Informe de Ambientes

Isabella carrera cabrera
Juan Manuel Gutierrez
Felipe Tovar

1. AMBIENTES

El proyecto Codexy se organiza a través de cuatro ambientes que permiten controlar el proceso de desarrollo, prueba y despliegue de todos los módulos del sistema. Esta estructura garantiza que cada cambio pase por un proceso de revisión progresivo antes de llegar a los usuarios finales. Los ambientes son los mismos para los cuatro repositorios principales de la plataforma (codexy-api, codexy-portal, codexy-app y codexy-docs), lo que asegura consistencia entre el backend, el portal administrativo, la aplicación móvil y la documentación oficial.

El primer ambiente es **Development**, que corresponde al espacio donde se integran las nuevas funcionalidades y donde se realiza la mayor parte del trabajo técnico diario. Su rama principal es *develop*, y allí los desarrolladores incorporan el código resultante de las historias de usuario. En este ambiente se realizan ajustes, correcciones rápidas y pruebas iniciales. Es un entorno dinámico, con cambios frecuentes y donde se consolidan las primeras versiones de cada funcionalidad.

Una vez se completa una funcionalidad y supera las pruebas básicas de desarrollo, pasa al ambiente de **QA**, representado por la rama *qa*. Este entorno está destinado exclusivamente a pruebas funcionales y de regresión. Aquí se verifica que cada historia de usuario cumpla correctamente con sus criterios de aceptación y que no se presenten efectos colaterales en otras áreas del sistema. QA actúa como un filtro de calidad, permitiendo detectar errores antes de que el sistema avance a etapas más estables.

Luego de superar las validaciones en QA, el proyecto se traslada al ambiente de **Staging**, asociado a la rama *staging*. Este espacio funciona como una preproducción y permite simular el comportamiento real del sistema en condiciones muy similares a las de operación. En Staging se realizan pruebas de aceptación por parte del cliente, validaciones finales y revisiones generales del comportamiento integral del sistema. Este ambiente sirve para confirmar que todo lo construido e integrado funciona correctamente antes de ser expuesto al usuario final.

Finalmente, el último ambiente es **Producción**, vinculado a la rama *main*. Este es el entorno donde se encuentra la versión estable y oficial del sistema Codexy. Aquí se despliega únicamente el código que ha superado todas las etapas previas. En este ambiente operan los usuarios finales, por lo que se prioriza la estabilidad, la disponibilidad y la seguridad. Cualquier cambio que llegue a Producción ha sido previamente validado, probado y revisado en los ambientes anteriores, lo que minimiza riesgos y garantiza una operación confiable.

Cada historia de usuario sigue este recorrido de forma ordenada, avanzando desde Development hacia QA, luego a Staging y finalmente a Producción. Este flujo permite mantener control sobre las versiones, facilita la trazabilidad de los cambios y evita que funcionalidades incompletas o inestables lleguen al entorno final. Gracias a esta organización por ambientes, todos los componentes de Codexy —incluyendo la API, el Portal Web y la aplicación móvil— mantienen coherencia entre sí y pueden evolucionar de manera controlada, segura y predecible.

2. ARQUITECTURA DEL PROYECTO

El proyecto Codexy está construido sobre una arquitectura modular que permite mantener independencia entre componentes, facilitar el mantenimiento y asegurar una correcta escalabilidad del sistema a futuro. La plataforma se divide en varias capas y servicios, cada uno con responsabilidades claramente definidas para evitar acoplamientos innecesarios. En su núcleo se encuentra el backend desarrollado en .NET, el cual actúa como proveedor central de datos, lógica de negocio y servicios transversales. Este componente concentra todo el manejo de entidades, seguridad, autenticación, validaciones y comunicación con la base de datos.

El portal administrativo, desarrollado en Angular, consume estos servicios mediante API y se encarga de la interacción con los usuarios encargados de la gestión, supervisión y verificación dentro del sistema Codexy. Su arquitectura está orientada a componentes y permite organizar la interfaz en módulos independientes, como inventarios, usuarios, reportes y administración. Por su parte, la aplicación móvil, desarrollada en Ionic/Capacitor, está diseñada para los operativos en campo, ofreciendo funcionalidades de escaneo QR, registro de elementos, revisión de inventarios y sincronización con el backend. Finalmente, el proyecto de documentación consolida todos los lineamientos técnicos, manuales y registros que soportan el funcionamiento integral de la plataforma.

En conjunto, esta arquitectura permite que cada módulo del sistema evolucione de manera independiente, manteniendo una comunicación constante y estable a través del backend. De esta forma, Codexy garantiza coherencia en su comportamiento, facilita la integración continua y habilita un crecimiento ordenado de nuevas funcionalidades sin comprometer la estabilidad del sistema.

3. REPOSITORIOS DEL ECOSISTEMA CODEXY

El ecosistema Codexy está conformado por varios repositorios independientes, cada uno orientado a un propósito específico dentro de la plataforma. El repositorio denominado *codexy-api* corresponde al backend y es el encargado de la lógica de negocio, los controladores, la seguridad y la conexión directa con la base de datos. Desde este proyecto se centralizan los servicios que consumen tanto la aplicación web como la aplicación móvil.

El repositorio *codexy-portal* contiene el desarrollo del panel administrativo en Angular. Aquí se gestionan las vistas, componentes, servicios y módulos asociados a la administración del sistema. Es el entorno utilizado por los responsables de validar inventarios, consultar reportes y gestionar usuarios. Por otro lado, el repositorio *codexy-app* corresponde a la aplicación móvil desarrollada en Ionic, enfocada en los operativos que realizan escaneo de códigos QR, registro de elementos y verificaciones en campo. Este proyecto integra servicios optimizados para dispositivos móviles y mantiene sincronización constante con la API.

Además, existe el repositorio *codexy-docs*, el cual centraliza la documentación técnica, funcional y administrativa del sistema. Allí se construyen manuales, instructivos, diagramas y cualquier recurso necesario para la operación y soporte del proyecto. Finalmente, el repositorio *codexy-db* contiene los scripts relacionados con la estructura de la base de datos, definiciones de tablas, procedimientos y cualquier ajuste requerido para la evolución del modelo de datos. En conjunto, estos repositorios permiten que el ecosistema se mantenga organizado, mantenible y altamente escalable.

4. FLUJO DE VERSIONAMIENTO Y RAMAS GIT

El proyecto implementa una estrategia de control de versiones basada en un flujo de ramas que permite organizar el desarrollo de manera progresiva y ordenada. Este modelo garantiza que cada cambio pase por un conjunto de revisiones antes de ser incorporado a la versión final del sistema. Las ramas principales utilizadas son *develop*, *qa*, *staging* y *main*, cada una asociada a un ambiente específico del proyecto.

El proceso inicia en la rama *develop*, donde se integran las nuevas funcionalidades provenientes de las historias de usuario. Cada historia genera una rama temporal destinada a su desarrollo independiente, la cual es fusionada en *develop* una vez finalizada. Posteriormente, el código es promovido hacia la rama *qa*, donde se llevan a cabo pruebas funcionales y de regresión. Este nivel actúa como un filtro que valida el comportamiento correcto de los cambios introducidos.

Después de superar la etapa de QA, los cambios se integran en la rama *staging*, que funciona como un entorno de preproducción. En esta fase se realizan pruebas de aceptación por parte del cliente, validaciones finales y revisiones completas del sistema. Solo los cambios que han pasado por todas estas etapas son finalmente incorporados a la rama *main*, la cual representa la versión estable y operativa del sistema. Este flujo de trabajo proporciona control, trazabilidad y seguridad sobre cada versión generada.