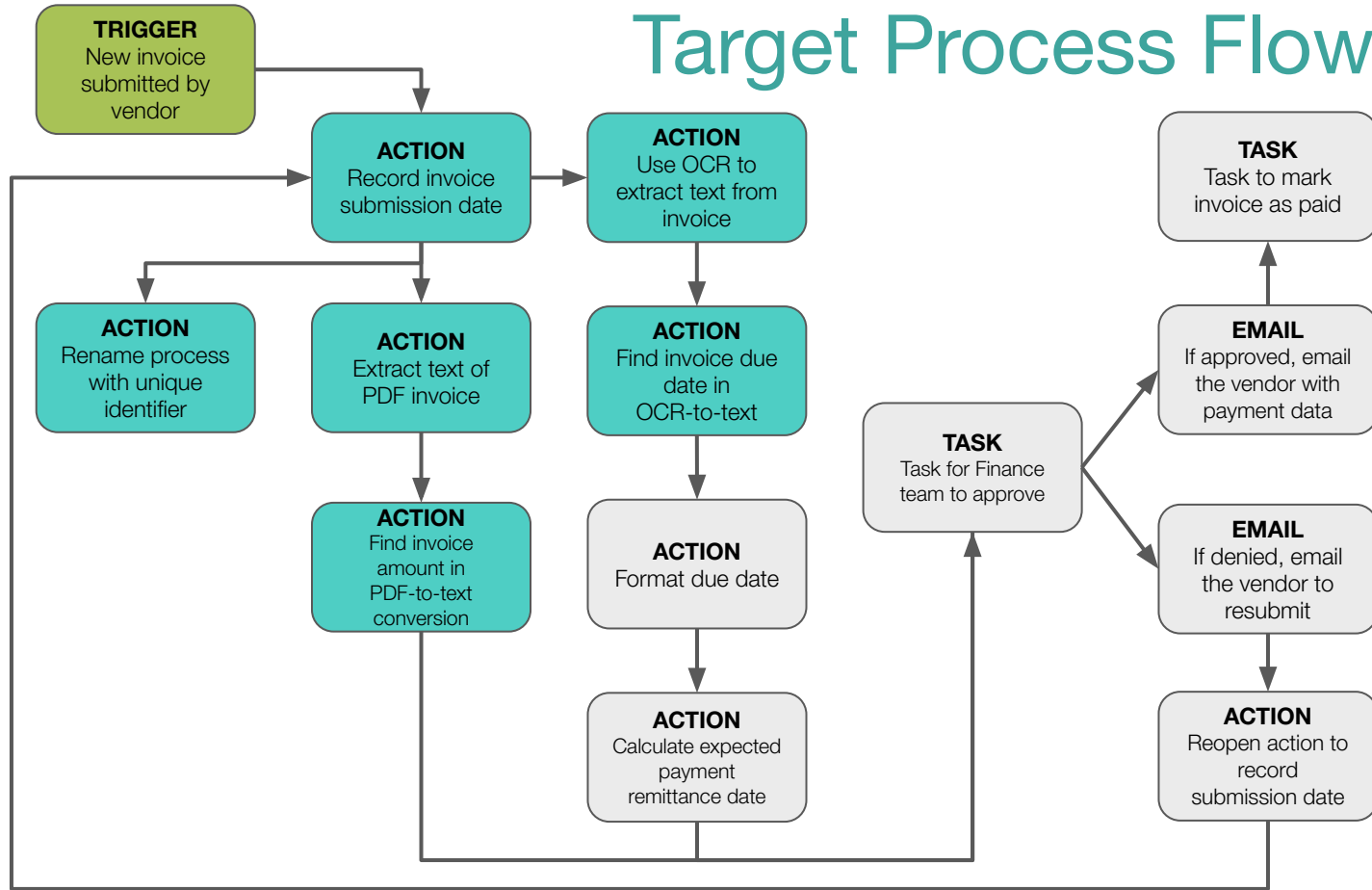
CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Image File</i>	The file from which the text will be extracted. This action works with PNG, PDF, JPEG, or JPG files.	{{invoice}} 
<i>Include output data table</i>	Add an additional output in the form of a data table	FALSE
<i>Output field name</i>	Similar to “Output Field Name.” This is the field to which the text will be saved. We’ll reference that field when we want to do something with the text later on.	invoice-ocr

### TIP

Preview and test this action to see the fields that are returned

DEPENDENCIES	COMPLETED: Record invoice submission date
CONDITIONS	

# Target Process Flow




# Action 6: “Find invoice due date in OCR-to-text”

## PURPOSE

Find the invoice due date in the text extracted from the PDF invoice and save it to a field.

## ACTION TYPE

Text: Find text next to other text

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Text to search</i>	The text in which we will look for certain text next to other text	{{invoice-ocr--horizontal-scan}} 
<i>Start type</i>	Designates where you'll be looking in the text.	after-text
<i>Start value</i>	What Catalytic should look for at the start of the text we're searching.	Due Date:
<i>End type</i>	Designates where to stop looking in the text.	line-end
<i>End value</i>	What Catalytic should look for at the end of the text we're searching	
<i>Case sensitive</i>	Designates whether you are searching for an exact match	FALSE
<i>Output field prefix</i>	This is the name of the field where the results will be saved	due-date

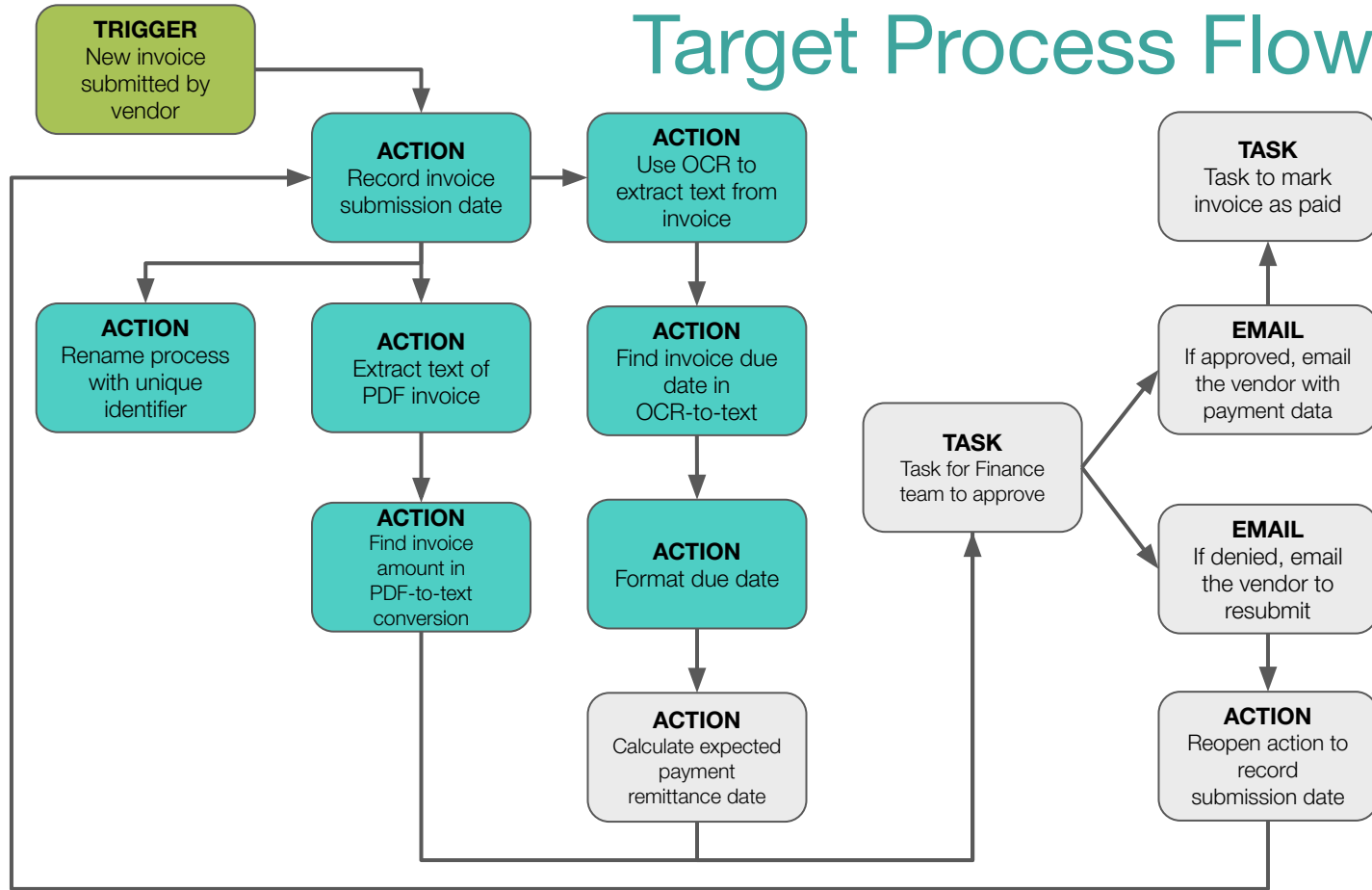
DEPENDENCIES	COMPLETED: Use OCR to extract text from invoice
CONDITIONS	

# Deep Dive: Extracting Text from a .PDF

Two actions are available to extract text from a .pdf file:

<b>PDF: Extract text to a field</b>	<b>Images: Optical character recognition (OCR)</b>
Converts a .pdf file to into a large block of text	Provides a horizontal scan, vertical scan, and data table breakdown
Better for shorter .pdf files (a few pages)	Better for longer .pdfs (many pages)
Only for electronically generated .pdfs	Can be used for hand-scanned .pdfs, images, and handwritten text; requires high-quality scans
Used to search for simple text patterns	Used to search for complex text patterns

# Target Process Flow



# Action 7: “Format due date”

## PURPOSE

Reformat the due date that was submitted into the format we’d like to present it in.

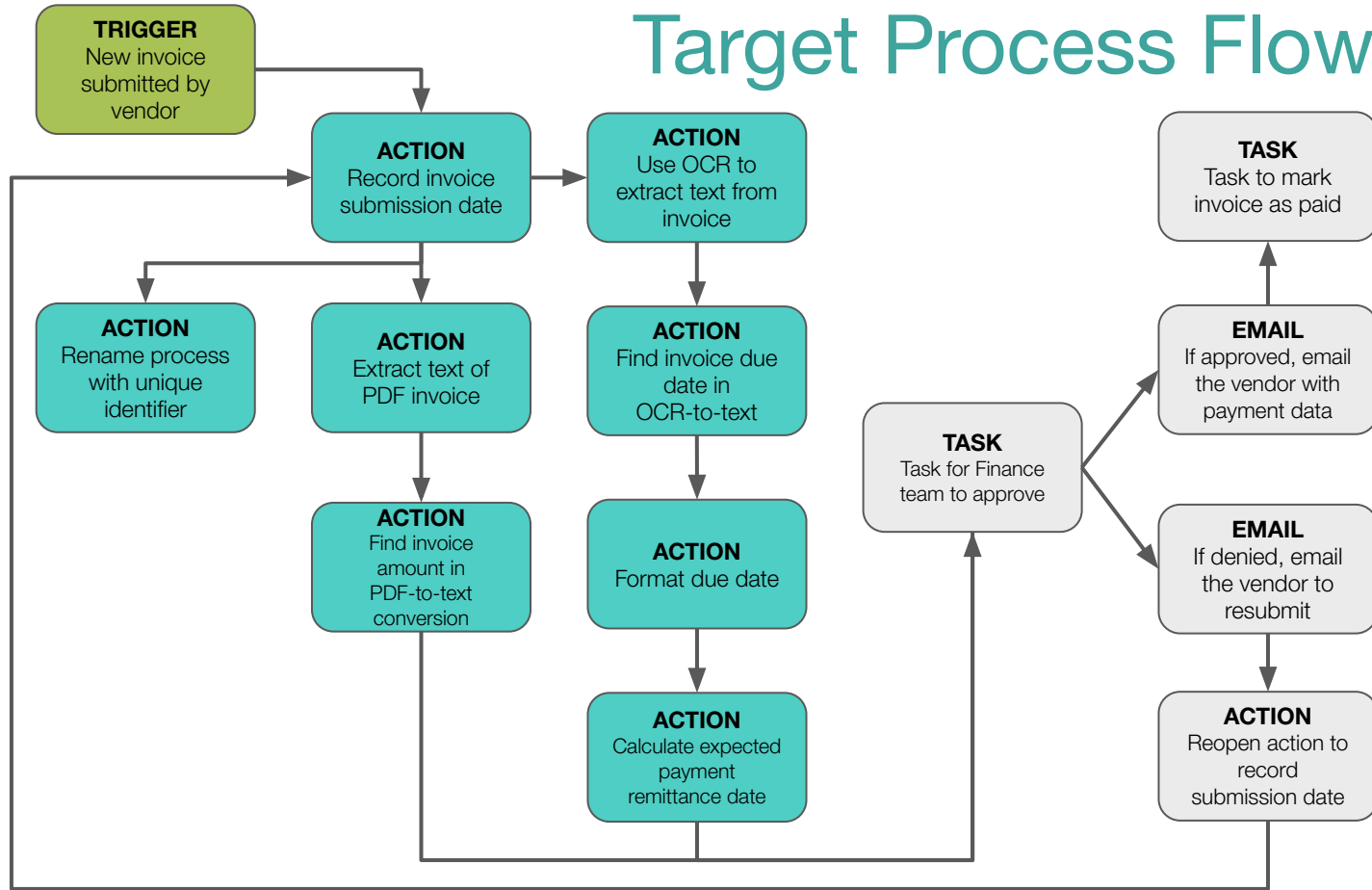
## ACTION TYPE

Dates: Format a date time

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Input</i>	The date that will be reformatted. Usually will reference a date previously collected and saved to a field.	{{due-date--first-match}}
<i>Format</i>	The format you’d like the date to be saved in.	YYYY-MM-DD
<i>Offset</i>	Used to offset the date from UTC	
<i>Return field name</i>	This is the name of the field where the reformatted date will be saved	formatted-due-date

DEPENDENCIES	<b>COMPLETED:</b> Find invoice due date in OCR-to-text
CONDITIONS	

# Target Process Flow






# Action 8: “Calculate expected payment remittance date”

**PURPOSE**

Calculate date five (5) days before due date to suggest as recommended payment date.

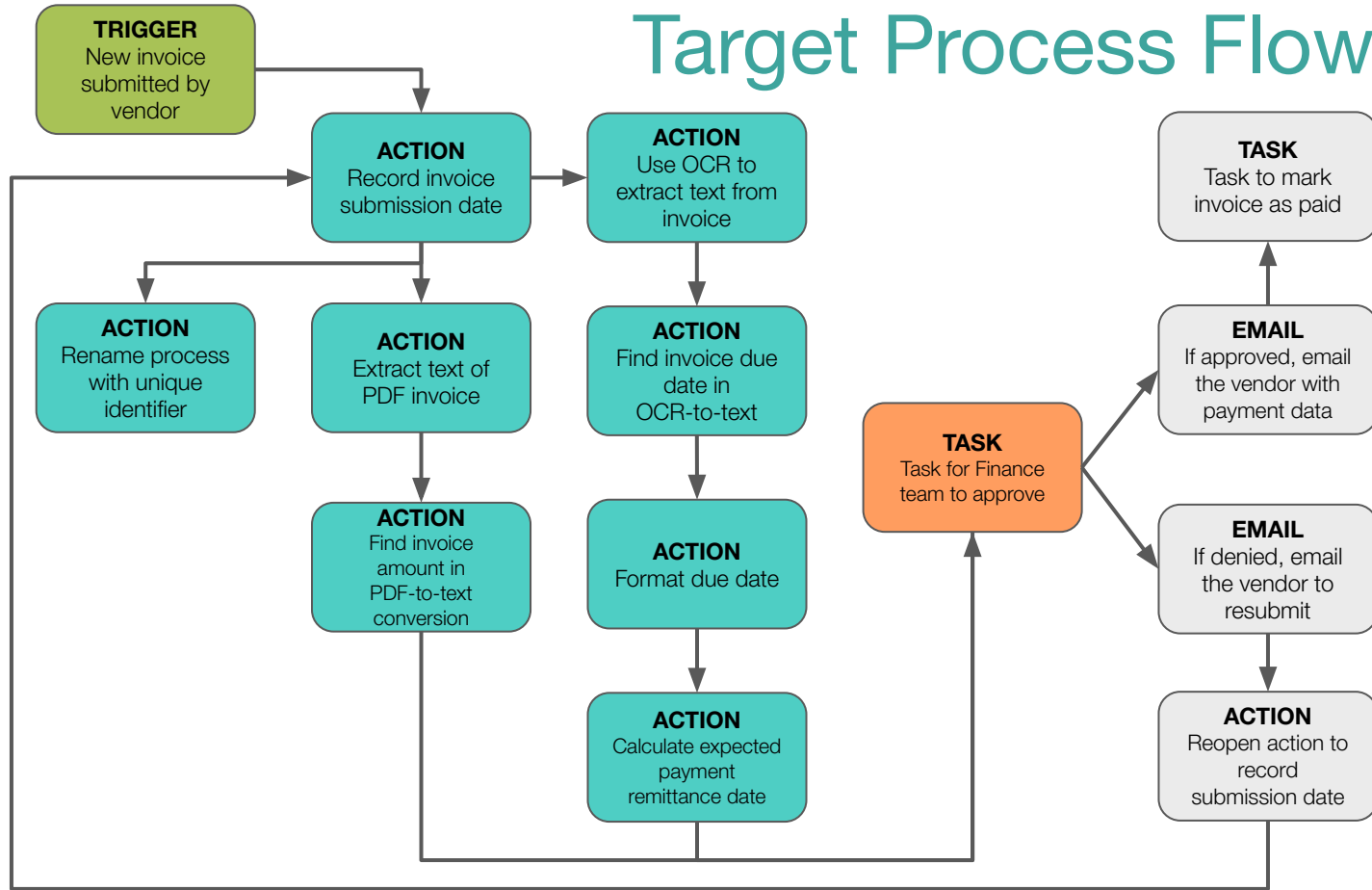
**ACTION TYPE**

Dates: Adjust a date

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Original date</i>	The date that will be adjusted. Usually will reference a date previously collected and saved to a field.	{{formatted-due-date}} 
<i>Adjustment amount</i>	How many days/business days you'd like to add or subtract	-5
<i>Units</i>	Can use DAY or BUSINESS for days or business days	BUSINESS
<i>Return field name</i>	This is the name of the field where the adjusted date will be saved	payment-remittance-date

DEPENDENCIES	COMPLETED: Format due date
CONDITIONS	

# Target Process Flow




# Action 9: “Task for Finance team to approve”

## PURPOSE

Route the invoice to the Finance team requesting approval or denial.

## ACTION TYPE

Assign task to a person

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Assign to</i>	Whom the task will be assigned to, individual or group. Can use a Catalytic username or an email address.	*Select yourself*
<i>Instructions</i>	Any instructions for the task assignee. Can include field references in {{handlebars}}.	Please review and approve or deny the invoice for {{organization}} in the amount of \${{amount-due--first-match}}.  {{invoice}} 
<i>Fields</i>	Any form fields the assignee should complete. Works the same as fields in a webform or emailed form.	[See following slides]
<i>Deadline</i>	The date by which you prefer the task be completed.	

# Fields: Template-Level vs. Action-Level

## Refresher:

- **Fields** can collect information at the **template-level**, or the **action-level**.
  - When starting a process, **template-level fields** gather information from people *before any actions start*.
  - These fields are completed through a webform, or will appear when you use the **Start an Instance** button.

## New Concept:

- **Action-level fields** are collected *after the process starts, when an action runs*. Actions that can collect human inputs are:
  - Assign a task
  - Email: Send a form

# Field Conditions

Field **conditions** allow for a field to appear conditionally based on an input in a previous field(s) in the form.

- Multiple conditions can be used and joined (and/or)
- Field conditions tend to be written as **[field] [operator] [value]**
- Example, only display field(s) if:
  - **fields['approved'] is equal true OR**
  - **fields['number'] is greater than or equal 1**

# Action 9: “Task for Finance team to approve”

## PURPOSE

Route the invoice to the Finance team requesting approval or denial.

## ACTION TYPE

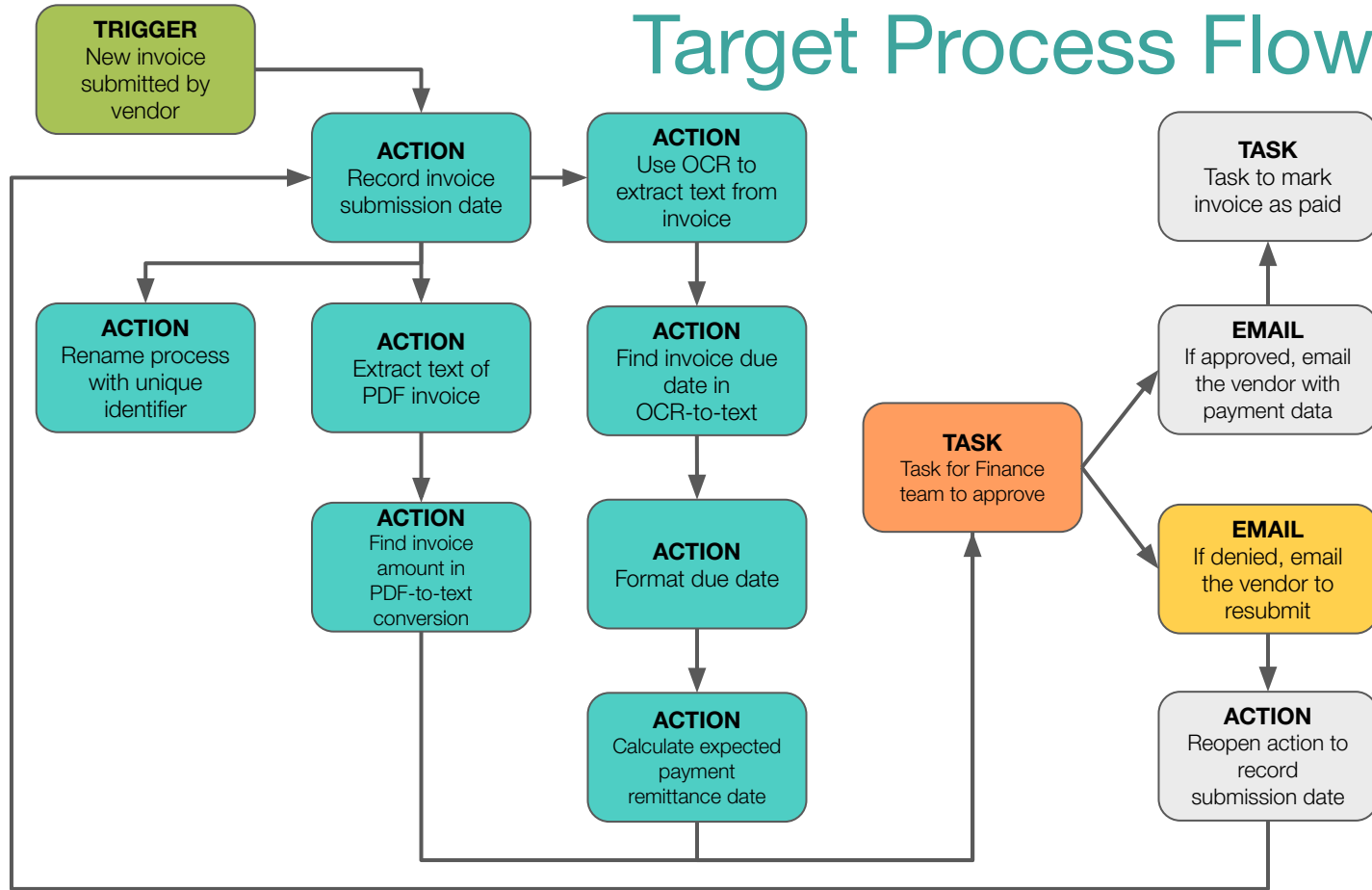
Assign task to a person

### Add the following required fields:

NAME	TYPE	CONDITIONS
<i>Invoice approved</i>	Choose One (Approved; Denied)	
<i>Please provide a reason for denial</i>	Text	fields[“invoice-approved”] IS EQUAL Denied
<i>Payment remittance date</i>	Date	fields[“invoice-approved”] IS EQUAL Approved

DEPENDENCIES	<b>COMPLETED:</b> Find invoice amount in PDF-to-text conversion; Calculate expected payment remittance date
CONDITIONS	

# Target Process Flow



# Action Conditions

Action **conditions** designate whether an action should start. If the condition(s) is unmet, the action does not start.

- Multiple conditions can be used and joined (and/or); if using more than 3-4 conditions, you may wish to consider distinct actions.
- Conditions are assessed **after** dependencies.
- Action conditions often leverage field values, and are written as e.g.  
**[field] [operator] [value]**
- Example, only start action if:
  - **fields['approved'] is equal true AND**
  - **fields['number'] is greater than or equal 1**



# Actions: Conditions

Some examples of where you might use action conditions:

- If a vacation request is approved, notify the requestor, and add a meeting to a team calendar
- If an expense is over \$10,000, assign a task to a manager to review the expense
- If the email contains the word "refund," start a refund process

# Deep Dive: Fields and Single Name Space

## DID YOU KNOW?

- Catalytic uses **single name space** for fields. When the same name is assigned to two fields, they act as the same field. For example:
  - In our process, we have **Invoice** in the initial webform and will have an **Invoice** field in the form sent to the vendor to resubmit.
  - Since these are given the same name, the second **Invoice** will inherit all of the configuration of the first, and the new file will replace the first file uploaded.
- In practice:
  - Use single name space to update the contents of a field throughout the process.
  - **Do not use the same name for a field if the contents of the field should not be replaced.**


# Action 10: “If denied, email vendor to re-submit invoice”

## PURPOSE

Send the vendor an email stating that the invoice was not approved and should be re-submitted.

## ACTION TYPE

Email: Send a form

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>To address</i>	The address to which the email will be sent. Can include an email previously collected in {{handlebars}}.	{{email-address}} 
<i>Sender name</i>	The name that will appear as the sender in the recipient’s inbox.	XYZ Corporation
<i>Email subject</i>	The subject of the email that will be sent to the recipient from Catalytic	Invoice Denied

DEPENDENCIES	COMPLETED: Task for Finance team to approve
CONDITIONS	fields["invoice-approved"] IS EQUAL Denied

### QUESTION FOR DISCUSSION

Why do we need a condition on this action?

### TIP

Use “if” in your action names as a reminder for when you need to include a condition.


# Action 10: “If denied, email vendor to re-submit invoice”

**PURPOSE**

Send the vendor an email stating that the invoice was not approved and should be re-submitted.

**ACTION TYPE**

Email: Send a form

CONFIGURATION FIELD	DESCRIPTION	USE HERE
Email body	The body of the email that will be sent to the recipient	
Attachments	Any attachments that should be included. Usually will be a file uploaded during a process.	{{invoice}} 
Form title	The heading that will appear at the top of the form sent to the email recipient.	Re-submit Invoice
Output field prefix	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	resubmit-invoice

**Add the following required fields:**

NAME	TYPE	CONDITIONS
Invoice	File	

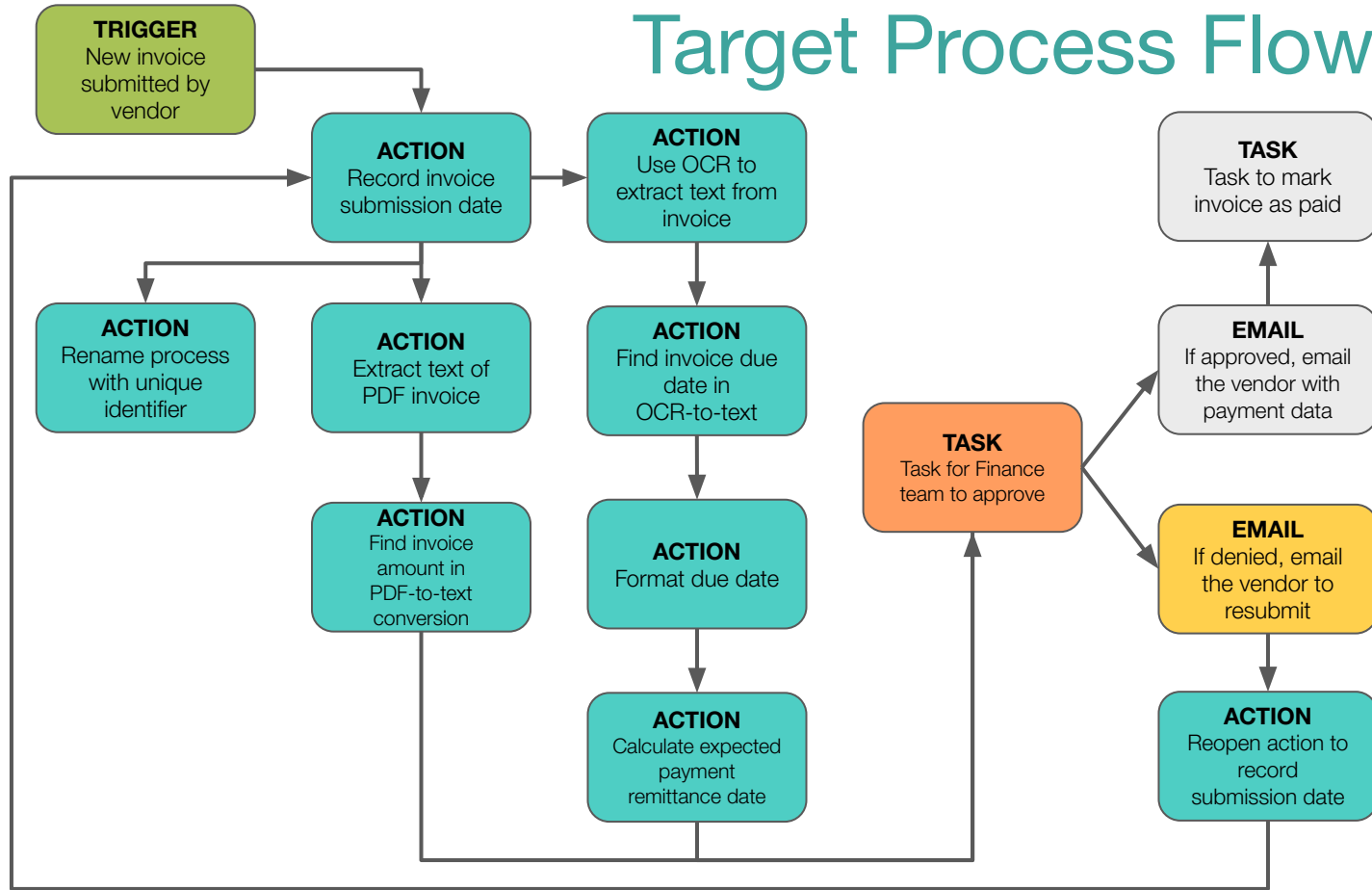
# Assign Task to a Person vs. Email: Send a Form

- Take care using **Email: Send a form** action, web form-type triggers
  - No authentication to access and complete these forms
  - If default values or descriptions use handlebar references, those field values will be visible without authentication

# Assign Task to a Person vs. Email: Send a Form

Capability	In-App Task	Email Form
Get task delivered via email	✓	✓
Fill out fields, upload files	✓	✓
Audit security through SSO authentication	✓	✗
Capture who completed a task	✓	✗
Get task in app	✓	✗
Manage all tasks in one place	✓	✗
See overall instance progress	✓	✗
Comment on an instance	✓	✗

# Target Process Flow



# Action 11: “Reopen action to record submission date”

## PURPOSE

Creates a loop in the process so that every time an invoice is re-submitted by the vendor to the Finance team it is re-sent for approval.

## ACTION TYPE

Pushbot: Reopen task and reset dependent tasks

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Task name</i>	The name of the task that will be reopened by this action. We type the task name in this format: “This is a task name” → <b>this-is-a-task-name</b>	record-invoice-submission-date
<i>Mark complete if</i>	This is the condition under which we want the loop to stop. We write this as a conditional statement. In this instance this configuration is optional, as we have a condition in the previous step.	

### TIP

You can find the task name in the format required above by referencing the URL on that task page.

DEPENDENCIES	<b>COMPLETED:</b> If denied, email the vendor to re-submit invoice
CONDITIONS	



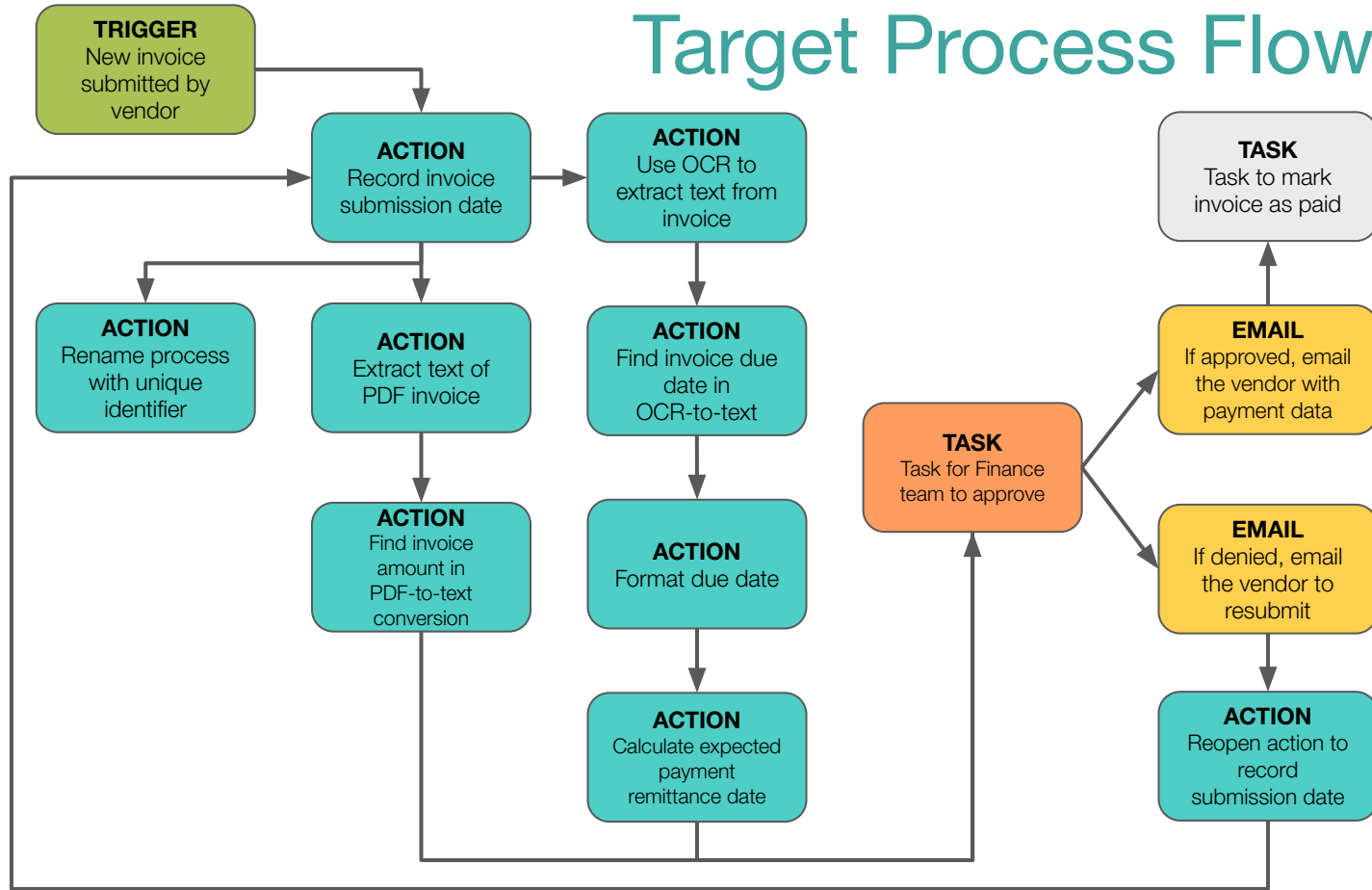
# Deep Dive: Creating Feedback Loops

- We use **Pushbot: Reopen task and reset dependent tasks** to create process loops
- Loops allow for feedback to be gathered, and subsequent responses to that feedback, without needing to leave Catalytic
- Loops are most commonly used for approvals

## QUESTION FOR DISCUSSION

Can you think of some processes at your organization that might use a loop? What task would be reopened?

# Target Process Flow




# Action 12: “If approved, email the vendor with payment date”

## PURPOSE

Once the invoice has been approved, notify the vendor of approval and the expected payment remittance date.

## ACTION TYPE

Email: Send an email

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>To address</i>	The address to which the email will be sent. Can include an email previously collected in {{handlebars}}.	{{email-address}} 
<i>CC</i>	Any email addresses to be cc'd on the email	
<i>Sender name</i>	Who the email will appear to be from in the recipient's inbox	XYZ Corporation
<i>Email subject</i>	The subject of the email that will be sent to the recipient from Catalytic	Invoice Approved

DEPENDENCIES	COMPLETED: Task for Finance team to approve
CONDITIONS	fields["invoice-approved"] IS EQUAL Approved

# Action 12: “If approved, email the vendor with payment date”

## PURPOSE

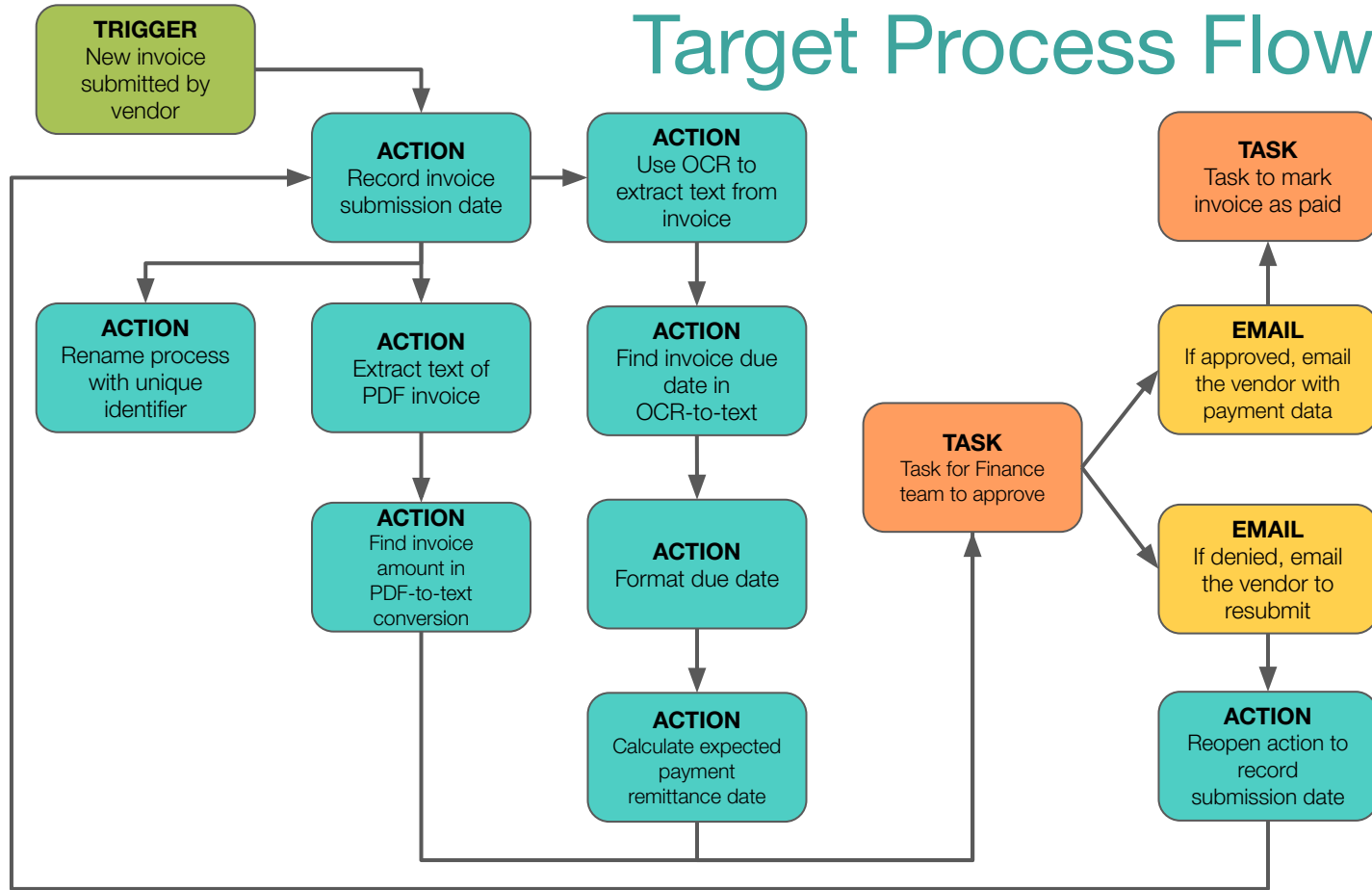
Once the invoice has been approved, notify the vendor of approval and the expected payment remittance date.

## ACTION TYPE

Email: Send an email

CONFIGURATION FIELD	DESCRIPTION	USE HERE
Email Body	The body of the email that will be sent to the recipient.	Hello,  The invoice you submitted to XYZ Corporation for payment (attached) has been approved. You can expect payment by {{payment-remittance-date}}. <div></div>
Attachments	Any attachments that should be included. Usually will be a file uploaded during a process.	{{invoice}} <div></div>
Output field prefix	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	invoice-approved-notice

# Target Process Flow



# Deep Dive: Deadlines

Actions that assign work, whether through a manual task or a webform to complete, provide the option to set a deadline. The deadline can be set:

- To a certain number of days or hours after the task was assigned
- To a specific date

Reminders are sent at different intervals depending on the deadline:

- If a task deadline is 24 hours or less, a single email reminder will be sent at the due date.
- If the deadline is more than 24 hours away, five (5) email notifications will be sent:
  - At 70% of the total amount of time
  - On the deadline
  - One (1) day after the deadline
  - Two (2) days after the deadline
  - Three (3) days after the deadline



# Action 13: “Task to mark invoice as paid”

## PURPOSE

Manual task to remind the Finance team to pay the invoice by the due date

## ACTION TYPE

Assign task to a person

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Assign to</i>	Whom the task will be assigned to: individual or a group. Can use a Catalytic username or an email address.	[Select yourself]
<i>Instructions</i>	Any instructions you might include for the task assignee. Can include field references in {{handlebars}}.	Please pay the invoice for {{organization}} by {{payment-remittance-date}}  {{invoice}} 
<i>Fields</i>	Any form fields you might need the assignee to complete. Works the same as fields in a webform or an emailed form.	
<i>Deadline</i>	The date by which the task must be completed.	{{payment-remittance-date}} 

DEPENDENCIES	COMPLETED: If approved, email the vendor with payment data
CONDITIONS	

# Not Yet Used: Action Timing

The timing parameter allows us to delay any action in a process either:

- By a certain amount of time
- Until a certain date

Some examples where this could be useful:

- In an employee onboarding process, set a delay of seven (7) days after the employee start date that surveys the employee on his/her first week
- In a sales process, allow the salesperson to select how long s/he would like to wait before receiving a task to follow up with a prospect
- In a year-end close process, set tasks to start on specific dates leading up to the end of the fiscal year

## YOUR TURN

Try adding a delay on your first *Email: Send a form* action.



# Let's Run It!

Check your dependencies. Submit a sample invoice found in the Process Builder folder via the webform you created. Use one of the below contacts:

NAME	COMPANY	INVOICE FILE
Brian Jones	ABC Incorporated	Sample_Invoice_ABC_Incorporated.pdf
Barb Williams	ABC Incorporated	Sample_Invoice_ABC_Incorporated.pdf
Dennis Smith	GHI Company	Sample_Invoice_GHI_Company.pdf
John Miller	GHI Company	Sample_Invoice_GHI_Company.pdf
Amy Wright	Sandbox et Cie	Sample_Invoice_Sandbox_et_Cie.pdf
Adam Davis	Sandbox et Cie	Sample_Invoice_Sandbox_et_Cie.pdf
Kim Johnston	Test Company, Inc.	Sample_Invoice_Test_Company_Inc.pdf
Kevin Brown	Test Company, Inc.	Sample_Invoice_Test_Company_Inc.pdf



Questions, errors?





# Testing and Troubleshooting

# Test Mode, Preview & Test

- **Test Mode** and **Preview & Test** allow us to test processes without worrying about how the test will impact other Catalytic users.
- You should test all of your process's paths to ensure they work as expected.
- In **Test Mode** you can:
  - Choose to send all emails and tasks in the test run to yourself, without having to change the email address configured in the action.
  - Complete and test web forms without needing to visit the active URL for the form.
  - Easily find test runs on the **Pushbot Details** page by looking for (TEST) written next to the run name.
- In the **Preview & Test** tab you can:
  - Run just one action to see if it returns what you would expect.
  - Confirm that your configuration is correct by checking for any errors returned. If an error message is returned, we can use that error to help diagnose a misconfiguration.


# Fix Tasks

- When an action is misconfigured, it will yield an error message when it runs. We call these error messages **Fix Tasks**, and they appear in your **Tasks** list.
- When a Fix Task is returned, you have two options to troubleshoot:
  1. **Retry task now** will try running the task again. If you retry the task and see the same error, there's likely a configuration issue.
    - a. If there is an issue with a field in the run, you can edit the content of that field by selecting **:** then **Fields**.
  2. **Skip task** will skip the task causing the error, and continue to the next.
    - a. If specific fields are captured in that task, this could affect the remainder of process steps.
    - b. The next action only starts if the dependency relationship is set to **completed or skipped**.

## TIP

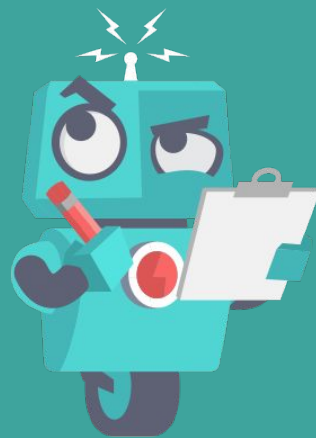
Use the error provided in the Fix Task. It will almost always tell you where to look for a mistake.

# Avoiding Common Building Pitfalls

ISSUE	SOLUTION
Missing a dependency or errors in dependencies	Always review your process in the Flowchart view to check the sequence.
Mistakes in {{handlebar}} usage	Use the handlebar picker  whenever possible to avoid mistakes. When unavailable, double check all of your field references have two { on both sides and that you've replaced all spaces and punctuation with -.
Missing a task condition	Use “if” in your action names as a reminder to include a condition.
Trouble with field names	Use easy-to-remember field name for quick reference later. Use all lowercase in field output names; get in the habit of <b>writing-like-this</b> .



Questions?



# Recap: What We've Learned So Far

- You have now covered the fundamentals of process building in Catalytic, and should understand the building blocks of a process:
  - Triggers
    - Template-Level Fields
  - Actions
    - Action-Specific Configurations
    - Action-Level Fields: Inputs and Outputs
  - Dependencies
  - Conditions

Catalytic actions covered:

- Date: Get current date
- Date: Format a date
- Date: Adjust a date
- PDF: Extract PDF text
- Images: Optical Character Recognition (OCR)
- Text: Find text next to other text
- Email: Send a form
- Email: Send an email
- Assign task to a person

**Questions?**



# Coming Next

## 1. Catalytic Data Tables

- a. What Is a Data Table?
- b. Data Table Configuration
- c. Process Interactions with Data Tables

## 2. Batch Processing

- a. Intro: Starting Processes for Each Row of a Table or Spreadsheet
- b. Referencing Fields in Batch Processes
- c. Building with Batch Processes

## 3. Predictive Model

- a. What Is a Predictive Model?
- b. Requirements
- c. Incorporating Predictive Models in Processes

# Session 3: Data Tables





# Introduction to Data Tables

# What Is a Data Table?

**Data tables are a programmable, dynamic database that can interact with processes. They:**

- Are structured in rows and columns, and their values can be used in processes
- Can be converted to spreadsheet, .csv, markdown, and other text formats for email communication, or for reporting and manipulation outside Catalytic
- Can serve as a record for human workflows (dates, statuses, etc)
- Are often used in conjunction with or alongside Excel

# Accessing Data Tables

- To access your team's data tables, select **Data** from the top menu.
- There are three types of data tables, and you can filter for each:
  - **Master:** Automatically created by creating a process template (read-only)
  - **Imported:** Uploaded from a spreadsheet or .csv (dynamic)
  - **Application:** Tables containing Catalytic metadata (read-only)
- Click into any of the tables to see the columns and data it stores. Once in a data table, you can click the **:** button at the top right to see edit and export options.

## YOUR TURN

Visit the **Data Tables** page and find the “Master Vendor List” table.

# Master Data Tables

- **Refresher:** When a process is started, either manually or with a trigger, that transaction creates a process run
- Each production run adds row to master data table
  - Runs started with the 'Test' button are not added
- Fields collected (from human forms), generated (automated actions) are each represented as a column in this table
- This table can be useful for reporting purposes

# Imported Data Tables

- From the **Data** section, you can upload .csv and .xlsx files as tables of record, or your own database
  - Consult *Pushbot System Limits* article for details on table limits
- Frequently useful where one column contains a unique identifier or key (for lookups)
- These tables can be copied as templates, and a table-type field used in your tasks to record variable unique records
- Rows can be added, removed, or updated as needed

# Application Data Tables

- There are currently two application-type data tables, both containing Catalytic metadata
- **Application Table: Users** contains user profile data
  - Can be useful to do a lookup for dynamic task assignment within process
- **Application Table: Process Templates** is useful for Catalytic administrators
  - Can be used to see who owns/has created which processes
  - Highlights which process templates are active or archived



# Data Tables Outputted by Automated Actions



# Example: Vendor Database

Imagine you have a table of vendors with which your company works. Each row of the table would be a different vendor, and each column an attribute of that vendor (Name, Address, Category, Pricing, Satisfaction rating).

You could leverage this table in several ways:

- In a *Vendor Approval process*, use the **Tables: Add a row** action to add a vendor to the table once approved.
- In a *Vendor Selection process*, use the **Tables: Apply filters** action to filter the table only to vendors offering specific products, or services.
- In a *Vendor Renewal process*, use the **Tables: Start a process for each row** action to initiate a renewal process for vendors after contract expiration.

# Common Data Table Use Cases

## Mapping Tables

Combining contents of two spreadsheets with different formats into one, with columns reordered.

*Example: Two spreadsheet exports from a CRM with the same content, but different formatting.*

## Batch Processing

Start a process for each row in a spreadsheet. This is used to perform bulk actions on an entire spreadsheet.

*Example: An email notification or survey to each contact in a spreadsheet.*

## Maintain a Database

Make periodic updates to a table of record housed in Catalytic. The reference table may be used in any Catalytic process.

*Example: Making annual updates on benefit elections to a database of all employees.*

# Catalytic Data Tables & Excel

- Most of the data you engage with day-to-day likely lives in Excel spreadsheets or comma-separated values (.csv) files
- Catalytic data tables allow one to automate work in those files, but you should:
  1. Convert them to data tables
  2. Perform automated actions on and using their data
  3. Convert back to an Excel file (if needed as your process output)

## QUESTION FOR DISCUSSION

Do you have any regular processes or work you complete in Excel right now? What Excel actions do you perform that we could automate in a data table?

# Converting Spreadsheets to Data Tables

A spreadsheet can be converted to a data table either manually or through an automated action.

- **Manual:** Use when the contents of the spreadsheet are mostly static.
  - Examples: An employee database, vendor list, table of product IDs, etc.
  - Upload spreadsheet directly on the **Data** page
- **Automated:** Use when the spreadsheet is regularly changed, and will be submitted to start a process.
  - Examples: A monthly budget spreadsheet, a quarterly pricing spreadsheet, etc.
  - Use *Excel: Save spreadsheet to table* to convert a spreadsheet as part of a process.
  - If needed, you can locate imported data table ID in process run by clicking **:** and then **Fields**. To review while building, use this URL format:  
[https://\[yourteamname\].pushbot.com/tables/\[yourtableid\]](https://[yourteamname].pushbot.com/tables/[yourtableid])

# Data Table Actions vs. Excel Actions

ACTION	DATA TABLE ACTION?	EXCEL ACTION?
Add rows and columns	✓	✗
Apply filters	✓	✓
Average and sum data in a column	✓	✗
Compare two tables	✓	✗
Convert table to text table	✓	✓
Count rows	✓	✗
Find similar text	✓	✗
Lookup and return data	✓	✓
Max and min data in a column	✓	✗
Remove columns and rows	✓	✗
Remove duplicates in a column	✓	✓
Update a row	✓	✗
Start a process for each row	✓	✓

# Creating a Blank Data Table

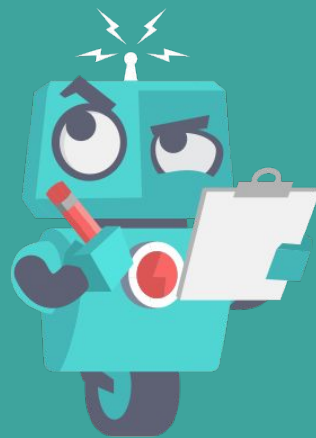
- Sometimes we create tables from an existing spreadsheet, but often we need to create a blank table that we'll populate through the course of a process
- Use the *Tables: Create an empty table* action with the following configuration:
  - *Column names* - Names for each column header
  - *Column types* - The type of data in the column (e.g. text, integer, date, file, etc.)
  - *Output field name* - The field name under which the data table can be referenced

## TIP

When building a process that will add to a blank table, make this the first action in your process so that you have it ready for later actions.



Questions?







# Configurations with Data Tables

# Target Process

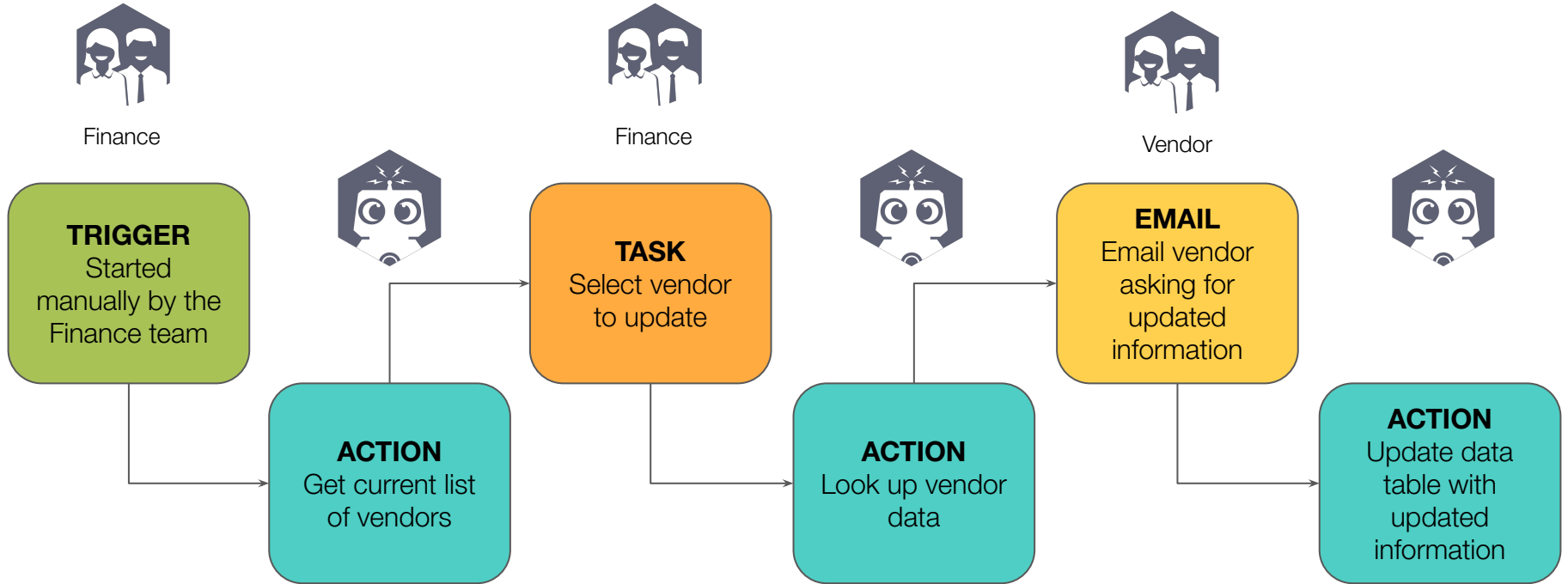
We're going to build a process to update a database of approved vendors. The process will:

1. Start manually
2. Retrieve a list of current vendors to populate a field
3. Contain a task to select the vendor to update
4. Look up the contact information for the vendor
5. Email the vendor asking for updated information
6. Update the row in the data table for that vendor

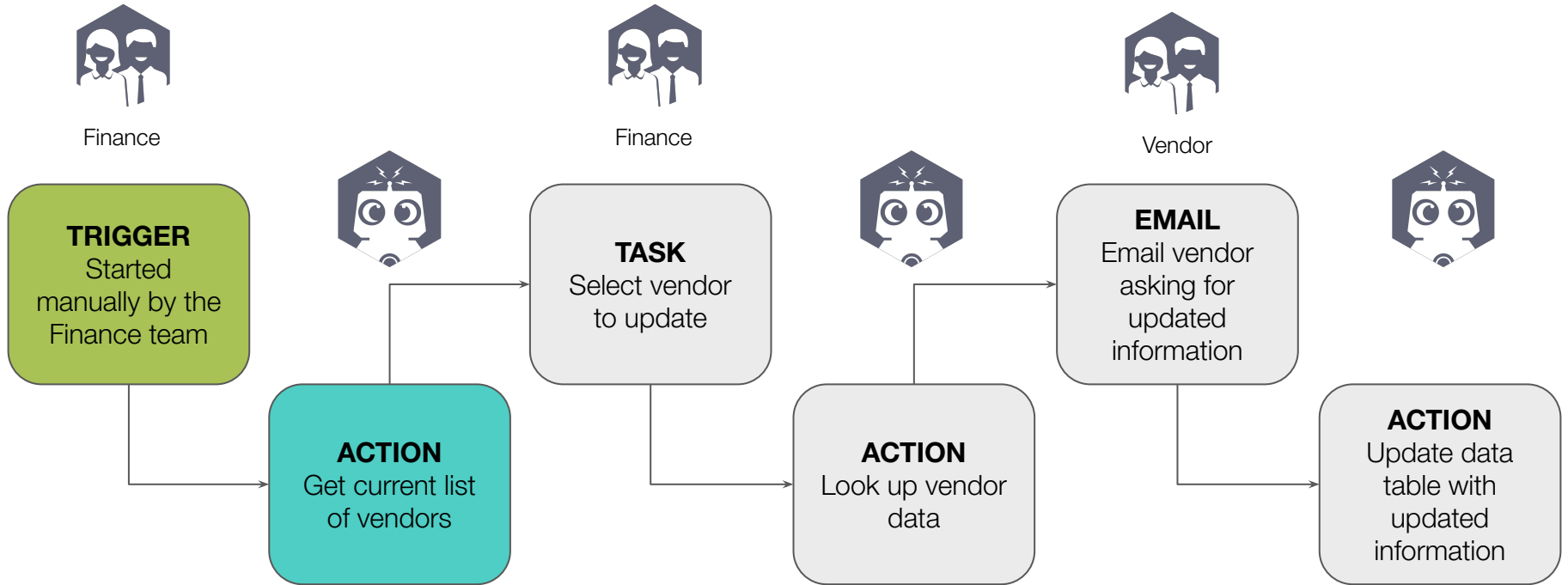
## **YOUR TURN**

Create a new process template in which you will build your version of the Vendor Update Process. You can name the template “\*Your Name\* Vendor Update Process.”

# Target Process Flow



# Target Process Flow



# Action 1: “Get current list of vendors”

## PURPOSE

Convert table’s Vendor Name column to text for a list of current vendors in a later step. Allows maintenance of table as source of truth for current vendors.

## ACTION TYPE

Tables: Convert data table to text

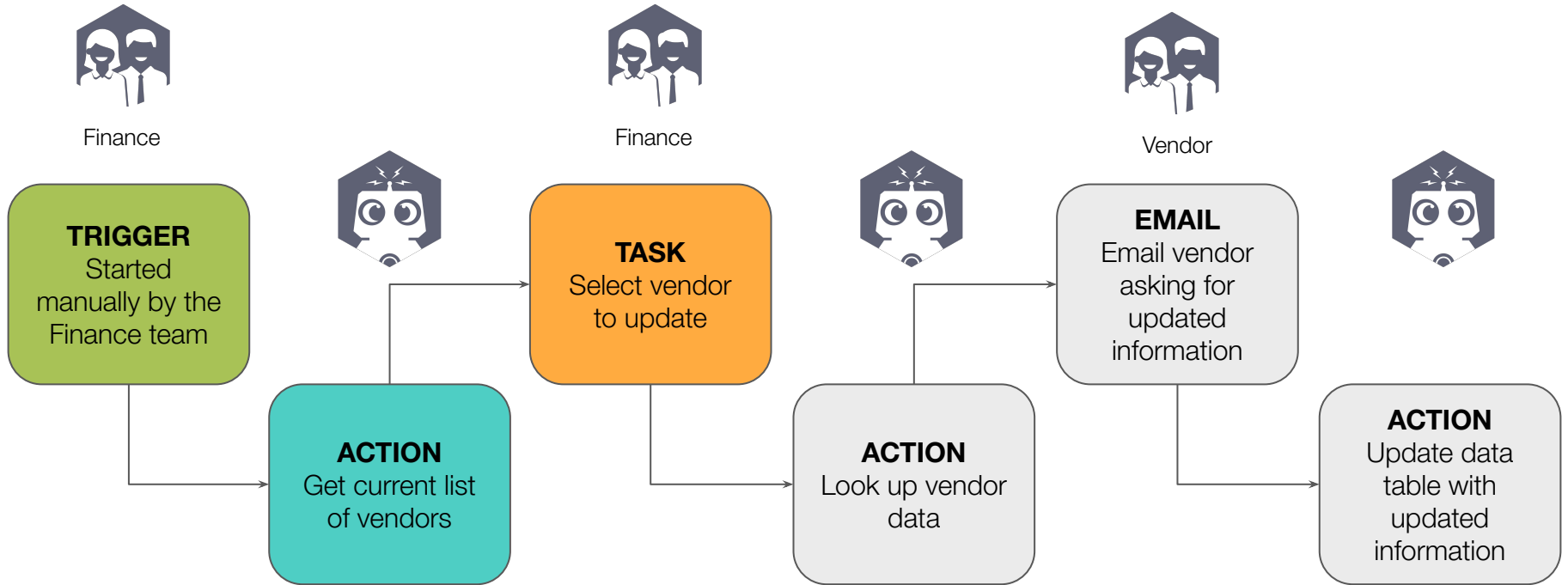
CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table</i>	We can find the ID of the table we want to look up from by going into the table and copying the alphanumeric code from the end of the URL.	Locate in imported tables in the <b>Data list</b> .
<i>Row template</i>	The column(s) to convert to text.	Columns[‘Vendor Name’]
<i>Row delimiter</i>	The delimiter you wish to use between entries. Defaults to new line.	
<i>Output field name</i>	The name of the text field we will create and reference later.	vendor-list

DEPENDENCIES	
CONDITIONS	

## YOUR TURN

After completing the action configuration, click the **Preview & Test** tab to run the action and see what is returned from the data table.

# Target Process Flow



# Action 2: “Select vendor to update”

## PURPOSE

Select vendor from current list to update the record, using text generated from previous step.

## ACTION TYPE

Assign task to a person

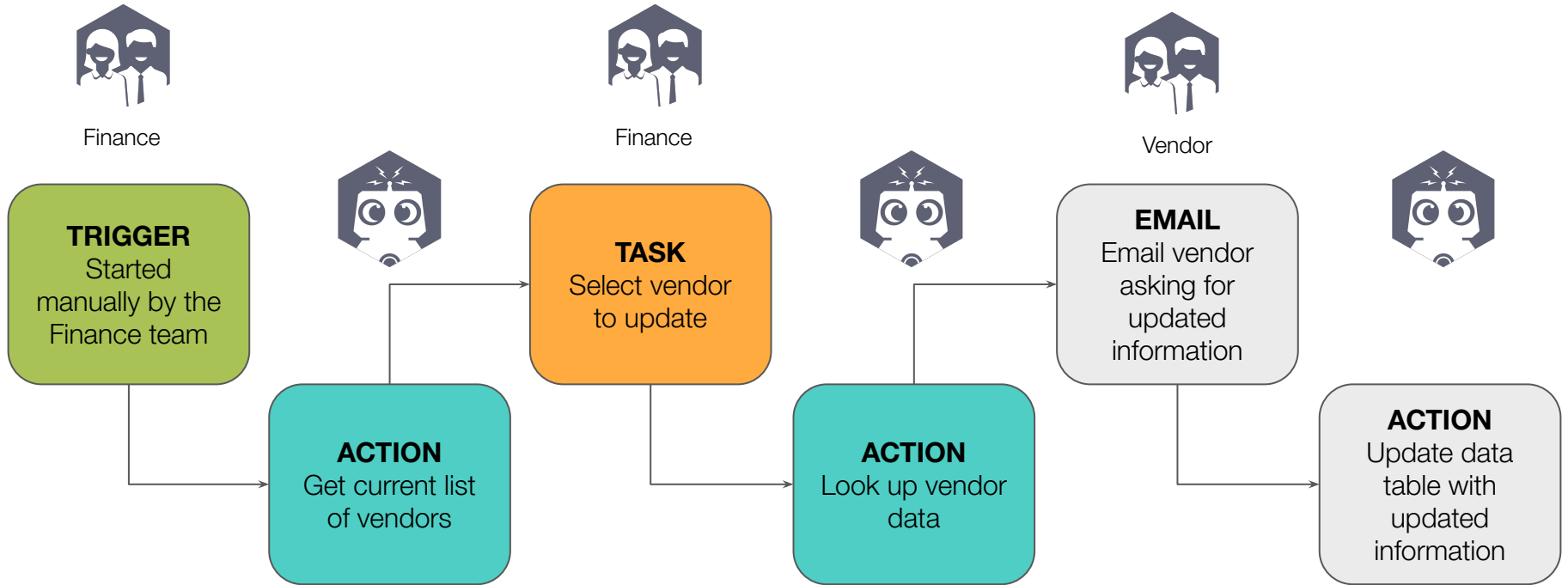
CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Instructions</i>	Any instructions you might include for the task assignee. Can include field references in {{handlebars}}.	Select the vendor you wish to update.
<i>Fields</i>	Any form fields you might need the assignee to complete. Works the same as fields in a webform or an emailed form.	<b>Name:</b> Vendor to Update <b>Type:</b> Choose One {{vendor-list}} <b>Required:</b> Yes

DEPENDENCIES	<b>COMPLETED:</b> Get current list of vendors
CONDITIONS	

### QUESTION FOR DISCUSSION

Why do we need to include the Vendor to Update field in the task, rather than as a template-level field?

# Target Process Flow





# Action 3: “Look up the vendor data”

## PURPOSE

Return data from a data table using a search term to find matching rows.

## ACTION TYPE

Tables: Look up data in a column

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table</i>	<p>We can find the ID of the table we want to look up from by going into the table and copying the alphanumeric code from the end of the URL. Example:</p> <p><a href="https://certification.pushbot.com/tables/55d87868-10a2-4ace-b762-0c017d8e5fee">https://certification.pushbot.com/tables/55d87868-10a2-4ace-b762-0c017d8e5fee</a></p>	<p>Locate in imported tables in the <b>Data</b> list.</p>
<i>Lookup column name</i>	<p>The column in which you want to search for matches. All other columns from the matching row will be returned.</p>	Vendor Name

DEPENDENCIES	COMPLETED: Select vendor to update
CONDITIONS	


# Action 3: “Look up the vendor data”

## PURPOSE

Return data from a data table using a search term to find matching rows.

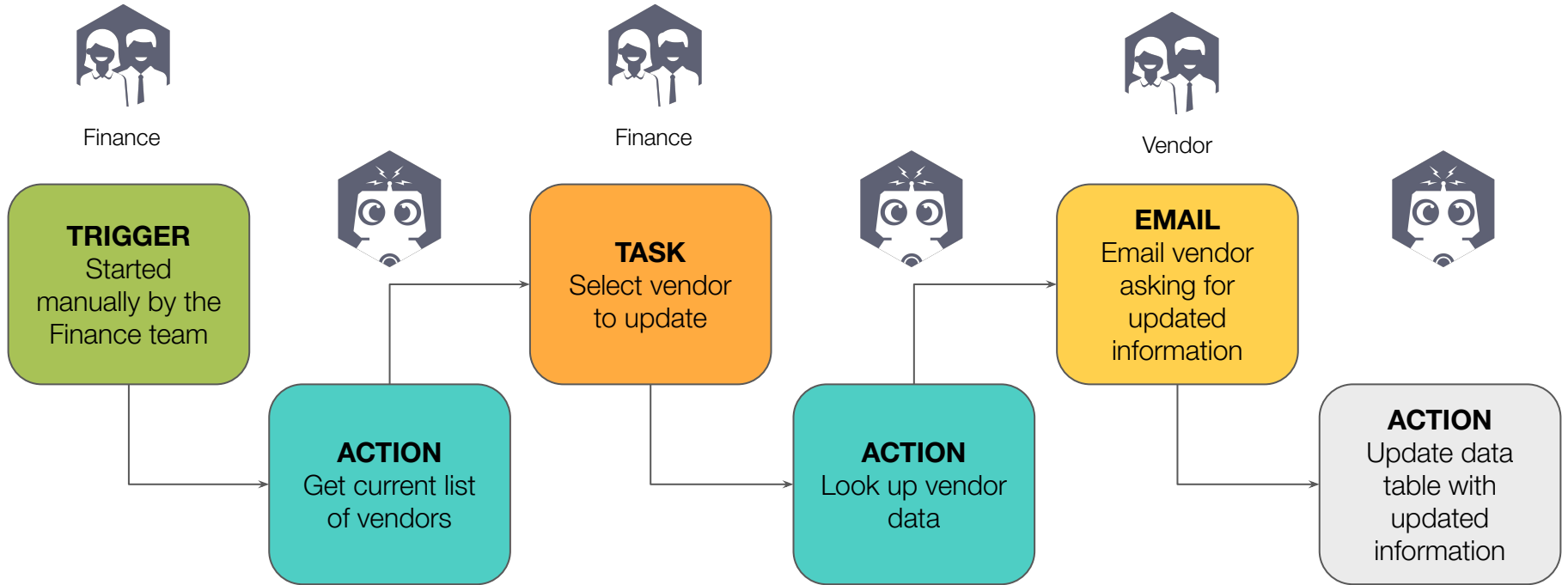
## ACTION TYPE

Tables: Look up data in a column

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Search term</i>	The search term that will be searched for in the lookup column. All other columns from the matching row will be returned.	{{vendor-to-update}} 
<i>Columns to return</i>	Option to designate only certain columns to return. Will default to all columns.	
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	vendor

DEPENDENCIES	COMPLETED: Select vendor to update
CONDITIONS	

# Target Process Flow




# Action 4: “Email vendor for updated information”

## PURPOSE

Email the vendor asking them to confirm or update contact information

## ACTION TYPE

Email: Send a form

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>To address</i>	The address to which the email will be sent. Can include an email previously collected in {{handlebars}}.	{{vendor--contact-email-address}} 
<i>CC</i>	Any email addresses to be cc'd on the email	
<i>Sender name</i>	Who the email will appear to be from in the recipient's inbox	ABC Corp
<i>Email subject</i>	The subject of the email that will be sent to the recipient from Catalytic	Please Provide Updated Information

DEPENDENCIES	COMPLETED: Look up the vendor data
CONDITIONS	

# Action 4: “Email vendor for updated information”

## PURPOSE

Email the vendor asking them to confirm or update contact information

## ACTION TYPE

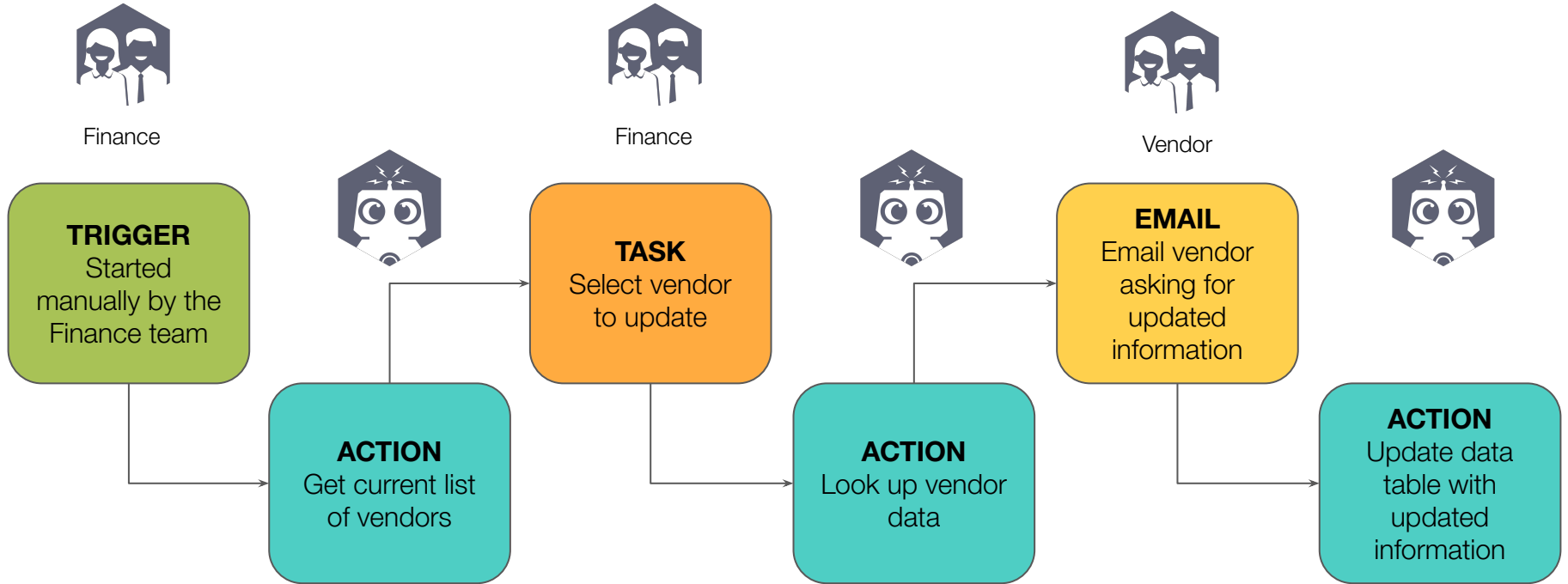
Email: Send a form

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Email body</i>	The body of the email that will be sent to the recipient	
<i>Form title</i>	The heading that will appear at the top of the form sent to the email recipient	Update Information
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	vendor-update

### Add the following required fields:

NAME	TYPE	DEFAULT
<i>Phone Number</i>	Text	{{vendor--contact-phone-number}}
<i>Street Address</i>	Text	{{vendor--street-address}}
<i>City</i>	Text	{{vendor--city}}
<i>State</i>	Text	{{vendor--state}}

# Target Process Flow




# Action 5: “Update data table with updated information”

**PURPOSE**

Update a record in the data table with the new information submitted

**ACTION TYPE**

Tables: Update a row

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table</i>	The “ID” of the data table we want to add to.	<i>Locate in imported tables in the <b>Data</b> list.</i>
<i>Look up column names</i>	To update a specific record/row, we need to locate the record we want to update. This is the column in which we will be searching for a given search term to locate the record.	Vendor Name
<i>Look up column criteria</i>	The search term we’ll use to locate the record we want to update.	{{vendor-to-update}} 
<i>Output field name</i>	Catalytic automatically saves to a field the number of rows that are updated. This is the name of that field.	number-updated
<b>DEPENDENCIES</b>		<b>COMPLETED:</b> Email the vendor for updated information
<b>CONDITIONS</b>		


# Action 5: “Update data table with updated information”

## PURPOSE

Update a record in the data table with the new information submitted

## ACTION TYPE

Tables: Update a row

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Columns</i>	The columns for which we will be updating data. You do not need to include all columns in the table.	Contact Phone Number, Street Address, City, State
<i>Values</i>	Each value corresponds with a column you listed above. Add in a comma-delimited list. Example: <ul style="list-style-type: none"><li>- Columns: First Name, Last Name</li><li>- Values: John, Smith</li></ul>	{{phone-number}}, {{street-address}}, {{city}}, {{state}} 
<i>Update all</i>	This will determine whether you want to update every row or just the first matching row.	

## TIP

Column names must match those found in the table. Always double check spelling and special characters.



# Let's Run It!

Remember to check your dependencies!

Click **Start** or **Test** on the process template to start it.



Questions?



# Session 4: Batch Processing



# Recap of Data Tables

- Data tables allow one to structure data in rows and columns, and to use those data in actions within automations
- Data tables are created in one of the following ways:
  1. Automatically through the creation of a process template
    - a. Every process template automatically stores process run data in master table
  2. Through actions that save spreadsheets as data tables
  3. Through an action that creates a blank data table
  4. By manually uploading a spreadsheet and saving it as a data table
- Catalytic data tables are often used in conjunction with or alongside Excel.

One of the most common uses of data tables is **batch processing** of data.

# Dividing Processes

- It can be helpful to split a process into more than one Catalytic process template (each called a *subprocess*).
- We might divide a process if:
  - The process can be clearly split into discernible sections
  - Different sections have their own outputs (that may be used for the final output)
  - The long-term maintenance of the process would be facilitated by several smaller process templates, instead of one very large process template
  - Making process improvements or updates would be easier
- Adhere to a naming convention for your suite of processes:
  - Employee Reporting 1 Starter
  - Employee Reporting 2 Calculate Tenure

# Subprocesses

- Occasionally business processes must be split into multiple *subprocesses*.
- We use subprocesses in cases where:
  - We want to perform a sequence of actions on each row of data in a spreadsheet
    - e.g. send an email notification / form, add responses to table
  - We want to break a complex process into distinguishable sections for easier maintenance
- A subprocess is started during the course of a *starter process*.
- The starter process triggers the subprocess using any of the following actions:
  - Tables: Start Pushbot for each row
  - Excel: Start a process for each row
  - Pushbot: Start another Pushbot

# Introduction to Batch Processing

Using the *Tables: Start Pushbot for each row* action, **batch processing** allows us to perform actions in a process on each individual row of data in a data table. When we batch process a table, we start a run of a process template for each row in the table.

Advantages of batch processing:

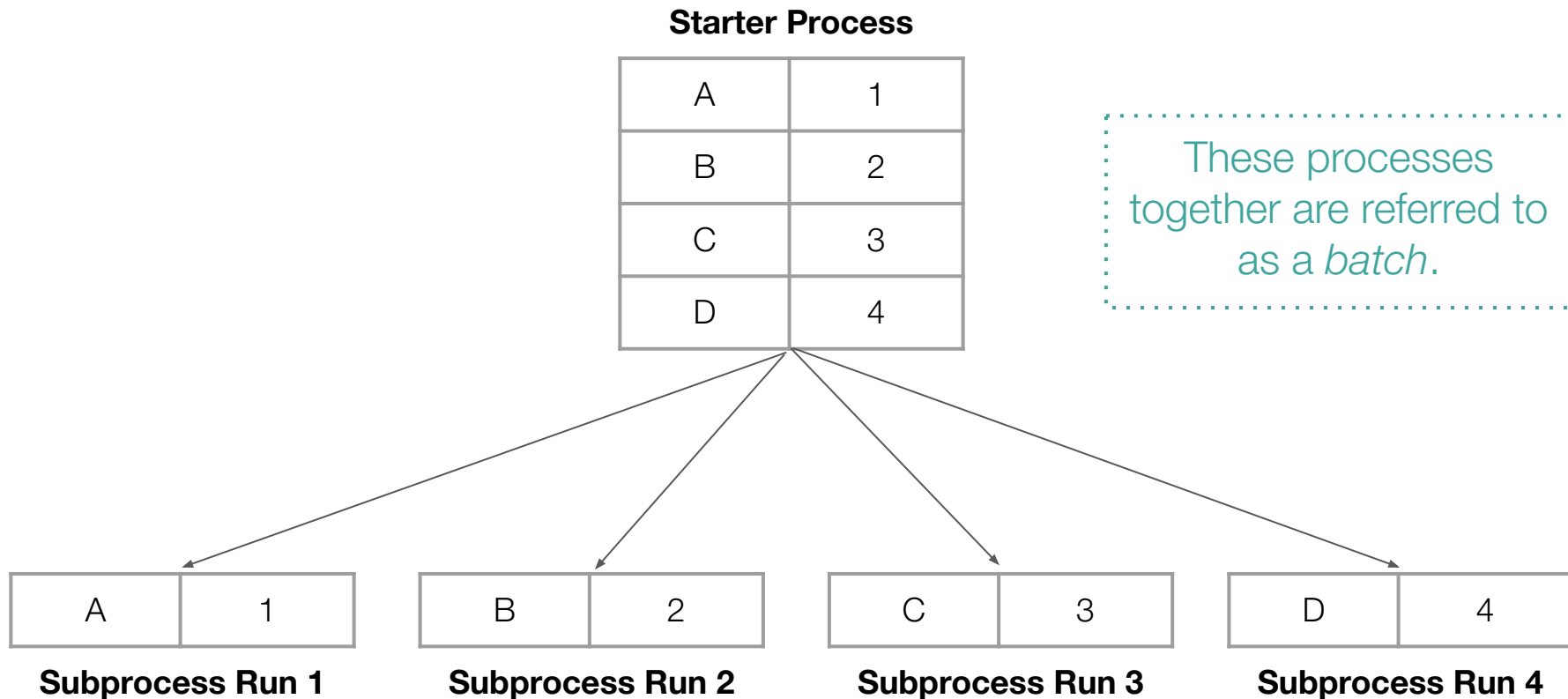
- Requires a single process template to perform actions on thousands of rows
- Each row's column values can be referenced as `{{fields}}` to make runs unique, and dynamic

# Introduction to Batch Processing

- To create a batch process, we must build at least two process templates:
  - 1. Starter Process:** The “starter” (or “parent”) is the initial process that is usually made up of only one or two actions, including *Tables: Start Pushbot for each row*. It is from this process template that the batch process is triggered.
  - 2. Subprocess:** A subprocess (or “child”) is where most of the work takes place. In the subprocess, we sequence the actions that we want to perform on the row data that flows down from the starter.



# Introduction to Batch Processing



# Fields in Batch Processing

- When a subprocess is started as part of a batch, it inherits fields from the starter process.
  - Any fields created in the starter process are passed to the subprocess
  - These fields can be referenced in `{{handlebars}}` in actions in the subprocess
  - **Note:** Fields from a starter process will not appear in the subprocess, and must be referenced manually using `{{handlebars}}`
- Because each subprocess run corresponds to a single row in the table, Catalytic also automatically saves each cell of that row to a field by the cell's *column name*

Example:

First Name	Last Name	Date of Birth
John	Smith	1/1/1970

`{{first-name}}` = John

`{{last-name}}` = Smith

`{{date-of-birth}}` = 1/1/1970

# Examples of Batch Processes

1. Sending an email to every individual listed in a spreadsheet. Often used for reminders on overdue invoices, notifications of noncompliance, etc.
2. Combining multiple spreadsheets into one data table by adding each row individually to the combined data table. Often used when mapping to a different output format/order.

# Tables: Apply Formulas to Columns

- *Tables: Apply formulas to columns* can be used to set the value of one or more columns for every row in a data table.
- Requires knowledge of basic Javascript syntax.
- If  $A + B = C$  in the example below, we can fill with a single action, rather than needing to use a batch for each row.

<b>A</b> Integer	<b>B</b> Integer	<b>C</b> Integer
1	2	
2	4	
3	6	

- **Syntax:** `columns['C'] = columns['A'] + columns['B']`

# Tables: Apply Formulas to Columns

- *Tables: Apply formulas to columns* can be used to set the value of one or more columns for every row in a data table.
- Requires knowledge of basic Javascript syntax.
- If  $A + B = C$  in the example below, we can fill with a single action, rather than needing to use a batch for each row.

A Integer	B Integer	C Integer
1	2	3
2	4	6
3	6	9

- **Syntax:** `columns['C'] = columns['A'] + columns['B']`



# Sample Process: Email Reminder Batch Process

# Target Process

We're going to build a process to remind business owners about past due invoices. It will:

1. Start via email trigger when the spreadsheet of past due invoices is sent as an attachment.
2. Start a subprocess for each *individual* listed in the spreadsheet. Individuals may be listed more than once if they have more than one overdue invoice.
3. In the subprocess, create a table of just that individual's past due invoices.
4. Send each business owner an email notification with the table of his/her past due invoices and a reminder to finalize them.

## YOUR TURN

Create a new process template in which you will build your version of the Past Due Invoices process. You can name the template “\*Your Name\* Past Due Invoices 1 Starter”.

# Your Turn: Mapping the Process

## Goals:

1. The starter process is triggered automatically when the weekly list of past due invoices is submitted.
2. Each business owner listed in the spreadsheet receives only one email (Hint: We only want to start one subprocess run per business owner).
3. The body of the email contains a table of the business owner's past due invoices, meaning that a text format must be used.
4. After all subprocesses are completed, the Finance team receives an email with a count of emails sent (i.e. the number of business owners with past due invoices).



# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner

# Create the Email Trigger

1. To create the email trigger linked to starter process, click **Edit** and then **Add a Trigger** on the **Pushbot Settings** page.
2. Select the **Email** trigger type.
3. Complete the required configuration fields:

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Email address</i>	Whenever an email reaches this email address, the process associated with this trigger will start.	*yourname*-past-due-invoices
<i>This trigger will start</i>	Select the process you'd like to start when this email address receives an email. [*May not display, depending on path to trigger page.]	*Your Name* Past Due Invoices 1 Starter

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner


# Starter Action 1: “Convert spreadsheet to data table”

## PURPOSE

Convert the spreadsheet submitted via email into a data table.

## ACTION TYPE

Excel: Save spreadsheet to table

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Initial file</i>	The file to be saved as a table. This will often be referenced in {{handlebars}} after being submitted into a field earlier in the process.	{{trigger-attachment}} 
<i>Column headers row</i>	The row in which the headers are found. This is usually Row 1.	1
<i>Output field name</i>	The name of the field where the table ID will be saved for reference later in the process.	past-due-invoices

DEPENDENCIES	-
CONDITIONS	-

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner

# Starter Action 2: “Remove duplicate business owner rows”

**PURPOSE**

End up with a list of all business owners without any duplicates. This will be the table we will batch start from.

**ACTION TYPE**

Tables: Get unique rows

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table ID</i>	The “ID” of the data table we want to remove duplicates from.	{{past-due-invoices}} 
<i>Column name</i>	The column name from which you want to remove duplicates.	Business Owner Name
<i>Output field name</i>	The field name to which you want to save the new table with no duplicate business owner names.	no-duplicate-business-owners

DEPENDENCIES	COMPLETED: Convert spreadsheet to data table
CONDITIONS	

**TIP**

Whenever we do something to an entire table (i.e. remove duplicates, filter, etc.), Catalytic saves the result to a *new* data table. That way, we still have access to the original if we need it.

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner




# Starter Action 3: “Count number of business owners”

**PURPOSE**

Count number of emails that will be sent to report back to the Finance team.

**ACTION TYPE**

Tables: Count rows in a table

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table ID</i>	The “ID” of the data table we want to count the number of rows.	{{no-duplicate-business-owners}} 
<i>Name the output of this action</i>	The field name to which you want to save the number of rows counted.	number-of-emails

DEPENDENCIES	COMPLETED: Remove duplicate business owners rows
CONDITIONS	

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner


# Starter Action 4: “Start subprocess for each business owner”

## PURPOSE

Start the email notification subprocess for each business owner listed in the “No Duplicates” spreadsheet.

## ACTION TYPE

Tables: Start Pushbot for each row

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Process</i>	The “ID” of the process template we want to start for each row. We find the ID of the process template the same way was the Table ID: all the alphanumeric code after the / in the URL.	
<i>Data table</i>	The “ID” of the data table we want to start the process for each row of. Can reference a table saved to a field name in a previous action.	{{no-duplicate-business-owners}} 
<i>Run name</i>	The name each subprocess run will be assigned. Can use field values in {{handlebars}} to make this unique.	Past invoices for column['Business Owner Name']

DEPENDENCIES	COMPLETED: Count number of business owners
CONDITIONS	

# Starter Action 4: “Start subprocess for each business owner”

## PURPOSE

Start the email notification subprocess for each business owner listed in the “No Duplicates” spreadsheet.

## ACTION TYPE

Tables: Start Pushbot for each row

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Deadline</i>	You can use this configuration to pass on a a deadline to the subprocess	FALSE
<i>Complete immediately</i>	Determine whether you want the action to be marked complete and start any dependent actions immediately or wait until all actions in the subprocess have been completed. In most cases you should wait to start subsequent actions until the batch has completed.	FALSE
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	invoice-sent

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner

# Target Process

We now need to create a subprocess template to email each manager.

## YOUR TURN

Go to the **All Pushbots** list and create a new subprocess template. You can name the template “\*Your Name\* Past Due Invoices 2 Email Manager”.

# Subprocess Action 1: “Filter data table by business owner”

## PURPOSE

Filter the full dataset down to just the business owner to get a table of just that business owner’s invoices.

## ACTION TYPE

Tables: Apply filters

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data table ID</i>	The “ID” of the data table we want to filter	{{past-due-invoices}}
<i>1st column to filter</i>	The name of the column to be filtered. Use the exact name including all capitalizations and special characters.	Business Owner Name
<i>1st column filter criteria</i>	The term we want to use to filter by. Using one filter term will look for exact matches. <i>If you use the Advanced filter logic configuration, this filter will be ignored.</i>	{{business-owner-name}}

DEPENDENCIES	-
CONDITIONS	-

# Subprocess Action 1: “Filter data table by business owner”

## PURPOSE

Filter the full dataset down to just the business owner to get a table of just that business owner’s invoices.

## ACTION TYPE

Tables: Apply filters

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>2nd column to filter</i>	The name of the 2nd column to be filtered. Use the exact name including all capitalizations and special characters. *not required*	
<i>2nd column filter criteria</i>	The term we want to use to filter the 2nd column by. Using one filter term will look for exact matches. <i>If you use the Advanced filter logic configuration, this filter will be ignored.</i>	
<i>Advanced filter logic</i>	This field will be needed if you need to use a comparison operator (>, <, !=, ==), the in platform description does a great job of explaining everything you can do.	
<i>Output field prefix</i>	The field name to which we will save the filtered table.	filtered-by-business-owner



# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner


# Subprocess Action 2: “Convert business owner’s table to markdown”

## PURPOSE

Create a table to be embedded in the body of the email notification.

## ACTION TYPE

Tables: Convert data table to markdown text

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Table ID</i>	The “ID” of the data table we want to convert to markdown table.	{{filtered-by-business-owner}} 
<i>Columns</i>	The columns from the data table to be included in the markdown table. Leave blank to include all columns.	Customer Number, Customer Name, Invoice Number, Invoice Date, Invoice Amount, PO Number
<i>Bold header row</i>	True/False on whether or not to bold the header row.	TRUE
<i>Output field name</i>	The field name to which we will save the markdown table.	table

DEPENDENCIES	COMPLETED: Filter data table by business owner
CONDITIONS	

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner



# Subprocess Action 3: “Email business owner with invoices”

## PURPOSE

Send an email to the business owner with a table of their invoices in the body of the email.

## ACTION TYPE

Email: Send an email

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>To address</i>	The recipient of the email.	{{business-owner-email}} 
<i>Email subject</i>	The subject of the email being sent to the business owner.	Please Finalize Invoices
<i>Email Body</i>	The body of the email being sent to the business owner.	Your past due invoices are listed in the table below. Please take immediate action to finalize these invoices.  {{table}} 
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	business-owner

DEPENDENCIES	COMPLETED: Convert business owner’s table to markdown
CONDITIONS	

# Target Process Flow

## Starter Process

**TRIGGER**  
Started via  
email trigger

**ACTION**  
Convert  
spreadsheet  
to data table

**ACTION**  
Remove  
duplicate  
business  
owners

**ACTION**  
Count  
number of  
business  
owners

**ACTION**  
Start subprocess  
for each business  
owner in the no  
duplicates table

**ACTION**  
Email Finance  
team to confirm  
batch has  
completed



## Subprocess

**ACTION**  
Filter table by  
business  
owner

**ACTION**  
Convert  
filtered table  
to markdown

**EMAIL**  
Send business  
owner email  
with table  
embedded



Business  
Owner

# Starter Action 5: “Email finance team confirmation of batch completion”

**PURPOSE**

Email the finance team confirming the number of business owners who received emails.

**ACTION TYPE**

Email: Send an email


CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>To address</i>	The recipient of the email.	<i>Use your address here.</i>
<i>Email subject</i>	The subject of the email being sent to the business owner.	Past Due Invoices Emails Sent
<i>Email Body</i>	The body of the email being sent to the business owner.	This is to confirm that {{number-of-emails}} emails were sent to managers for past due invoices.
<i>Output field prefix</i>	Action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	batch-complete

DEPENDENCIES	COMPLETED: Start subprocess for each business owner
CONDITIONS	

# Starter Action 4: “Start subprocess for each business owner”



Don’t forget! When we built our starter process, we left the “Pushbot” field blank because we didn’t have that ID yet. Let’s go back and fill it in now.

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Process</i>	The “ID” of the process template we want to start for each row. We find the ID of the process template the same way was the Table ID: all the alphanumeric code after the / in the URL.	
<i>Data table</i>	The “ID” of the data table we want to start the process for each row of. Can reference a table saved to a field name in a previous action.	{{no-duplicate-business-owners}} 
<i>Started Pushbots name</i>	The name each subprocess run will be assigned. Can use field values in {{handlebars}} to make this unique.	column[‘Business Owner Name’]’s Past Due Invoices

# Let's Run It!

Remember to check your dependencies!

Send an email to the trigger email address you created with the sample data attached to kick off the process.



# Batch Processing: Tips and Tricks

- Wait until batch completes to proceed to subsequent actions
- Use a small, non-production dataset for testing
  - Do not test an incomplete process with thousands of rows of data
- Create a standalone process with a single action:
  - **Action:** Tables: Remove a row
  - Configure as seen on right, to remove data from your tables as you test.
  - Use Test & Preview function

Remove table row > Tables: Remove a row



Tables: Remove a row

Search columns in a data table for matching data and delete matching rows

Table ID — {{id}}

required

Look up column names — {{column}}

required

Look up column criteria — {{value}}

required

Delete all — true

Output field name — rows-removed

required



Questions?



# Session 5: Predictive Model





# Introduction to Predictive Model

# Predictive Model: What Is it?

A predictive model is a machine learning model based on a Catalytic data table. It has two important aspects:

- A **predicted field** defines the field that the predictive model will *attempt to predict* based on the overall data in the data table.
- The **relevant fields** define which data should be *used to determine the predicted field*. By default, all are included.
  - Consider omitting fields without impact, or to remove biases.

**Predictive models can be used to provide a recommendation to a person, send a default response, or to completely automate decision making if your confidence threshold is met (via Catalytic conditions).**

# Predictive Model: Why Use One?

- Many processes have key decision points that require an employee or manager to make a judgment or prediction based on input variables.
- Many have low stakes and do not justify time investment; they can also create bottlenecks
- Some decisions have a large number of variables
- Organizations spend a significant time and money training employees to make decisions, still find high error rates

# Predictive Model: Examples

## **Examples of how predictive models could be used in processes:**

- Determine if a company expense should be automatically approved
- Predict likelihood of a sales opportunity to close within next quarter
- Categorize and route an email based on text in the subject and body
- Recommend compensation bonus based on employee performance
- Suggest the best leader for a project based on project criteria

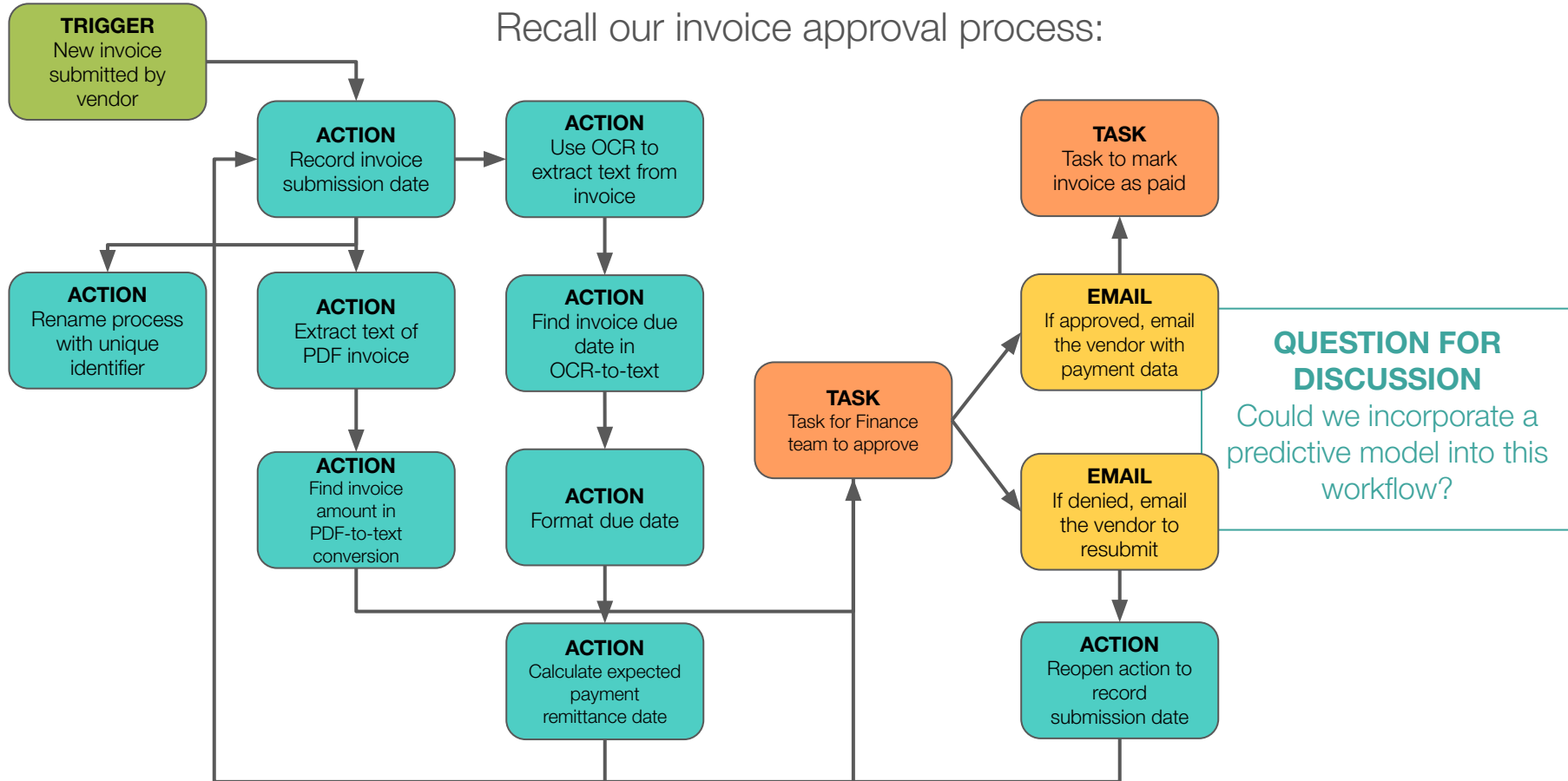
# Predictive Model: Data Requirements

- Ideal: **several thousand examples** (rows) of data to train your model.
  - If heavily reliant upon natural language processing, **tens of thousands of examples** is preferred.
  - If the prediction is straightforward, a **few hundred examples may suffice**, but such processes are rare.
- Both **master** and **imported** data tables can be used for predictions
- If you use a template's **master** table (containing process run data), subsequent process instances will continually train your model
- If you use an imported data table, leverage a **Tables: Add a row** step to continually train your model.



# Predictive Model: Applied

Recall our invoice approval process:



# Predictive Model: Applied

Consider if an individual or particular vendor regularly submitted invoices that contained errors and/or were rejected.

**Sample Invoice Approval Table**

...

Name Text	Organization Text	Submission Date Text	Invoice Amount Decimal	Approved First Round Text
Brian Jones <input checked="" type="checkbox"/> Open	ABC Incorporated	2019-01-01	2018.24	Approved
Dennis Smith	XYZ Company	2019-01-02	756.34	Approved
Kim Johnston	Test Company, Inc.	2019-01-03	54.14	Approved
Amy Wright	Sandbox et Cie	2019-01-04	8009.1	Denied
Brian Jones	ABC Incorporated	2019-01-05	1	Approved
Dennis Smith	XYZ Company	2019-01-06	6024.22	Denied
Kim Johnston	Test Company, Inc.	2019-01-07	1754	Approved
Amy Wright	Sandbox et Cie	2019-01-08	500	Approved

# Predictive Model: Applied

We could leverage historical data, specifically individual and organization name, and invoice amount.

**Sample Invoice Approval Table**

...

Name Text	Organization Text	Submission Date Text	Invoice Amount Decimal	Approved First Round Text
Brian Jones <a href="#">Open</a>	ABC Incorporated	2019-01-01	2018.24	Approved
Dennis Smith	XYZ Company	2019-01-02	756.34	Approved
Kim Johnston	Test Company, Inc.	2019-01-03	54.14	Approved
Amy Wright	Sandbox et Cie	2019-01-04	8009.1	Denied
Brian Jones	ABC Incorporated	2019-01-05	1	Approved
Dennis Smith	XYZ Company	2019-01-06	6024.22	Denied
Kim Johnston	Test Company, Inc.	2019-01-07	1754	Approved
Amy Wright	Sandbox et Cie	2019-01-08	500	Approved

# Predictive Model: Applied

To make a prediction as to whether invoice will be approved on initial submission, or if it requires additional scrutiny by the finance team.

**Sample Invoice Approval Table**

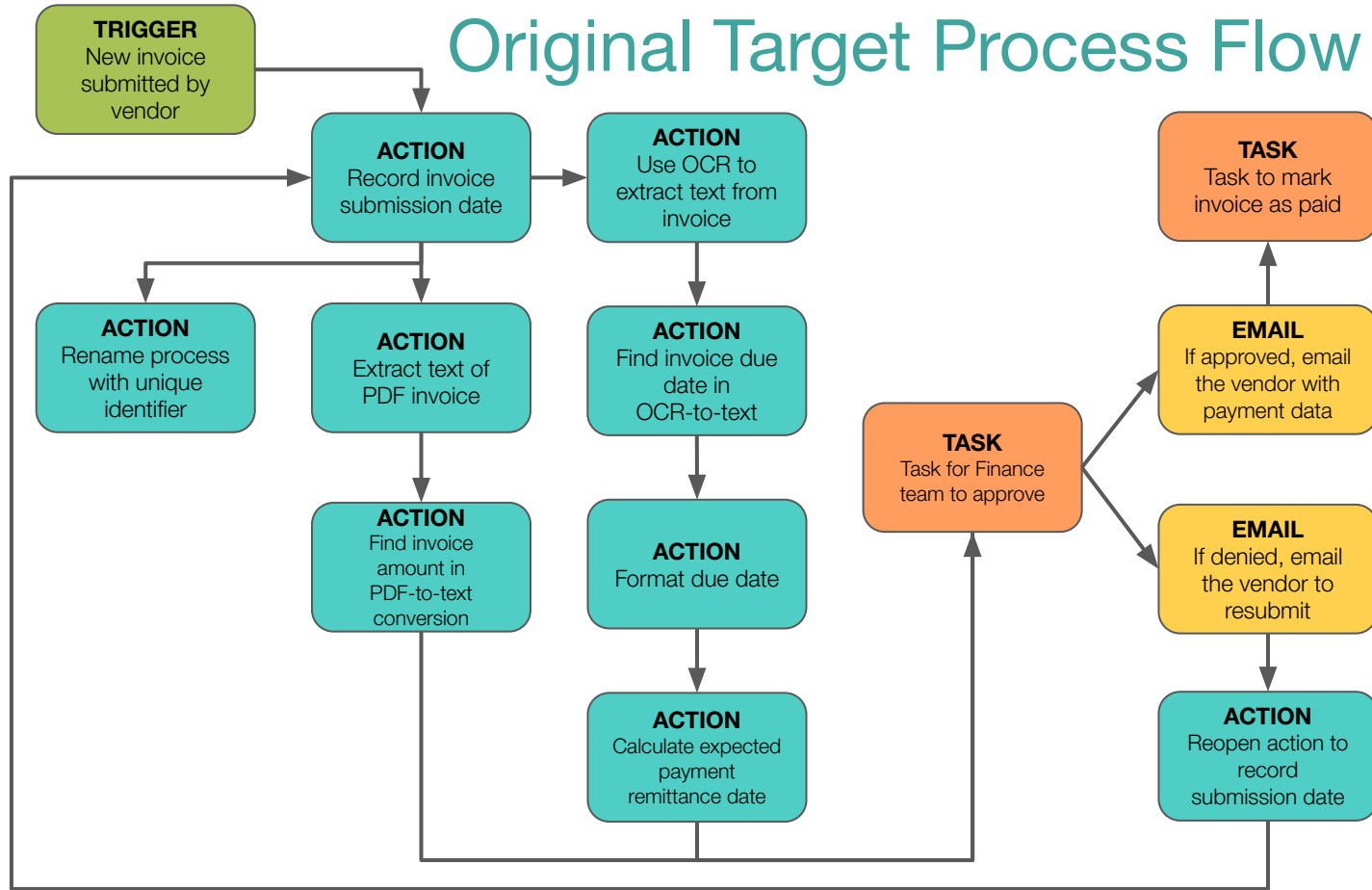
...

Name Text	Organization Text	Submission Date Text	Invoice Amount Decimal	Approved First Round Text
Brian Jones <a href="#">✓ Open</a>	ABC Incorporated	2019-01-01	2018.24	Approved
Dennis Smith	XYZ Company	2019-01-02	756.34	Approved
Kim Johnston	Test Company, Inc.	2019-01-03	54.14	Approved
Amy Wright	Sandbox et Cie	2019-01-04	8009.1	Denied
Brian Jones	ABC Incorporated	2019-01-05	1	Approved
Dennis Smith	XYZ Company	2019-01-06	6024.22	Denied
Kim Johnston	Test Company, Inc.	2019-01-07	1754	Approved
Amy Wright	Sandbox et Cie	2019-01-08	500	Approved

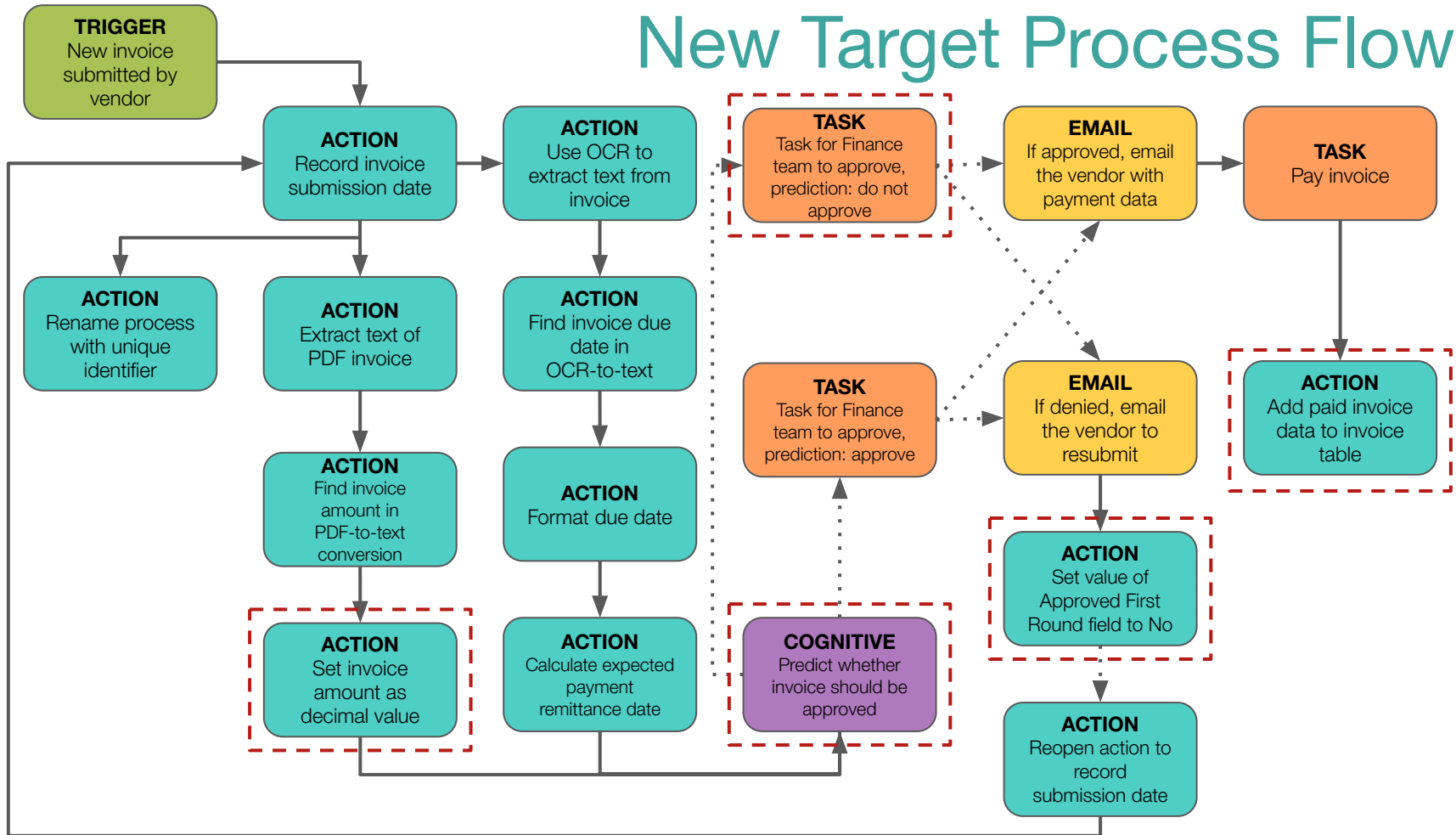


# Predictive Model: Implementing

# Original Target Process Flow



# New Target Process Flow



# Predictive Model: Creation

## Create a Predictive Model

After creating a model, you will be able to add an action to your processes to automatically predict or recommend a value based on historical data. Predictions will become more accurate over time as the model is trained on more data.

What would you like to name your model? \*

Invoice Approval Model

What data would you like to analyze? \*

Select the data table to use to train your model.

Sample Invoice Approval Table

Which field would you like to target for prediction? \*

Select the field for which you would like to automatically predict or recommend the value.

Approved First Round

What fields should be included in the analysis? \*

Include only those fields that should be considered as factors in the prediction. Exclude fields that will not yet have values at the time that the prediction will be made in the process.

Name Organization Invoice Amount

Cancel

Create Model

## YOUR TURN

1. Navigate to the **All Pushbots** list >> select [your] **Invoice Approval Process**.
2. Select **Edit**, then **Add More** under **Add-ons**; then select **Predictive Model**.
3. Click **Create Predictive Model**.
4. Configure as on left, click **Create Model**; you'll note that your model is being trained.

## FOR DISCUSSION

If you'd like to start from scratch, try importing Sample Invoice Approval Table.xlsx as a new table, and use that for your model.



# Predictive Model: Creation

Once trained, your model will indicate an accuracy rating.

## Training Model In Progress

The predictive model is currently being trained. It may take several minutes or even a few hours, so feel free to close this message. You can go ahead and add the action, "Field: Make a prediction," now.



**Current task:** Preparing data for analysis

**Estimated time remaining:** 15 - 90 minutes

## Predictive Models Completed



**Approve Invoice Prediction — Model Accuracy: Moderate**

Predicts: approved-first-round

Table: Sample Invoice Approval Table



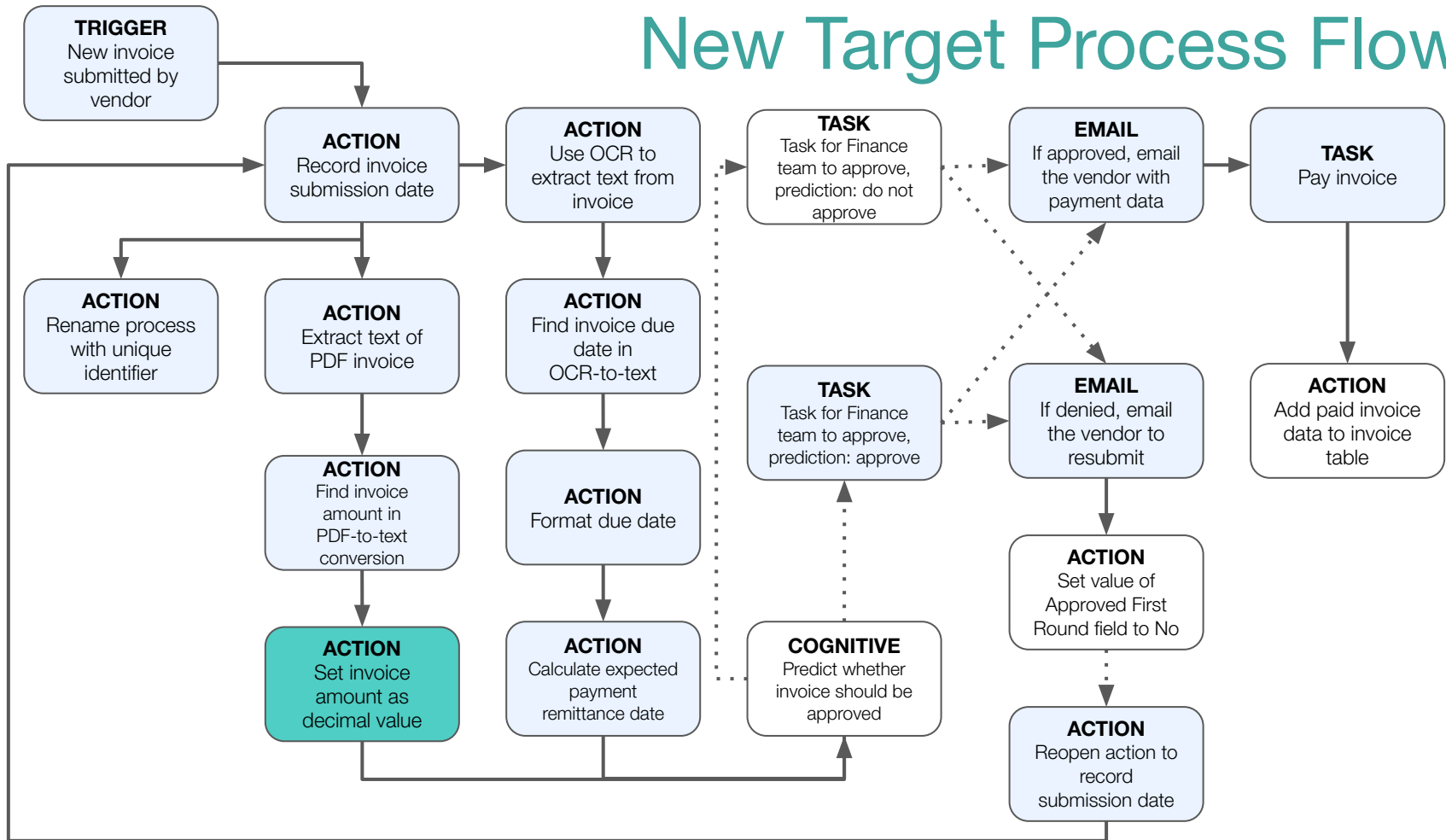
Create Predictive Model

# Updating Your Webform Trigger

We will add a hidden template-level placeholder field, capturing whether the invoice is “Approved” on first pass. If it is denied by the Finance team, the value will be automatically changed in a subsequent step to “Denied”.

1. To access the template-level fields page, click **Edit**, **Add an Action**, then **:** , and finally **Fields**.
2. We need to add a field called **Decision**
  - **Required:** Yes
  - **Type:** Choose One (Approved, Denied)
  - **Default:** Approved (Hidden)

# New Target Process Flow




# Action 1: “Set invoice amount as decimal value”

## PURPOSE

We will save the amount due text string as a numeric (decimal) value that can be added to our table.

## ACTION TYPE

Field: Set value of a decimal field

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Output field value</i>	The field or variable you wish to set as a decimal.	{{amount-due--first-match}} 
<i>Output field name</i>	The name of the field in which the decimal will be saved.	invoice-amount
<i>Output decimal places</i>	Designates the number of digits right of the decimal point to keep. To round the number, enter number of digits. Leave blank to leave unchanged.	2

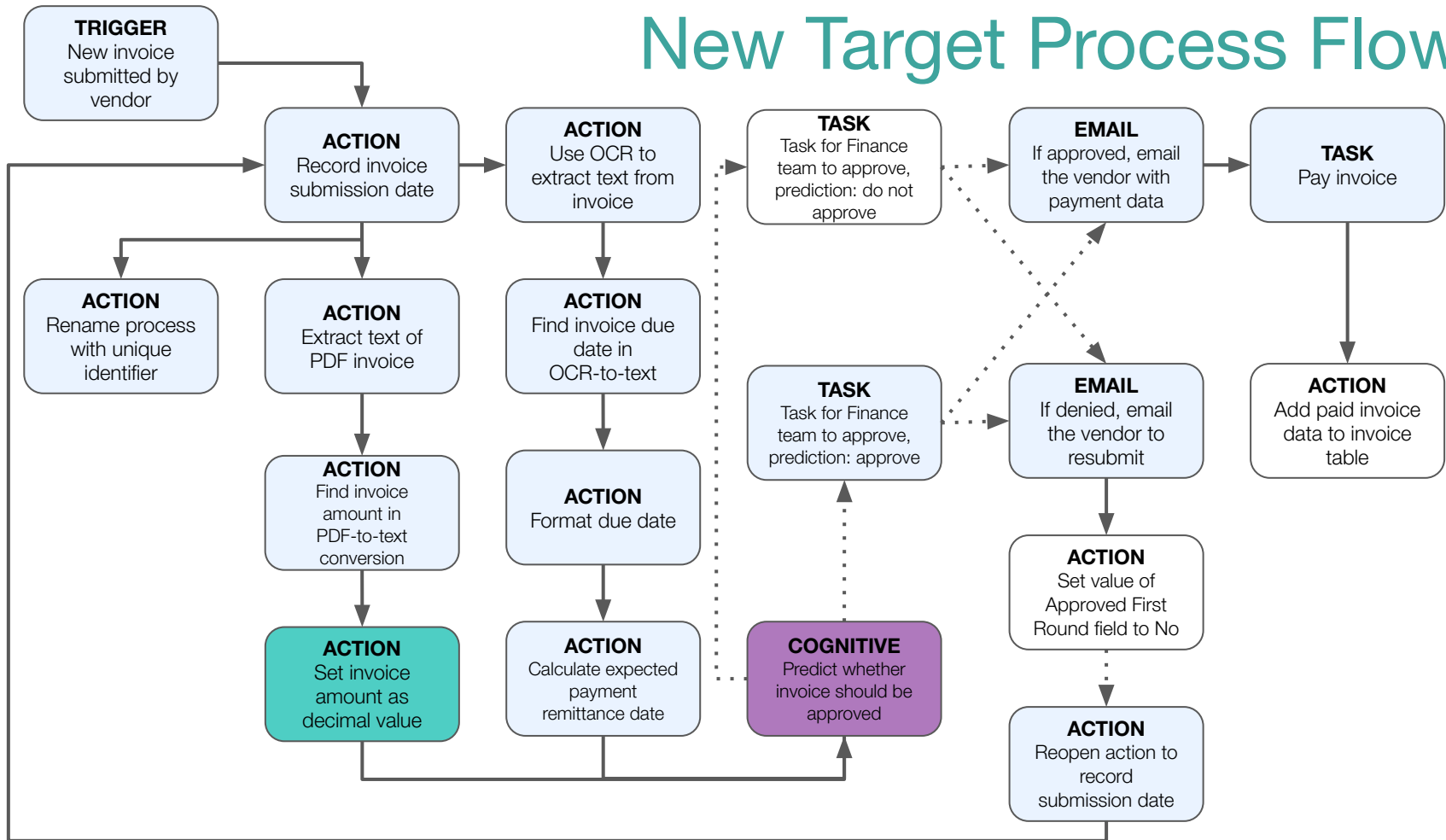
Use the pencil icon to reposition the action between:

- **Find invoice amount in PDF-to-text conversion**
- **Use OCR to extract text from invoice**

Click the check mark when finished.

DEPENDENCIES	<b>COMPLETED:</b> Find invoice amount in PDF-to-text conversion
CONDITIONS	

# New Target Process Flow



# Action 2: “Predict whether invoice should be approved”

## PURPOSE

Leverage our predictive model algorithm to guide on whether the submission should be approved, or given additional scrutiny, based on historical submissions.

## ACTION TYPE

Field: Make a prediction

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Predictive Model ID</i>	Designates the ID of the predictive model you wish to use in the action.	<i>Trainer to provide from trained model in list.</i>
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	invoice-approval-prediction

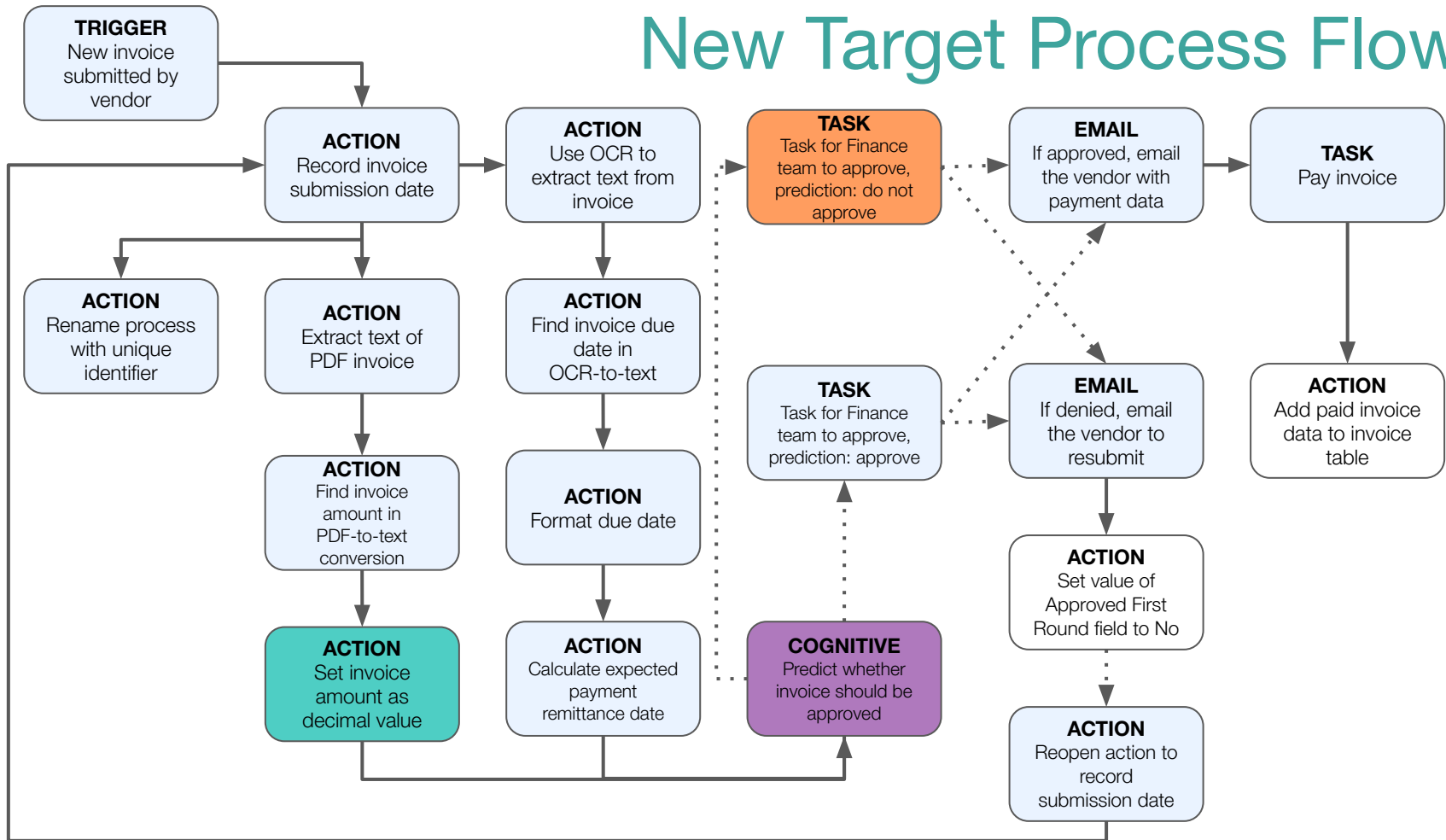
Use the pencil icon to reposition the action between:

- **Calculate expected payment remittance date**
- **Task for Finance team to approve**

Click the check mark when finished.

DEPENDENCIES	<b>COMPLETED:</b> Set invoice amount as decimal value; Calculate expected payment remittance date
CONDITIONS	-

# New Target Process Flow




# Action 3: “Task for Finance team to approve, prediction: do not approve”

## PURPOSE

Route the invoice to the Finance team requesting approval or denial, guiding that extra scrutiny is recommended.

## ACTION TYPE

Assign task to a person

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Assign to</i>	Whom the task will be assigned to: individual or a group. Can use a Catalytic username or an email address.	[Select yourself]
<i>Instructions</i>	Any instructions for the task assignee. Can include field references in {{handlebars}}.	Please review and approve or deny the {{invoice}} for {{organization}} in the amount of \${{invoice-amount}}. Catalytic reviewed historical invoices and predicts that you should review this invoice carefully, with {{invoice-approval-prediction--confidence}}% confidence. 

Use the pencil icon to reposition the action between:

- **Task for Finance team to approve**
- **If denied, email the vendor to re-submit**

Click the check mark when finished.

DEPENDENCIES	<b>COMPLETED:</b> Predict whether invoice should be approved
CONDITIONS	fields["invoice-approval-prediction--prediction"] == Denied



# Action 3: “Task for Finance team to approve, prediction: do not approve”

## PURPOSE

Route the invoice to the Finance team requesting approval or denial, with model guidance.

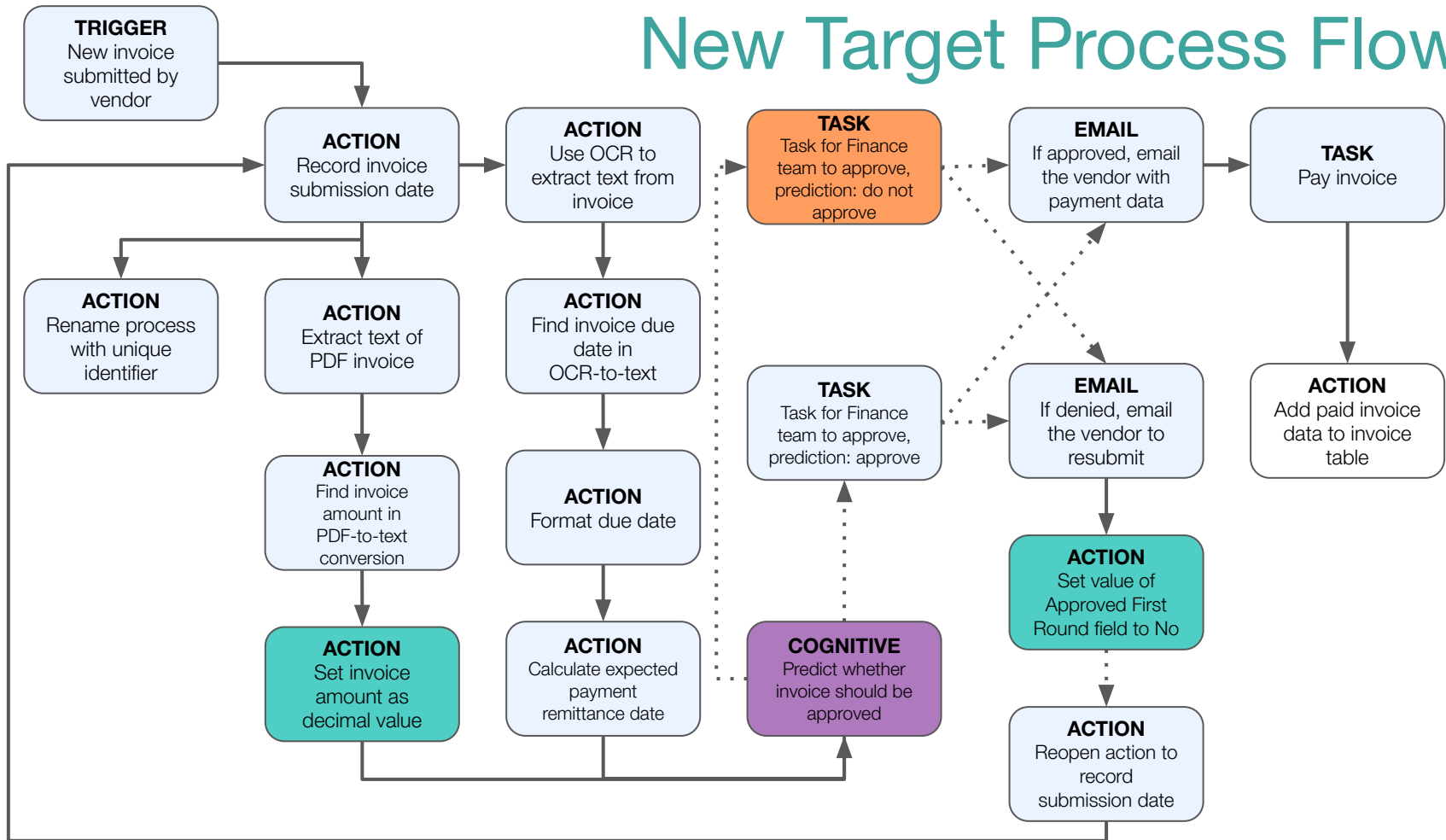
## ACTION TYPE

Assign task to a person

**Add the same fields used in the initial *Task for finance team to approve* action; note they are pre-configured due to single name space.**

FIELD NAME	TYPE	CONDITIONS
<i>Invoice approved</i>	Choose One (Approved; Denied)	
<i>Please provide a reason for denial</i>	Text	fields["invoice-approved"] == Denied
<i>Payment remittance date</i>	Date	fields["invoice-approved"] == Approved

# New Target Process Flow




# Action 4: “Set value of Decision field to Denied”

## PURPOSE

The first time the invoice is denied by the Finance team, we'll automatically record that the invoice was *Denied* in the hidden *Decision* field. This will be used to build the predictive model.

## ACTION TYPE

Field: Set the value of a multiple choice field

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Existing field name</i>	The name of the existing choose one or choose many field.	{{decision}} 
<i>Value</i>	Any text and fields in the {{field-name}} format to combine into the new value of the field.	Denied
<i>Choices</i>	Enter the possible choices for the field, one per line, to dynamically provide options in a task.	

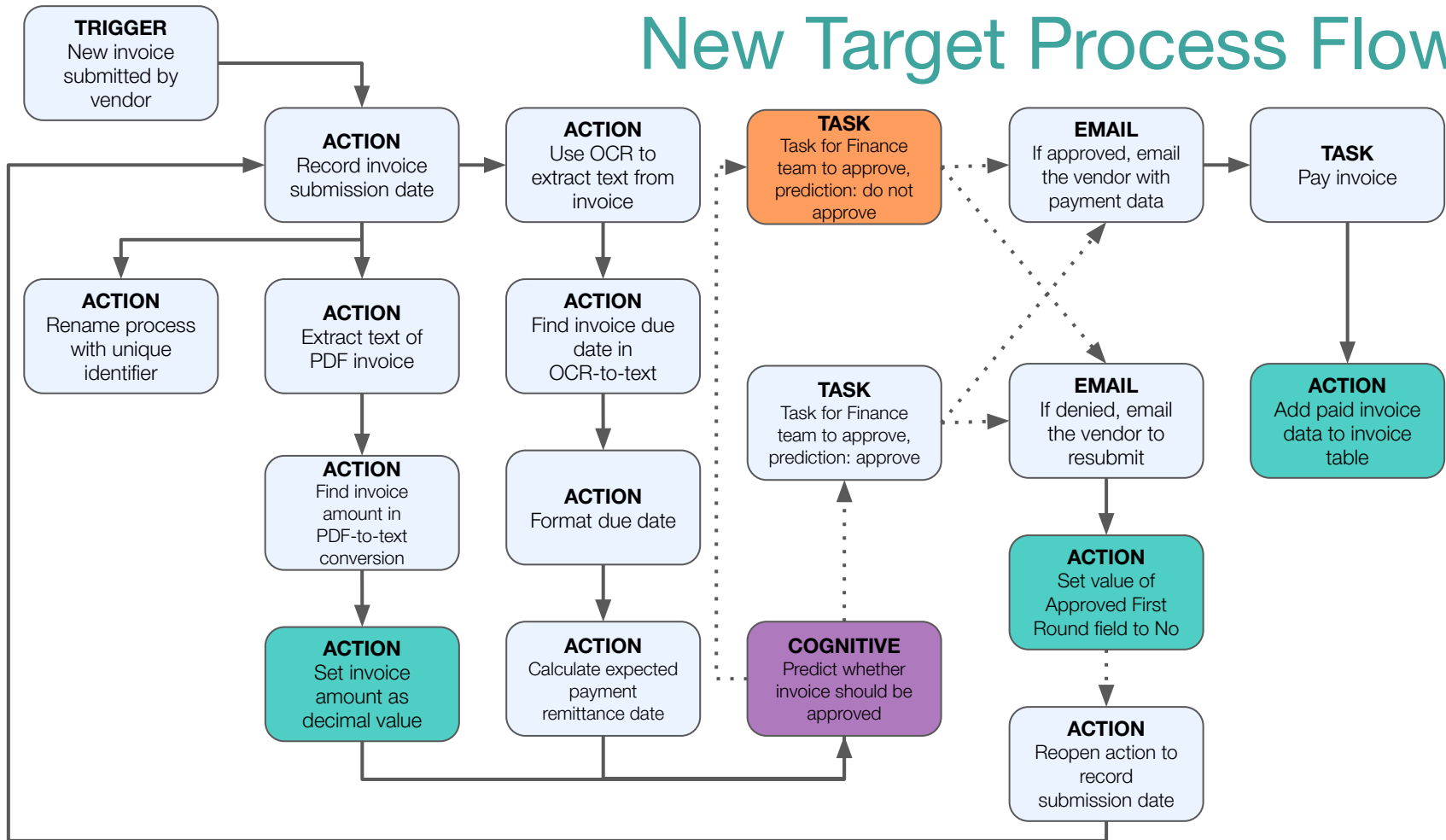
Use the pencil icon to reposition the action between:

- **If denied, email the vendor to re-submit**
- **Reopen action to record submission date**

Click the check mark when finished.

DEPENDENCIES	<b>COMPLETED:</b> If denied, email the vendor to re-submit invoice
CONDITIONS	fields["decision"] == Approved

# New Target Process Flow




# Action 5: “Add paid invoice data to invoice table”

## PURPOSE

Add row for invoice to keep data repository, and continually build the predictive model.

## ACTION TYPE

Tables: Add a row

CONFIGURATION FIELD	DESCRIPTION	USE HERE
<i>Data Table</i>	The table ID or reference in <code>{{handlebars}}</code> to which you will add a row.	25dbc37f-5842-475f-90fb-2ca56369a855
<i>Columns</i>	Comma-delimited list of columns you intend to record values to in the data table.	Name,Organization,Submission Date,Invoice Amount,Approved First Round
<i>Values</i>	Comma-delimited list of values to record to the above columns.	<code>{{name}}, "{{organization}}", {{submission-date}}, {{invoice-amount}}, {{decision}}</code> 
<i>Output field prefix</i>	This action may output more than one field. To help keep these fields organized, this prefix will be added to each output field name.	invoice-row-added

### FOR DISCUSSION

We put quotes (“ ”) around values to prevent delimiter collision.

### DEPENDENCIES

**COMPLETED:** Task to mark invoice as paid

### CONDITIONS

# UPDATE: “Task for Finance team to approve”

## PURPOSE

Route the invoice to the Finance team requesting approval or denial.

## ACTION TYPE

Assign task to a person

### Make the below changes to the action’s original configuration:

- **Change Name:** Task for Finance team to approve, prediction: approve
- **Dependencies:** Predict whether invoice should be approved [**Completed**]
- **Conditions:** fields[“invoice-approved-prediction--prediction”] IS EQUAL Approved
- **Instructions:**

*Please review and approve or deny the {{invoice}} for {{organization}} in the amount of \${{invoice-amount}}. Catalytic reviewed historical invoices and has {{invoice-approval-prediction--confidence}}% confidence in the accuracy of this invoice. {}*

# UPDATE: “If denied, email the vendor to re-submit invoice”

## PURPOSE

Send the vendor an email stating that the invoice was not approved and should be re-submitted

## ACTION TYPE

Email: Send a form

## Make the below changes to the action's original configuration:

- **Dependencies:**

- Start after all of these tasks are either **completed or skipped**:
- Task for Finance team to approve, prediction: approve
- Task for Finance team to approve, prediction: do not approve

- **Conditions:**

- fields[“invoice-approved”] IS EQUAL Denied

# UPDATE: “If approved, email the vendor with payment data”

## PURPOSE

Once the invoice has been approved, notify the vendor of approval and the expected payment remittance date.

## ACTION TYPE

Email: Send an email

## Make the below changes to the action’s original configuration:

- **Dependencies:**

- Start after all of these tasks are either **completed or skipped**:
- Task for Finance team to approve, prediction: approve
- Task for Finance team to approve, prediction: do not approve

- **Conditions:**

- fields[“invoice-approved”] IS EQUAL Approved



# UPDATE: “Reopen action to record submission date”

## PURPOSE

Creates a loop in the process so that every time an invoice is re-submitted by the vendor to the Finance team it is re-sent for approval.

## ACTION TYPE

Pushbot: Reopen task and reset dependent tasks

## Make the below changes to the action’s original configuration:

- **Dependencies:**

- Start after all of these tasks are either **completed or skipped**:
- Set value of Decision field to Denied

- **Conditions:**

- fields[“invoice-approved”] IS EQUAL Denied

# Let's Run It!

Check your dependencies. Submit a sample invoice found in the Process Builder folder via the webform you created. Use one of the below contacts:

NAME	COMPANY	INVOICE FILE
Brian Jones	ABC Incorporated	Sample_Invoice_ABC_Incorporated.pdf
Barb Williams	ABC Incorporated	Sample_Invoice_ABC_Incorporated.pdf
Dennis Smith	GHI Company	Sample_Invoice_GHI_Company.pdf
John Miller	GHI Company	Sample_Invoice_GHI_Company.pdf
Amy Wright	Sandbox et Cie	Sample_Invoice_Sandbox_et_Cie.pdf
Adam Davis	Sandbox et Cie	Sample_Invoice_Sandbox_et_Cie.pdf
Kim Johnston	Test Company, Inc.	Sample_Invoice_Test_Company_Inc.pdf
Kevin Brown	Test Company, Inc.	Sample_Invoice_Test_Company_Inc.pdf



# Predictive Model: Supplement

# Predictive Model: Qualifier

- Navigate to <https://catalytic.pushbot.com/form/predictive-model>
- Catalytic has a process/questionnaire to assess whether your use case is well-suited a predictive model:



## Catalytic Predictive Model Qualifier

Answer just a few questions to determine if you can add a [predictive model](#) with machine learning and NLP to your business process on Catalytic.

### Examples of use:

- Determine if an expense should be automatically approved
- Predict the likelihood of a sales opportunity to close within the next quarter
- Categorize and route an email based on text in the subject and body
- Recommend a compensation bonus amount based on employee performance data
- Suggest the best leader for a project based on project criteria

### Email address

Results of this analysis will be emailed to you after you submit this form

Required

### What do you want to predict?

Briefly describe your process and what you would like to predict or recommend

# Predictive Model: Starting Out

- Implement predictive models with a guided approach
- This leaves decisions to people while the model increases in accuracy
- Once confidence is high, you can remove manual tasks and configure action conditions to e.g. automatically approve invoices



Questions?

# Addendum: Building Best Practices



# Building Forms, Human Tasks

- Consult stakeholders to compile list of fields you will collect before form or human task creation
  - Ask: Do you really need this field? What is its value within process?
    - Imagine you are the user filling out this form, is it as simple as it can be?
    - If the field is optional, why include it?
    - Consider the process flowchart. How is process flow impacted the if field not captured?
  - Can this field be collected via an automation rather than a person (such as via integration or from a data table)?



# Building Forms, Human Tasks

- Your process may be highly automated, but may not be adopted by users if they do not like your form design.
- For fields, start with the easiest questions.
- Use concise, consistently formatted field names.
  - Leverage the **description** rather than field **name** for long questions.
  - This will make referencing fields easier (using {{handlebars}}).
- Consider conditions to hide fields until they become relevant in the form.
- If long, consider separating your form or task into sections
  - **Tip:** You can use an instruction-type field with hyphens; will render as a separator

# Field Permissions

- Be mindful when using **Send a form** or **Assign task to a person** actions, and configured assignee is dynamic {{email-address}}
  - Can be automatically converted to **Email: Send a form** if your recipient **does not** have an account within your Catalytic team
  - Unique links to forms **can** be shared by the original recipient
  - Fields classified internal, confidential, highly confidential will not display in public forms

# Field Permissions

- Master process run table respects field permission configuration
  - If a field is not visible to a user on the process run, it will **also not** visible in the process run data table
- Field permissions do not flow to imported data tables:
  - Example: A field is configured as confidential
    - Not masked if a process action is configured to write it to an imported data table row
    - If needed, you should maintain data table permissions to control access to process data

# Leveraging Markdown for Formatting

- You can format your human tasks and email messages using markdown formatting.
  - **Bold:** **\*\*text\*\***
  - *Italics:* *\*text\**
  - Hyperlink: [text](url)
    - Consider using [.]({{run.runID}}) in your email actions. This will make the email body unique, and reduces the chance of mail limits being hit.

# Application Limits

Regularly check the following two articles:

- **Pushbot System Limits**

[https://\[team\].pushbot.com/help/docs/pushbot-system-limits/](https://[team].pushbot.com/help/docs/pushbot-system-limits/)

This outlines limits around files, data tables, Excel and .csv files, etc.

- **Build with System Limits**

[https://\[team\].pushbot.com/help/guides/build-with-system-limits/](https://[team].pushbot.com/help/guides/build-with-system-limits/)

This article outlines limits around simultaneous task and instance processing.

# Integration Considerations

- Catalytic has several integration actions and triggers that can be used to pass data between systems, or start processes
- **Recommendation:** Create a dedicated user account within any system you intend to integrate with Catalytic
  - Doing so allows for greater control, reliability, and security, and simplifies configuration, testing, and support of the integration
- **Before deploying:**
  - If your IT or governance team requires a questionnaire be completed to issue a credential, your admin or Center of Excellence should work with your Catalytic Account Manager
  - \*Once in place, any member of your team can build processes leveraging those integration actions

# Wrap Up



# What We've Learned

- How to map a process from a manual process → Catalytic
- How Catalytic process is structured, including
  - Actions
  - Fields
  - Dependencies
  - Conditions
- Common and action-specific configurations
- How we use data tables to store and manage large datasets
- How to use a batch process to bulk perform actions on a spreadsheet



# Actions We've Covered

- Assign task to a person
- Dates: Adjust a date
- Dates: Format a date time
- Dates: Get current date
- Email: Send a form
- Email: Send an email
- Excel: Save spreadsheet to table
- Field: Make a prediction
- Field: Set the value of a multiple choice field
- Field: Set value of a decimal field
- Images: Optical character recognition (OCR)
- PDF: Extract text to a field
- Pushbot: reopen task and reset dependent tasks
- Rename this Pushbot
- Tables: Add a row
- Tables: Apply filters
- Tables: Convert data table to markdown text
- Tables: Convert data table to text
- Tables: Count rows in a table
- Tables: Get unique rows
- Tables: Look up data in a column
- Tables: Start Pushbot for each row
- Tables: Update a row
- Text: Find text next to other text

# Next Steps

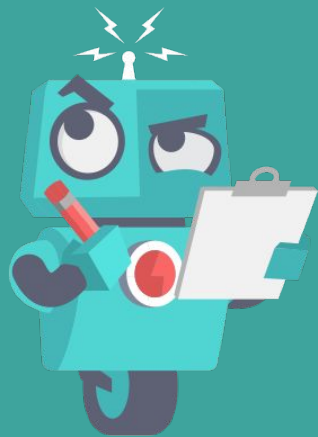
- Brainstorm process ideas and try mapping them to Catalytic actions
- Start to build out ideas to complete certification
- Practice, practice, practice!
- Reach out to us as needed

# Resources

- [help@catalytic.com](mailto:help@catalytic.com) for troubleshooting issues (i.e. bugs, errors, outages, etc.)
- [team.pushbot.com/help](https://team.pushbot.com/help) for support documentation
  - Help articles on every action configuration for when you're using a new one
- In-app documentation via global search, **Search Help**
- Stay tuned for upcoming continuous learning training



Questions?



# INTERNAL: Follow-Up for Process Builder Training

- Send Process Builder NPS survey
- Schedule assessment and check-ins
- **Starter:**  
<https://catalytic.pushbot.com/process/8654dbac-e7c6-4dde-8cae-eba6026f1cc0/edit>
- **Subprocess:**  
<https://catalytic.pushbot.com/build/563303f2-fa12-46ac-9206-f95bf47d19e6>
- **Data Table:**  
<https://catalytic.pushbot.com/data/tables/5a52114a-7827-437f-b8f7-3558029e7932>

