

GUIDES

6 min to read

Building a better runbook through automation and documentation

WRITTEN BY

Madhura Kumar

PUBLISHED ON

Jul 13, 2022

What is a runbook?

Runbooks can be thought of as manuals or playbooks used to guide you through a structured set of tasks. Each runbook has one or more tasks, each task building on the previous one.

Runbooks can be used for incident response, data processing and verification, customer account management, and many other processes. Their ability to standardize and verify internal processes makes them an essential part of any organization's documentation.

What is runbook automation?

[Runbook automation](#) is the process of using software to execute runbook tasks more reliably, quickly, and autonomously. It helps standardize workflows across teams, provides structure to

regular maintenance tasks and incident resolution, and allows information to be shared from a single source.

Runbooks are most effective when they demonstrate the “[five A's](#),” which are common attributes of a good runbook. In this guide we'll discuss how to use automation to make your runbooks actionable, accessible, accurate, authoritative, and adaptable.

Why is runbook automation important?

Runbook automation helps organizations share knowledge more efficiently. It provides integration with external services and access to your runbooks through the CLI or by the use of Python or Node.js, keeping your information up-to-date and easily accessible.

Runbook automation also provides security. A marketing associate likely doesn't need direct access to a production database, but they may need to transfer data from a CRM to begin an onboarding process. Access controls provided via runbooks can eliminate the need to navigate sensitive information.

How to improve runbooks with automation

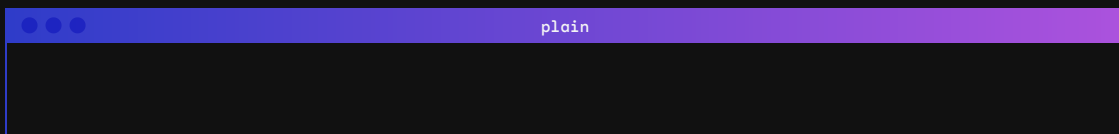
In short, automation lets you create documentation that is actionable, accessible, accurate, authoritative, and adaptable. Let's take a look at how you can use automation to create runbooks that meet each of these goals.

All the example code in this article can be found in [this GitHub repo](#).

Actionable

Automation allows you to extend manuals to be actionable. Traditional runbooks are not executable, which means that any action listed in a runbook must be done manually. Automation allows each step in a runbook to become executable documentation, providing a simple interface that includes information and the ability to act on it.

Here's an example of a simple process. Let's say that you're tasked with sending customer information from a CRM to a Slack channel on a weekly basis. The CRM database contains a table called `accounts` that has the following rows:



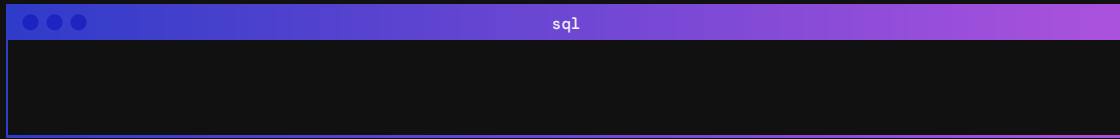
In a traditional runbook, querying the database and sending customer information to Slack would require the following five steps:

1. Log in to the CRM
2. Locate the customer using their ID
3. Export the customer information

4. Format the information for Slack
5. Send the information to a Slack channel

Using runbook automation, these five steps can be reduced to two tasks:

The first task will run the following `SQL` query against the selected database (this example uses the same CRM database mentioned above).



Then, using a Slack integration, the second task will format the data and post the message in your selected channel. By using the Slack integration rather than manually formatting the message and posting to the channel, you're saving time and preventing simple errors from corrupting the data you're passing along. You can use tools like [Airplane](#), which comes with an out-of-the-box [Slack integration](#), to spin up this workflow in just minutes.

Accessible

Some operations may require access to certain data, while still restricting access to other information.

You should consider using user-based and group-based permissions ([like these](#)) to limit access controls on sensitive operations. For example, your customer success managers may need to reference customer IDs, but you want to prevent them from viewing and altering billing information without approvals. You can set up your runbooks to require approvals before updating billing information or executing refunds that are above a certain threshold.

Access controls create a more secure environment for running tasks against production databases.

Accurate

Updating a traditional document-based runbook is time consuming and requires collaboration and verification to ensure that the document remains accurate when new information is added to it. Moreover, as processes change over time, the runbook becomes dated. Without a regular schedule, updates can easily be forgotten. Luckily, automation can do the scheduling for you.

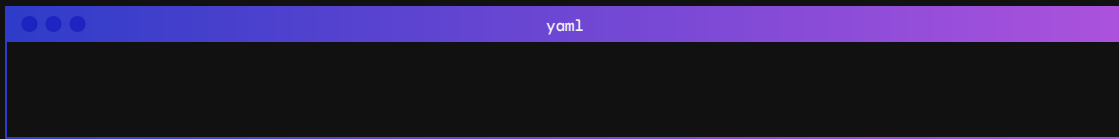
As you update the runbook itself, you can take advantage of testing to ensure all tasks in the runbook complete as expected. This gives you and your team the knowledge that the process will run successfully each time. If a runbook does not run successfully, you can look to the audit log as the source of truth by drilling down into the metadata of the execution to pinpoint where and when the error occurred.

To ensure accuracy, you might also want to use a command-line tool to programmatically create new tasks. This allows tasks to be templated, and the task description and code can be checked

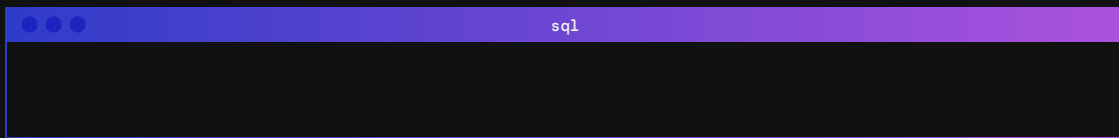
into a version control tool, like `git`. Using `git` will allow you to share scripts with your team for collaborative development, and easily roll back any changes that result in errors or missing data.

Automated task creation from the command line allows you to use simple configuration via YAML files that contain all of the relevant parameters needed for your action. The example below uses information from the `accounts` table in the CRM database mentioned above.

The YAML configuration file includes the following content:



The entrypoint `query_user.sql` file contains the following code:



When deployed, the information in `query_user.task.yaml` will be used to create a new task, if one doesn't already exist with the same name. You can now run this task by itself or add it as a step in a runbook.

Authoritative

In an ideal world, there would be only one source of truth for the documentation. However, in practice, there may be multiple drafts of the same document floating around, and locating the correct draft can easily become a chore.

You may not have time to ask around for the most recent copy and so you decide instead to make an educated guess and select a document that *seems* like it has the most recent information. However, even with an educated guess, you still can't be sure that the steps in the document are correct. If you choose to go down this path and encounter issues or errors, you'll likely end up feeling frustrated and losing valuable time.

Finding inaccurate data is a frustrating experience, especially when resolving an incident. Regular review cycles keep documentation updated, and storing documentation alongside a runbook as notes makes the revision a lot less time-consuming. During a review cycle, both documentation and runbook content will be reviewed and updated. After deploying an updated runbook, the tasks become immediately available to anyone who has the correct access permissions.

Adaptable

Runbooks are frequently used as a part of incident response, but automation can allow them to adapt to almost any purpose. For example, say the marketing development team would like a new process for onboarding prospective clients. An example runbook could instruct developers to enter and verify information, provide relevant access to company services, and automatically send a welcome email.

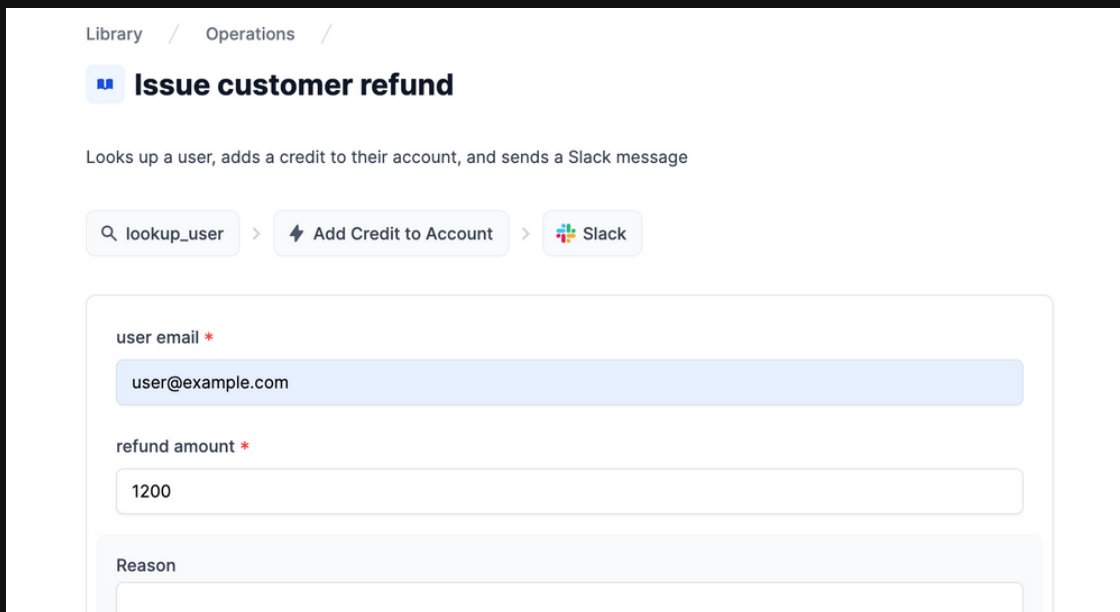
Additionally, as your organization grows, the processes must be modified to fit any new requirements. Runbook automation simplifies this process by providing a standard method of creating and executing runbooks and automates the review process of the newly created runbook to ensure that all new information is accounted for. This serves to make runbooks and runbook automation a key part of the documentation process.

Airplane for runbook automation

[Airplane](#) is a developer platform for quickly transforming scripts, queries, APIs and more into internal tools for you team. Airplane is a full-featured solution that takes just minutes to get set up with and offers runbook automation, [scheduled jobs](#), [nuanced permissions](#), [audit logs](#), and more out of the box.

You can create workflows in Airplane using the [Airplane CLI](#) and add resources like [PostgreSQL](#) and [SendGrid](#) to your processes.

If you're interesting in trying out Airplane, you can watch a [4-min demo video](#) on our website and [sign up for a free account](#) to get started.



The screenshot displays the Airplane interface for a workflow titled "Issue customer refund". The breadcrumb navigation shows "Library / Operations /". Below the title, a description reads: "Looks up a user, adds a credit to their account, and sends a Slack message". The workflow steps are visualized as a sequence of three boxes: "lookup_user", "Add Credit to Account", and "Slack". Below the steps, there are input fields for "user email *" (containing "user@example.com"), "refund amount *" (containing "1200"), and "Reason".

Author: [Alvin Charity](#)

Alvin Charity has over 10 years of writing and editing experience for a variety of environments, including retail, technology, entertainment, and internal communications. He is also a self-taught Javascript developer and musician.

Share this article: [f](#) [t](#) [in](#)