ORDO™: Infrastructure and Sub-Processes

Published: May 2025

SCOPE

This documentation describes the infrastructure environment, sub-processors and certain other entities material to the services and managed packages listed in the Infrastructure and Sub-processors Table below (collectively, for the purposes of this document only, the “Covered Services”). Services or features not yet generally available may be included within the list of Covered Services for the purpose of providing Customers advance notice of new sub-processors or processing locations. Any reference to future services or features does not obligate GBS to make those services or features available. Capitalized terms used in this documentation are defined in GBS MSA.

AWS SERVICES

ORDO™ is deployed on an AWS Cloud platform to ensure high availability, security and durability. The application is built on multi-stack programming languages. Below are the critical technical components involved in the development of the application.

- **Lambda**: AWS Lambda is a serverless compute service that lets GBS run code without provisioning or managing servers, creating workload-aware cluster scaling logic, maintaining event integrations, or managing runtimes.
- **DynamoDB**: Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully managed, multi-region, multi-active, durable database with built-in security, backup and restores, and in-memory caching for internet-scale applications.
- **DocumentDB**: Amazon DocumentDB is a fully managed NoSQL database service designed to be compatible with MongoDB. It's ideal for modern, high-availability applications requiring secure, scalable, and low-latency access to data.
- **API Gateway**: Amazon API Gateway is a fully managed service that makes it easy for GBS to create, publish, maintain, monitor, and secure APIs at any scale. APIs act as the "front door" for applications to access data, business logic, or functionality from your backend services.
- **EventBridge**: Amazon EventBridge is a serverless event bus that makes it easy to connect applications using data from GBS applications. And makes it easy to build event-driven applications because it takes care of event ingestion and delivery, security, authorization, and error handling.
- **Simple Storage Service(S3)**: Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.
- **CloudFront**: Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, and high transfer speeds, all within a developer-friendly environment.
- **Simple Email Service(SES)**: Amazon Simple Email Service (SES) is a cost-effective, flexible, and scalable email service that enables us to send mail from within any application. With Amazon SES, GBS can send an email(transaction/promotional) securely, globally, and at scale.
- **Cognito**: Amazon Cognito lets us add user sign-up, sign-in, and access control to our web and mobile apps quickly and easily. Amazon Cognito scales to millions of users and supports sign-in with social identity providers, such as Apple, Facebook, Google, Amazon, and enterprise identity providers via SAML 2.0 and OpenID Connect.

- **Simple Queue Service(SQS)**: Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables GBS to decouple and scale microservices, distributed systems, and serverless applications.
- **CloudWatch**: Amazon CloudWatch is a fully managed monitoring and observability service that provides real-time insights into AWS resources, applications, and services. It enables GBS to track metrics, collect logs, and respond to system-wide performance changes, helping optimize resource utilization and maintain operational health.
- **Elastic Container Service**: ECS is a container orchestration service that allows you to run and scale containerized applications on AWS.
- **Elastic Container Registry**: ECR is a fully managed Docker container registry service provided by AWS. It is used to store, manage, and deploy Docker container images securely.
- **Amazon Translate**: Translate is a fully managed neural machine translation service provided by AWS. It enables developers to easily translate text between different languages in a cost-effective, fast, and high-quality manner.
- **Amazon Textract**: Textract is a machine learning service offered by AWS that automatically extracts text, handwriting, and structured data from scanned documents.

TECHNOLOGY STACK

Application	Technologies
Web Applications	HTML5, CSS3, Javascript, React.js
Backend Application	Node.js and/or Python
Database	DynamoDB and DocumentDB
Cloud Infrastructure	Lambda
Amazon Web Services	EventBridge
	API Gateway
	Cognito
	Translate
	S3
	CloudFront
	CloudWatch
	Simple Email Service
	Secrets Manager
	Textract

ENVIRONMENTS

Environments	Description
Development	It's where coding, initial integration, continuous integration, and automated testing happen. It's primarily used for merging different modules developed by various developers and for developer testing.
Staging	This is where the external stakeholders will get regular updates to review. It's where extensive testing, including performance and security testing, happens prior to pre-production deployment.
Pre-Production	A final testing stage before deploying to production, focusing on thorough validation and user acceptance testing. Also, the team performs security

	testing to detect defects early. This enables teams to drive quality code with a high assurance of stability and security.
Production	The live environment where the stable and fully tested application serves end-users. Deployments will happen through the Auto DevSecOps pipeline based on successful testing in previous stages.

DEVELOPMENT PROCESS

GBS embraces agile project management methodologies to increase their development speed, expand collaboration, and foster the ability to better respond to market trends. We use Scrum as an agile development methodology in the development of Software based on iterative and incremental processes. Scrum is a fast, adaptable, flexible, and effective agile framework that is designed to deliver value to the customer throughout the development of the project. The primary objective of Scrum is to satisfy the customer's need through an environment of transparency in communication, collective responsibility, and continuous progress. The development starts from a general idea of what needs to be built, elaborating a list of characteristics ordered by priority (product backlog) that the owner of the product wants to obtain.

Phases	Activity
Design Phase	Mindmap Creation
	Sketch Creation
	Prototype Design and Design System
	Detailed User Stories
Development Phase	Backlog Refinement and Prioritization
	UX Design and UX Development
	Defining Acceptance Criteria
	Development of Product Increments
	Quality Assurance
	Done
	Release
	Increment Demo

Design Phase

The Design Phase in the software development lifecycle is a crucial stage where the conceptual ideas and requirements for the product are transformed into visual representations and detailed plans. The goal is to create a clear blueprint that guides the development team in building the software according to the intended user experience and functionality. The following are different stages in the design phase:

- **Mindmap Creation**: A mind map is a visual representation of ideas and concepts that are connected to a central theme. In the context of software design, a mind map can be used to brainstorm and organize the various features, functionalities, and components that need to be included in the software. It helps in capturing a holistic view of the product's scope and potential directions.
- **Sketch Creation**: Sketching involves creating rough, hand-drawn visualizations of the software's user interface (UI) and layout. These sketches provide an initial visual representation of how different elements of the software will be arranged on the screen. Sketches are quick and low-fidelity, meant to explore different layout possibilities before committing to a detailed design.
- **Prototype Design and Design System**: Prototyping involves creating a few interactive, clickable mockups of the software's user interface. These prototypes allow stakeholders and designers to

simulate user interactions and workflows, providing a better understanding of the user experience. A design system is a collection of standardized UI components, patterns, and guidelines that ensure consistency and efficiency in design across the software. It helps in maintaining a cohesive and polished user interface.

- **Adding Details to User Stories**: During the Design Phase, the user stories are enriched with more detailed information related to how the feature will work, what interactions are involved, and how the user interface will look. This step helps ensure that the development team has a comprehensive understanding of what needs to be implemented.

The Design Phase serves as a bridge between the product's conceptualization and its actual development. It's where the abstract ideas take on a tangible form, allowing stakeholders, designers, and developers to align their understanding and expectations. The output of this phase provides a clear direction for the subsequent development work and contributes to the overall success of the product.

Development Phase

The development phase transforms the design and planning into tangible, functional software that can be tested, validated, and eventually released to users. The following stages of the development phase collectively contribute to the continuous development and improvement of the software, aligning with agile principles and iterative development practices.

Backlog Refinement and Prioritization: This stage involves reviewing and refining items in the backlog (a list of pending tasks or features). The development team collaborates to ensure that each item is well-defined, estimated, and appropriately prioritized. This helps in preparing the backlog for upcoming sprints or development cycles.

- **UX Design and UX Development**: UX (User Experience) design is the process of creating an optimal and user-friendly interface for the software. This stage involves designing the visual layout, user interactions, and overall user flow. UX development translates these designs into functional UI components, ensuring that the software is intuitive and engaging for users.
- **Defining Acceptance Criteria**: Acceptance criteria are specific conditions that must be met for a user story to be considered complete and "accepted" by the product owner or stakeholders. This stage involves collaboratively defining the criteria that the software must fulfil in order to satisfy the user's needs and meet the product's requirements.
- **Development of Product Increments**: In an agile development approach, work is broken down into small, manageable increments or features. This stage involves the actual coding and development work to create these increments.
- **Quality Assurance**: Quality assurance (QA) involves testing the software to identify defects, bugs, or discrepancies that might affect its functionality or user experience. QA engineers create and execute test cases and perform various types of testing.
- **Done**: In the context of development, "Done" signifies that a user story has been completed from a development perspective. It has passed development, and testing, and meets the predefined acceptance criteria.
- **Release**: The release stage involves packaging and preparing the completed increments for deployment. It includes creating a release build, ensuring all components are integrated correctly, and conducting final testing before the software is deployed to a specific environment.
- **Increment Demo**: After each development iteration, a demo or presentation of the completed increment is conducted for stakeholders, including product owners, management, and end-users.

This demo showcases the new features and functionalities that have been developed in the recent iteration.

GBS incorporates Security Considerations into nearly phase of the Design and Development processes. For further information see:

Document	Description
Secure Development Policy	Aims to set out GBS's policy for developing software applications and components that maximize their inherent security.
Secure Development Practices	Outlines the secure development practices to ensure that security is integrated into every software development lifecycle (SDLC) stage.
Secure Development Change Management Process	Aims to set out GBS's policy for effective development change management.
Secure Development Environment Guidelines	Establishes guidelines for securing the software development environment, ensuring it is protected from unauthorized access, malicious attacks, and unintentional changes.
Secure Implemented Measures	Incorporating security checks in a continuous integration / continuous delivery (CI/CD) pipeline ensures that vulnerabilities are identified and addressed early in the development process, maintaining secure code deployment.

TESTING PRACTICES

We at GBS use a range of methodologies and techniques to evaluate software for defects, quality, and performance, ensuring that it meets requirements and functions as expected. Following are the testing practices we follow:

- **Behaviour Driven Development (BDD)**: An approach that emphasizes collaboration among developers, testers, and domain experts to define and understand the behaviour of a system in plain language, using scenarios structured as "Given-When-Then."
- **Functional and End-to-end Testing using a No-Code Platform**: Conducting functional and end-to-end testing of software using a platform that allows non-technical users to visually design and execute test scenarios without writing traditional code.
- **Performance Testing using Grafana Cloud K6**: Evaluating software performance using Grafana Cloud K6, a tool that simulates user behaviour to measure how the application performs under different loads, identifying performance bottlenecks and latency issues.
- **Session-Based Testing**: An exploratory testing approach where testers conduct testing sessions with defined objectives and timeframes, documenting their actions and findings to facilitate communication and collaboration.
- **Exploratory Testing**: A testing approach where testers actively explore the software, uncovering defects and learning about its behaviour in an unscripted manner, guided by their domain knowledge and intuition.
- **Integration Testing**: Verifying interactions between integrated components, modules, or services within a system to ensure they work together as intended and that data flows accurately between them, identifying potential integration issues.

RELEASE PROCESS

We at GBS embrace auto DevSecOps processes, with developers being tasked with releasing software into production at incredibly high velocity. DevSecOps is a set of automated processes and tools that allows developers and operations professionals to collaborate on building and deploying apps to a production environment.

Our DevSecOps CI/CD pipeline typically includes:

- **ESLint**: ESLint is a tool for identifying and reporting on patterns found in code. It's used to ensure code quality, enforce coding standards, and catch common programming errors.
- **License Compliance**: This stage involves checking the software's dependencies for proper licensing and compliance with the licenses of your project.
- **Dependency Scanning**: OWASP Dependency-Check is a tool that scans project dependencies to identify known vulnerabilities in third-party libraries. This helps ensure that the software is not relying on components with known security issues.
- **Static Application Security Testing (SAST)**: SAST tools like "njsscan" analyze the source code of an application to detect and identify security vulnerabilities, coding errors, and potential security risks without executing the code.
- **Secret Detection**: This involves scanning the codebase for sensitive information like passwords, API keys, and other credentials that might have been accidentally hard-coded.
- **Build**: The build stage involves compiling the code, packaging it into a deployable format, and preparing it for the deployment phase. Build tools like Jenkins, Travis CI, or CircleCI can automate this process.
- **BDD Test Execution**: Behavioral Driven Development (BDD) involves defining application behavior in plain language and then automating tests based on those behaviors. Cucumber is a popular BDD tool, and Istanbul is a code coverage tool often used with it to measure test coverage.
- **Deployment**: This phase involves deploying the tested and built application to various environments. Dynamic Application Security Testing (DAST): DAST tools assess applications from the outside, simulating real-world attacks to identify security vulnerabilities.
- **SiteSpeed**: SiteSpeed is a tool to measure website performance and analyze the speed of web pages. It helps developers optimize the loading times and overall performance of web applications.

GBS practices a “shift left” approach to delivery increments. Shift left is an approach that moves testing to earlier in the software development lifecycle (hence, “shifting left”). If security testing happens when code is ready for production, it can be difficult to go back and correct problems, and it’s often too late to fix problems quickly. This can lead to delayed handoffs, security issues, and silos between security and the rest of the DevOps teams.

Bringing quality and security testing earlier into the development lifecycle is critical. The way to do this is by integrating security testing into deployment pipelines so that code is continually tested. Following are the testing practices GBS follows:

- **Behavior-Driven Development (BDD)**: An approach that emphasizes collaboration among developers, testers, and domain experts to define and understand the behaviour of a system in plain language, using scenarios structured as "Given-When-Then."

- **Functional and End-to-end Testing using a No-Code Platform**: Conducting functional and end-to-end testing of software using a platform that allows non-technical users to visually design and execute test scenarios without writing traditional code.
- **Performance Testing using Grafana Cloud K6**: Evaluating software performance using Grafana Cloud K6, a tool that simulates user behaviour to measure how the application performs under different loads, identifying performance bottlenecks and latency issues.
- **Session-Based Testing**: An exploratory testing approach where testers conduct testing sessions with defined objectives and timeframes, documenting their actions and findings to facilitate communication and collaboration.
- **Exploratory Testing**: A testing approach where testers actively explore the software, uncovering defects and learning about its behaviour in an unscripted manner, guided by their domain knowledge and intuition.
- **Integration Testing**: Verifying interactions between integrated components, modules, or services within a system to ensure they work together as intended and that data flows accurately between them, identifying potential integration issues.