



Whitepaper

# GxP-Compliant Workloads on Databricks



## Table of Contents

---

<b>Overview</b>	3
Overview of GxP compliance for Databricks	3
What is GxP?	4
<b>Databricks Architecture</b>	5
<b>Databricks Corporate Responsibilities Related to GxP</b>	7
Compliance standards at Databricks	7
Document and records management	7
Employee hiring	7
Vendor management	8
Business continuity	8
SDLC	8
Incident management	13
Platform security and security functionality	14
<b>Shared Responsibilities for GxP</b>	15
Encryption and data security	15
Logging and system monitoring	16
Managing versions	16
Disaster recovery	16
Access management	17
Patching	17
Intrusion monitoring	17
Employee training	18
<b>Customer Responsibilities for GxP</b>	19
Standardized deployment models	19
Data availability and retention	19
Change management	21
Validation	23

## Overview

---

### OVERVIEW OF GXP COMPLIANCE FOR DATABRICKS

Databricks can support **GxP-compliant workloads**, and Databricks has customers that are running GxP-compliant workloads on the Databricks Lakehouse Platform. Because Databricks is a general-purpose data-agnostic compute platform provided as a PaaS, Databricks by its very nature does not need to be GxP-certified or GxP-compliant (and cannot be). Customers can still run GxP-compliant workloads on Databricks.

Overall, Databricks has over 350 customers in the Healthcare and Life Sciences industry, including 9 of the 10 largest pharmaceutical companies globally. Our customers — including industry leaders such as AstraZeneca, GSK, Amgen, Sanofi and Biogen — use Databricks for a broad range of use cases such as genomics, translational research, clinical trial optimization, inventory management and commercial analytics.

Databricks leverages a shared responsibility model, and the majority of GxP compliance relates to the design of the workloads that the customer chooses to run. As Databricks is a compute platform, customers will leverage standard operating procedures (SOPs) for deployment and monitoring, along with a strong software development lifecycle (SDLC) for their development pipeline. Databricks is well suited to support both of these, due to the existence of API support, strong enterprise security controls, log availability, and common integration with the continuous integration/continuous development (CI/CD) pipelines that support a strong SDLC. Databricks is happy to work with customers in all scenarios to help adapt their SOPs and SDLC requirements to Databricks.

Though the majority of requirements are customer-specific, there are key Databricks deployment strategies common among customers running GxP-validated workloads. The key controls that will be detailed in this document are:

- Managing the version of the Databricks Runtime available to your users
- Managing the version of your code for validated jobs (e.g., via Databricks Repos or other CI/CD processes)
- Reviewing logs from your code and your systems
- Deploying a disaster recovery and backup strategy
- Applying patches routinely via cluster restarts

Additionally, we will outline some strategies used by customers who have greater requirements around GxP compliance, such as using Databricks Container Services for extensive control over the runtime environment (operating system and package versions).

We will then provide details about the processes and controls in place within Databricks that customers evaluating GxP compliance often ask about, and move to discussing customer-side responsibilities.

In short, this document will provide guidance for how other customers have used Databricks as a part of their GxP-compliant workloads. Additionally, Databricks can recommend consulting partners who have worked with other customers to install, configure and validate Databricks in GxP-compliant environments.

## WHAT IS GXP?

GxP stands for “Good x Practices” where x refers to a specific discipline, such as clinical, manufacturing or laboratory. The goal of GxP compliance is to ensure that a compliant process runs reliably, can survive failures and human error, and can be relied upon. For example, there is a collection of guidelines, standards and regulations that govern the creation and manufacture of drugs, and it’s important to society that the final product provided to consumers can be relied upon. In the United States, many GxP guidelines are described in federal regulations such as 21 CFR Part 11.

When data is being processed in a workflow that falls under GxP, the data processing infrastructure generally also falls under GxP. The infrastructure that stores and processes that data may need to be validated to verify that the software/infrastructure is properly installed and configured. Validation can be achieved by defining SOPs for system installation or deployment, and by monitoring and auditing the configuration of the installed infrastructure.

Any software algorithms used may also need to be validated to verify that the algorithm reliably produces a correct result. For software, validation is normally achieved through strong SDLC processes.

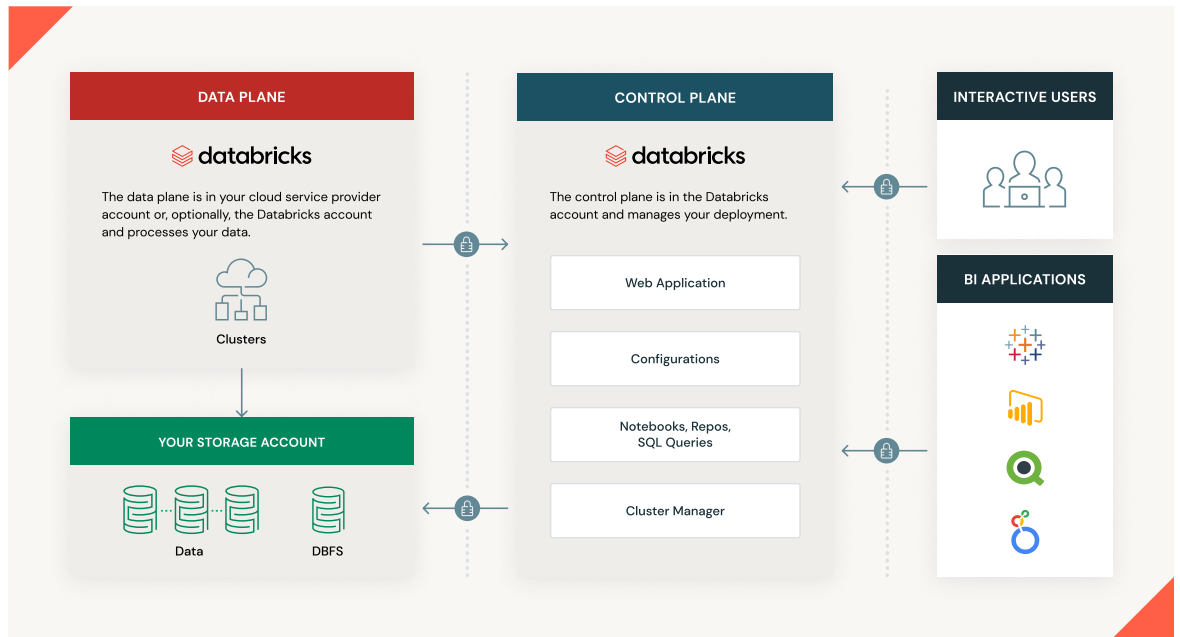
The requirements imposed by GxP are similar to those imposed in other regulated settings, such as CLIA/CAP requirements for validating results returned in clinical laboratories.

# Databricks Architecture

The Databricks architecture is somewhat unique, and it is useful to have a brief overview to help understand the rest of this document. Security is a broad topic, and you can find additional details in our [Security & Trust Center](#).

The Databricks architecture is split into two planes to simplify your permissions, avoid data duplication and reduce risk.

- **Data plane:** The data plane handles your data processing. With serverless deployments, the data plane exists in your Databricks account rather than your cloud service provider account.
- **Control plane:** The control plane manages your deployment. Includes web app, jobs, cluster management, notebooks.



Databricks leverages a shared responsibility model. See more details on customer security configurations within our Enterprise Security Guide, and the Security Best Practices document available in our Security and Trust Center under [Security Features](#). On the same page, you will also find the Security Analysis Tool, which you can use to analyze the hardening of your deployments against our best practices.

## END-TO-END EXAMPLE

Suppose you have a data engineer that signs in to Databricks and writes a notebook that transforms raw data in Kafka to a normalized data set stored in object storage (such as S3, GCS or Blob store).

### Six steps occur in the example

- 1 | Your single sign-on seamlessly authenticates the data engineer to the Databricks web UI in the control plane.
- 2 | As the data engineer writes code, their web browser sends it to the control plane. JDBC/ODBC requests also follow the same path.
- 3 | When ready, the control plane creates a Databricks cluster in the data plane.
- 4 | Once the cluster is online, the control plane sends the data engineer's code to the cluster.
- 5 | The cluster pulls from Kafka, transforms the data and writes it to your storage.
- 6 | The cluster reports status and any outputs back to the control plane.

The data engineer doesn't need to worry about many of the details — they simply write the code and Databricks runs it.

# Databricks Corporate Responsibilities Related to GxP

---

## COMPLIANCE STANDARDS AT DATABRICKS

While Databricks isn't itself subject to GxP compliance, Databricks undergoes a number of compliance certifications and audits that may be relevant to customers. Databricks is a multicloud solution, available on AWS and GCP. Additionally, the Azure Databricks product is available as a first-party Microsoft service to provide Databricks capabilities to the Azure ecosystem. For details on our overall compliance offerings (ISO 27001, 27017, 27018, HIPAA, PCI and more), please visit our [Security & Trust Center](#).

## DOCUMENT AND RECORDS MANAGEMENT

The primary source of truth for most document management relating to the Databricks platform is our JIRA platform, which tracks activities, changes and actions taken. Databricks utilizes a cloud deployment of JIRA managed by Atlassian and currently does not archive any older tickets from the platform.

Certain documents are also managed independently by the Google Suite. An example of this are design documents created to track material changes to the system. These are stored version-controlled, built based on a standard template, and there are standard review workflows for these documents.

Finally, a key component for document management within Databricks is our source code management system, currently stored in GitHub. This contains a record of what code has been merged and the approvals and conversations for those merges.

## EMPLOYEE HIRING

Databricks maintains robust processes around employee hiring. Excluding limited specialized roles (for example, executive leadership), Databricks builds standard job descriptions that are posted on our careers site. Hiring managers, in conjunction with recruiters, are responsible for building interview plans that are uploaded into a SaaS app used for tracking a candidate from initial interest through to offer acceptance, and which centrally tracks feedback from interviewers. Job descriptions are tracked within the SaaS app.

## VENDOR MANAGEMENT

Databricks has a centralized procurement team that coordinates the tracking, negotiation, contact, and contract management for external vendors. Databricks has a portal used by staff to submit details about new vendors for approval, and a typical purchase management process. Databricks maintains a supplier diversity statement.

[Click here](#) for a list of third parties who qualify as subprocessors for the Databricks service.

## BUSINESS CONTINUITY

Databricks has a formal, documented Business Continuity Plan and Business Impact Analysis that is reviewed at least annually. The Business Continuity Program is owned by the Assurance team, and is overseen by the Databricks CSO.

## SDLC

### Design

The first step in our SDLC is the design phase. At Databricks, future products start with just an idea in our [Ideas Portal](#). We use this to track and prioritize enhancements big and small, and we also allow voting on ideas both by internal employees and by our customers.

Significant changes (non-minor adjustments to the product itself) will go through a feature design process, which includes privacy and security by design. This involves creating one or more design documents that explain the problem, the planned resolution, and any major design decisions that need to be made. Design documents must be either sponsored or authored by senior-level engineers, and must be reviewed by senior engineering and by leadership. Databricks also has a section of the design document template focused on security, compliance and privacy impacts.

High-impact features (such as those that materially change handling of customer data or the security of the system) are also subject to a security design review from a security expert in engineering with the goal of identifying any potential security or compliance issues. A threat model is also generally required as an input into that process.

Once a design is complete and the business need for that design is established, it will generally be scheduled for development.



## Development

As we're a software company founded in open source that only provides an as-a-service offering, customer satisfaction and adoption are our primary drivers. This is true even in terms of performance evaluation, where possible staff compensation is driven not by contract agreements or product delivery, but rather by the amount of usage. Security or stability issues would impact our ability to meet those drivers, and so the company does not sacrifice around training or operational processes that meet those needs. The root of avoiding quality and security issues is in maintaining high-quality development.

We use an agile development methodology and break up new features into multiple sprints. Databricks uses JIRA for managing this process and as the official tool for tracking issues or concerns that arise either during the development process or after a feature is put into production. For a sizable project, generally the team will create an epic, and then a series of individual JIRA tickets that represent units of work as a part of that epic. The epic is typically tied to an Aha! Feature, which is typically linked to an Aha! Idea (furthering the end-to-end linkage).

The most important component in high-quality development is high-quality developers. Databricks does not outsource the development of the Databricks platform and maintains high standards for developer hiring. Once hired, all developers are required to go through secure software development training, including the OWASP Top 10 at hire and annually thereafter. Databricks maintains a hierarchy where newer developers are overseen by more senior engineers and/or engineering managers.

In the development process, production data and environments are separated from the development, QA and staging environments. All code is checked into our GitHub source control system, which requires single sign-on with multifactor authentication. When an engineer creates a pull request to merge their new code into the main codebase, their request will require approvals. Databricks has built out an in-depth system for approvals where merge requires approval from the functional engineering owners of each area that their code impacts (this can frequently be multiple staff). All code is peer reviewed.

## Quality assurance

Quality assurance checks are also a key element of providing quality and security. We run quality checks (such as unit tests and end-to-end tests) at multiple stages of the SDLC process, including at code merge, after code merge, at release and in production.

Our testing includes positive tests, regression tests and negative tests. Once deployed, we have extensive monitoring to identify faults, and users can get alerts about system availability via the [Status Page](#). Databricks also collects a large volume of metrics about system functionality so that we can quickly make root cause determinations and understand whether issues are new or recurrent.

In the event of any closed significant (PO or P1) issue or any issue that impacts customers, Databricks automation changes the JIRA status to trigger a postmortem process. The Databricks postmortem process follows a “5 whys” root cause analysis methodology. The engineer who owned the issue is responsible for creating the initial report, and their work is overseen by a member of the postmortem virtual team (senior engineers within the organization). Timelines are required for each stage of the postmortem process so that it doesn’t linger and staff don’t forget details. The final result is emailed to a postmortem email alias that includes engineering staff and leadership, and follow-ups are tracked (generally also in JIRA).

### Security scanning

Detecting and quickly fixing vulnerable software that you rely on is among the most important responsibilities of any software or service provider. We take this responsibility seriously and share our remediation timeline commitments in our [Security Addendum](#).

Internally, we have automated vulnerability management to effectively track, prioritize, coordinate and remediate vulnerabilities in our environment. We perform daily authenticated vulnerability scans of Databricks and third-party/open-source packages used by Databricks, along with static and dynamic code analysis (SAST and DAST) using trusted security scanning tools, before we promote new code or images to production. Databricks also employs third-party experts to analyze our public-facing sites and report potential risks.

Databricks has funded a Vulnerability Response Program for monitoring emerging vulnerabilities before they’re reported to us by our scanning vendors. We accomplish this using internal tools, social media, mailing lists and threat intelligence sources (e.g., US-CERT and other government, industry and open-source feeds). Databricks monitors open vulnerability platforms, such as CVE Trends and Open CVDB. We have an established process for responding to these so we can quickly identify the impact on our company, product or customers. This program allows us to quickly reproduce reported vulnerabilities and resolve zero-day vulnerabilities.

Our Vulnerability Management Program is committed to treating Severity-0 vulnerabilities, such as zero days, with the highest urgency, prioritizing their fix above other rollouts.

### Vulnerability timeline starting upon the availability of a fix

- 1 | Critical: 14 days
- 2 | High: 30 days
- 3 | Medium: 60 days
- 4 | Low: As appropriate

Please note the shared responsibility surrounding patching [covered below](#).

### Release management

Databricks has implemented a “train conductor process” for managing releases. The train conductor process takes approximately two weeks and includes several components. Changes go through testing procedures that are designed to help ensure when a release arrives to customers; it does not contain regressions and that new functionality has been tested on realistic workloads.

Code versioning software, GitHub, is used to track and provide control over changes to source code. The software allows development personnel to check out code and modify it locally on their computer to ensure the production code will not be affected during the development process. Each piece of code that is checked back in is assigned a different version number, which allows users to roll back to previous versions if changes impair system operation. The ability to modify source code within the versioning software is restricted to authorized personnel. Development and testing environments are logically separated from production, and customer data is not used for testing purposes unless authorized by the customer.

Pull requests have supporting engineering tickets or design docs, based on the size of the change. Once a pull request is ready, peer review is performed and review comments addressed, the pull request can be approved by a functional owner for that area. Approval is documented in GitHub and automatically validated.

Prior to migration to the production environment, management approval is obtained through a “Go-No-Go” meeting. Before release, code should pass all checks (such as unit tests) designed to provide assurance the product works as expected by the design document. To implement separation of duties, only our deployment management system can release changes to production, and multiperson approval is required for all deployments.

Documentation for docs.databricks.com is driven by tickets just like new code. Engineering tickets that require docs are tagged, then linked to docs tickets that track progress. Docs tickets can also be created for general enhancements. The docs site is source controlled via GitHub just like the Databricks product and goes through a limited CI/CD process to push to production after approval by the docs team.

Our change management policy is formally documented, communicated to constituents, approved by management and reviewed at least once per year.

### Rollout

Databricks has a staged rollout process with monitoring to identify issues at early stages. Typically, releases are rolled out over the course of a week, with the busiest regions being deployed last. Release management staff is responsible for rollouts, and the staff members responsible for development do not have access that allows them to publish new releases, consistent with the separation of duties principle.

We follow the immutable infrastructure model, where systems are replaced rather than patched, to improve reliability and security by avoiding the risk of configuration drift. When new system images or application code is launched, we transfer workloads to new instances with the new code. This is true both for the control plane and for the data plane.

Once code is in production, a verification process confirms that artifacts are not added, removed or changed. For example, when deploying a new container, the container checksum is calculated and compared against the expected checksum. If there is a mismatch (and thus the potential that the container itself has been compromised), the container won’t launch.

With the exception of Databricks on GCP, Databricks does not rely on a public container registry. Containers are created by our build process and stored in an object store within the cloud service provider (such as in a Databricks-owned S3 bucket).

## Documentation

The last phase of the SDLC process is to create customer-facing documentation. Databricks docs are managed similarly to code, where the documentation is stored within the same source control system. Significant changes require technical review as well as review from the docs team before they can be merged and published.

## INCIDENT MANAGEMENT

Databricks is responsible for providing incident management in multiple areas: security and privacy incidents, availability incidents and support incidents.

### Security and privacy incidents

As discussed in more detail in our [Security & Trust Center](#), Databricks maintains an incident response program that handles incoming security and privacy incidents. Databricks has a standard process for these incidents, and uses the same systems of record discussed above in [Document and records management](#).

### Availability incidents

As discussed above under [Quality assurance](#), once code is deployed, we have extensive monitoring to identify faults, and users can get alerts about system availability via the [Status Page](#). Databricks also collects a large volume of metrics about system functionality so that we can quickly make root cause determinations and understand whether issues are new or recurrent.

Databricks has an automatic postmortem (corrective and preventive action, or CAPA) process in the event of any PO or PI issue or any issue that impacts customers. Databricks automation triggers a workflow to launch the postmortem process, which follows a “5 whys” root cause analysis methodology. Automation also selects a member of the postmortem team to oversee the review. Timelines are established for the review process so that it is started immediately and completed promptly. Follow-ups are tracked, and the outcome report is emailed to a distribution list that includes senior staff across many teams.

### Support incidents

Databricks provides support directly to customers for issues ranging from general questions to production-impacting incidents. Customers can choose the level of service required for their use case, as described in more detail on our [support site](#).

## PLATFORM SECURITY AND SECURITY FUNCTIONALITY

Databricks has always differentiated itself based on our strong security program, both from our internal security team and the security features available to customers within our product. There are many security capabilities that aren't explicitly tied to GxP and so aren't covered in this document – even though they're often required by customers who run GxP-compliant workloads due to the sensitivity of the data or the criticality of the workload. Common examples include customer-managed encryption keys, IP access lists, Private Link, and more.

For more detailed information, please refer to our Security & Trust Center:



### Architecture →

Overview of the Databricks architecture (duplicated above)



### Trusted platform →

Covering capabilities we maintain internally



### Security features →

Features commonly used by customers, along with best practice guides



### Compliance →

Features commonly used by customers, along with best practice guides



### Privacy →

Legal guidelines and contractual language related to our security and privacy program

## Shared Responsibilities for GxP

---

### ENCRYPTION AND DATA SECURITY

Customers can easily enable encryption at rest and in motion within the data plane, and Databricks encrypts data at rest and in motion in other systems. Data at rest is AES-256 bit encrypted, while data in motion is TLS 1.2 encrypted. Customers that require control over the encryption key can utilize the Customer-Managed Key feature.

As a part of the Databricks shared responsibility model, customers may be responsible for implementing encryption for data in transit between cluster members within a customer's data plane. This is mandatory for AWS customers using our [security compliance profile](#) (customers of HIPAA, PCI or FedRAMP Moderate offerings), and cannot be disabled by customers. Databricks on GCP automatically encrypts all data. If a customer determines that intra-cluster encryption is required on Azure, it can be implemented for all clusters in a workspace using init scripts.

Providing encryption for data at rest is also key to customers. Much like data in transit, this configuration is already in place for Databricks systems if a customer utilizes our security compliance profile on AWS or GCP. Azure customers who determine they need to encrypt local SSDs used for shuffle data on clusters can enable SSD encryption via cluster policies that will apply to all hosts in the environment.

Additionally, while Databricks accesses and processes data sets at your instruction, Databricks is typically not used as a data storage system. Most data is stored in your cloud hosting provider accounts using bulk data storage services such as AWS S3 or Azure Data Lake Storage (ADLS). Customers must configure their storage accounts to ensure that data stored there is encrypted.

## LOGGING AND SYSTEM MONITORING

Databricks provides built-in audit logging, though the customer must configure it. Databricks will typically deposit audit events every five minutes into an Azure Blob or S3 bucket owned by the customer. This audit logging aims to capture activities of Databricks users on the control plane (logins, cluster launches, and similar). Find details around the audit events [on our public docs](#).

Databricks also integrates well into typical cloud monitoring tools. For example, AWS customers can monitor the provisioning activities of Databricks (such as launching a new instance) via AWS CloudTrail and can monitor their network and storage accounts accordingly. Similarly, they can monitor the utilization of data plane hosts via CloudWatch.

## MANAGING VERSIONS

Databricks provides long-term support (LTS) releases of the Databricks platform, as detailed below in the change management system. Databricks administrators at our customers can control what versions of the software are available to their users, along with the instance types that can be used, via a feature called [cluster policies](#). Cluster policies can facilitate that certain configurations are applied to a workspace and that they use an approved version of the Databricks Runtime on an approved instance type. In addition, the Databricks auditing and logging framework provides timestamped audit trails of which runtime versions and policies were applied historically.

## DISASTER RECOVERY

While Databricks does have a business continuity plan, a backup policy and procedure that is routinely tested, and disaster recovery plans, Databricks does not offer disaster recovery capabilities to customers. Instead, we provide best practice guidance and field support for customers that implement their own required disaster recovery capabilities.

Databricks can be deployed across many regions via the multi-workspace API to survive regional infrastructure faults. We provide details on how to support this configuration for [AWS](#), [GCP](#) and [Azure](#) as a part of a broader cloud provider-specific DR approach, and Databricks provides tools that enable customers to implement backup or sync such as [databricks-sync](#).



## ACCESS MANAGEMENT

Databricks is responsible for managing access of Databricks employees (along with our contractors) within the production environment for Databricks. Databricks employs least privilege and separation of privileges, and has a just-in-time access control system, along with standard IT best practices such as password policies. Databricks also reviews the accounts of users who have access to internal systems on a quarterly basis.

Customers are responsible for managing the access of their employees (or any other users they grant access to, such as contractors). Implementation varies by cloud (please see our [security best practices](#) along with our [data governance capabilities](#)), but all customers are able to implement privilege-based data access. Customers who are implementing any type of electronic signature workflow that utilizes Databricks should also be attentive to access management requirements.

## PATCHING

Customers are responsible for restarting clusters periodically. Databricks does not live-patch systems — when a cluster is restarted and newer system images or containers are available, the system will automatically use the latest available images and containers. AWS and GCP workspaces with our Compliance Security Profile (PCI, HIPAA, FedRAMP Moderate) have automatic scheduled restart enabled, which automatically restarts clusters older than 25 days during a scheduled maintenance window.

Customers are responsible for the patching of the code they provide, or any libraries they install (such as via pip). Most commonly this is implemented via periodic restarts of the clusters.

If customers use Databricks Container Services (DCS), they are responsible for patching the Docker container.

## INTRUSION MONITORING

Databricks leverages a capability for security information event management, alert management and centralized monitoring. This includes open source products, third-party products and the use of Databricks itself for managing the security posture of Databricks.

Among these capabilities, Databricks runs AWS GuardDuty on all of its AWS accounts, including the AWS control plane. GuardDuty can monitor and alert on malicious activity, providing intelligent threat detection. Databricks performs similar monitoring on GCP control plane systems using Google Security Center. For the Azure Databricks product, Azure is responsible for monitoring and can provide detail on what is available.

Customers can choose to implement capabilities within the data plane hosted within their environment, such as by enabling GuardDuty or Google Security Center.

## EMPLOYEE TRAINING

Databricks uses a centralized learning management system that tracks onboarding training and annual compliance training, along with optional training and one-off training programs. The system retains records for which employees have completed training activities, and Databricks maintains manager and leadership dashboards reporting on completion percentage for mandatory training, along with email reminders to staff.

All employees are required to go through standard training at hire and annually thereafter. These trainings cover topics such as security awareness and privacy. These trainings typically include quizzes to measure learning and engagement.

Certain groups of employees have additional mandatory training, such as our developers, who are required to go through secure software development training, including the OWASP Top 10.

Databricks also makes available certain role-specific training internally based on specific requirements, typically through the same centralized learning management system. There are many examples of these trainings, but a key example would be the new manager training for managers who are either new to management or new to Databricks.

Customers also have a responsibility to train their staff. **Databricks provides** both online and instructor-led training. Customers can implement required training for their staff of the essentials of Databricks, along with role-specific training into deeper topics such as administration, data science or data engineering. Additionally, customers who qualify specific versions of the Databricks platform may release new internal training to coincide with the availability of a new version.

## Customer Responsibilities for GxP

---

Databricks is not in a position to provide authoritative guidance for all customer responsibilities while deploying GxP workloads on the Databricks platform, but we would like to share the available guidance we have based on existing customer conversations. There is a wide variety of requirements and implementation requests where Databricks has advised customers – within this section, you will find the most common requirements.

### STANDARDIZED DEPLOYMENT MODELS

Some Databricks customers using compliant GxP workloads focus heavily on perfecting automated deployments so that they can scale out their workloads to a large number of use cases. Their goal is to validate their deployment mechanism so that the incremental lift is marginal, and to provide uniformity around configurations: for example, **limiting approved DBR versions** via cluster policies.

The recommended and most common approach for automating deployments of Databricks is using Terraform. As shown in [our documentation](#), Terraform allows for straightforward configuration of the Databricks configurations and can also be used to configure your Azure, AWS or GCP networks so that the entirety of the deployment is automated. Terraform configurations are implemented in code to allow for easy tracking and change management.

### DATA AVAILABILITY AND RETENTION

Within Databricks, customers own the data that they analyze and process, along with the assets that are stored in the platform (such as notebooks, models, and similar).

To understand data availability and retention, it's useful to break out data into categories: core customer data, customer platform assets, and operational metadata.

#### Core customer data

Core customer data consists of the customer data sets that the customer processes using Databricks, which are stored at rest in customer-designated storage. When a data engineer processes a data set (such as enriching, summarizing or cleaning data), they will often choose to save that data to some organized storage location within the customer's cloud storage provider account such as an S3 bucket, GCS bucket, or to ADLS. The customer is then responsible for maintaining the availability and managing the retention of this data (such as using S3 replication or lifecycle policies).

## Customer platform assets

Customer platform assets are created by a customer on Databricks to process data.

Examples include:

- Code artifacts: Notebooks, repos and models, queries (including the model binary and associated feature values)
- Results available to display to users (Job results, Databricks SQL query results and notebook results)
- Customer-created credentials including PATs, Secrets, Git credentials

While a workspace is active, the customer is responsible for managing the retention for most of these assets. Most customers will leave them available in perpetuity as they typically represent intellectual property of the customer. Databricks has APIs available for management where required.

As discussed above in [Disaster recovery](#), while Databricks does perform some backups of this data, customers are responsible for full data availability coverage and should perform their own backups.

## Operational metadata

Operational metadata is the information that Databricks collects during the provision, administration and usage of the service and includes:

- Service configuration data — reflecting configuration of the Databricks platform by customers via the platform-provided interfaces (such as table schemas, and the configuration of accounts, workspaces, identity providers, and similar)
- Product telemetry that consists of metrics and logs of customers' usage of the Databricks platform
- Billing information
- Account information
- Security and audit logs

To the extent that customers need to provide data availability coverage of this data (for example, lists of clusters), these can be pulled automatically via the tools covered in [Disaster recovery](#). Log data, such as cluster logs and security audit logs, can be downloaded into customer storage or third-party log analysis tools, to enable data retention and data availability requirements.

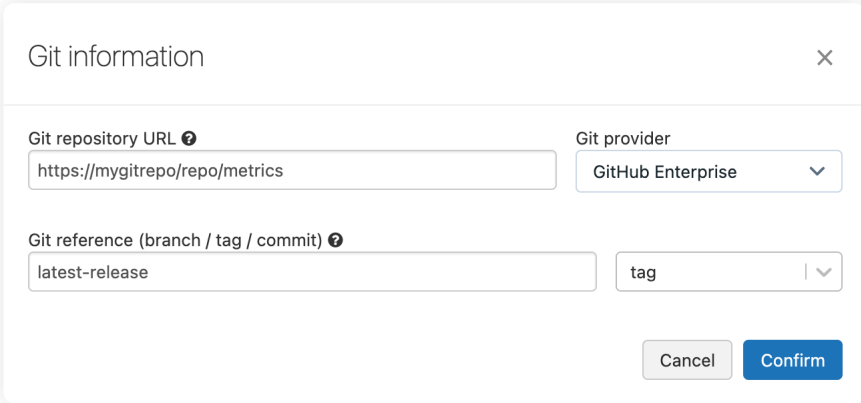
## CHANGE MANAGEMENT

GxP requirements necessitate effective change management. Change management depends on the workloads deployed, and Databricks is well suited to meet customer needs in this area. It is easiest to separate change management into five levels: code, Databricks Runtime, OS and libraries, infrastructure, and control plane.


### Source code change

The element with the most significant impact on GxP workloads is the organization's code. Accessing data, the business logic for parsing or manipulating data, any machine learning models would all fall into an organization's source code. Customers are responsible for managing their code, and advanced customers may leverage automated [CI/CD processes](#).


Customers can leverage the [Repos for Git integration](#) feature, which allows the code running in jobs to be pulled directly from a Git repository. This can be configured to run the code currently active within a given repo, or it can be tagged to a specific branch, tag or commit.



Git information

Git repository URL 

Git provider

Git reference (branch / tag / commit) 

We discuss Databricks responsibilities around change management for Databricks-owned code [above](#), in Databricks corporate responsibilities.

### Databricks Runtime

The Databricks Runtime (DBR) contains the code used to run customer jobs, including the Databricks version of Apache Spark™. When launching a cluster or configuring a job, customers can choose which version of DBR they wish to use. We highly recommend our [Long Term Support \(LTS\)](#) releases to provide a reliable environment for long periods: we declare a “canonical” feature version, for which we provide three full years of support.

Databricks will apply bug fixes within a DBR LTS version, but LTS versions are intended to avoid any behavioral changes. It is against Databricks policy to introduce any breaking changes in the same DBR version (regardless of whether LTS releases are used).

When customers are ready to migrate from an older LTS release to a newer release, they can adjust the job profiles or relaunch the clusters within their development or QA environments to use a different DBR version. If there are issues, they can roll back to an older release. Switching between LTS releases (for example, from 7.3 to 9.1) will often include changes in the Spark version or other behavioral changes.

Changes can be viewed at our [Databricks Runtime Release Notes](#). For example, within the release notes for DBR 11.0, you will find a [migration guide and behavioral changes](#). Within a given DBR version, you can also view the minor bug fixes ([example for DBR 11.0](#)).

### Operating system and libraries

Paired with the Databricks Runtime (DBR) is the operating system and libraries where the DBR runs. Most customers will use the default OS and libraries automatically (and transparently) provided for Databricks workloads. However, in some GxP workloads you may utilize additional software that requires a high level of control over the packages provided (such as specific point releases for an operating system package). In these cases, you can utilize [Databricks Container Services \(DCS\)](#).

If you believe you need to use DCS, a little background is helpful: all user code within the data plane runs within containers, typically one per host system (one container per EC2 host, Azure VM, etc.). For the most part, the container's existence is transparent to the user, as their interactions are all within the container. While customers will almost always leverage the Databricks-provided container, some will want the higher level of control possible by providing their own container.

As detailed in the docs linked above, DCS does shift an aspect of the shared responsibility model as customers become responsible for elements like vulnerability management within the container, and customers are encouraged to thoroughly test their environment to ensure that all requirements for their workloads are met. Fortunately, that is generally assumed for GxP workloads and doesn't represent a significant additional burden.

DCS can be integrated into the customer's CI/CD processes and provided to clusters as a configuration option, allowing customers to change the URL to select a different container for use in dev or QA and roll back containers in production.

### Infrastructure change

The data plane infrastructure includes the networking and host systems (such as EC2 instances, Azure VMs or Google GKS systems) that provide the compute used by Databricks.

There are few material changes to consider at this level. At the networking layer, Databricks can be deployed as a Databricks-managed VPC or as a customer-managed VPC (the latter is much more common for enterprise customers), but this will rarely change after deployment.

Customers can configure the instance types available to be used by clusters through cluster policies, described [above](#). Certain changes could have performance impacts on your workload, such as the number of cores or the number and size of locally attached storage.

### Control plane change

The final element for change management is the Databricks control plane. Customers with GxP-validated workloads typically defer the control plane changes to the Databricks change control process, or add a requirement for system owners to review the [platform release notes](#) for significant changes.

Unlike most PaaS data platforms, the Databricks architecture does not run customer jobs in the control plane. Instead, it is used for tasks like job management, API endpoints, and the UI for users. Actual customer workloads will run in the data plane, covered above under [Databricks Runtime](#).

### VALIDATION

The validation exercise is, of course, essential to GxP-validated workloads. As has been discussed throughout this document, GxP accountability falls on customers, as Databricks itself is not subject to GxP, and so GxP validation is primarily an exercise that the customer is responsible for. Given that, Databricks can provide assistance to customers undertaking a GxP audit. Please work with your account team if you have any questions, or to discuss how best to meet the specific scope of your GxP audit.