



VERSIÓN 1.0

27 DE NOVIEMBRE DE 2025

Ambiente de desarrollo: herramientas seleccionadas y flujo de trabajo

ESCUELA POLITECNICA NACIONAL – FACULTAD DE INGENIERIA DE SISTEMAS
CONSTRUCCIÓN Y EVOLUCIÓN DE SOFTWARE
[PROYECTO-TRANSCRIPTOR-BRAILLE]



CONTENIDO

1.	INTRODUCCIÓN	2
2.	AMBIENTE DE DESARROLLO	2
2.1.	Arquitectura del sistema	2
2.2.	Backend – FastAPI	2
2.3.	Frontend – React con TypeScript	2
3.	FLUJO DE TRABAJO - GITFLOW WORKFLOW	3
3.1.	Ramas principales.....	3
3.2.	Ramas de soporte.....	3
4.	CONCLUSIONES	3

1. INTRODUCCIÓN

El presente documento tiene como objetivo detallar la configuración del ambiente de desarrollo y el flujo de trabajo implementado en el proyecto. Se describen las tecnologías utilizadas, las dependencias del sistema y la metodología de control de versiones adoptada por el equipo.

2. AMBIENTE DE DESARROLLO

2.1. Arquitectura del sistema

El proyecto implementa una arquitectura cliente-servidor con separación clara entre frontend y backend:

- **Backend:** FastAPI (Python)
- **Frontend:** React con TypeScript

2.2. Backend – FastAPI

FastAPI es un framework moderno y de alto rendimiento para construir APIs con Python 3.7+. Se caracteriza por su velocidad, facilidad de uso y generación automática de documentación.

El ambiente de backend requiere las siguientes dependencias principales:

Dependencia	Versión Mínima	Propósito
fastapi	0.100.0	Framework principal para la API REST
uvicorn	0.20.0	Servidor ASGI de alto rendimiento
pydantic	2.0.0	Validación de datos y serialización
pytest	7.0.0	Framework de testing
httpx	0.24.0	Cliente HTTP asíncrono para testing
Pillow	10.0.0	Procesamiento de imágenes
reportlab	4.0.0	Generación de documentos PDF
pypdf	3.0.0	Manipulación de archivos PDF

2.3. Frontend – React con TypeScript

El frontend está desarrollado con React y TypeScript, proporcionando una experiencia de usuario moderna con tipado estático para mayor robustez y mantenibilidad del código.

Características Técnicas del Frontend

- **TypeScript:** Tipado estático para prevención de errores en tiempo de desarrollo
- **React:** Biblioteca para construcción de interfaces de usuario basadas en componentes
- **Component-Based Architecture:** Arquitectura modular y reutilizable
- **Type Safety:** Validación de tipos en tiempo de compilación

3. FLUJO DE TRABAJO - GITFLOW WORKFLOW

El proyecto implementa Gitflow Workflow, una estrategia de branching que utiliza ramas específicas para diferentes propósitos en el ciclo de desarrollo. Esta metodología facilita la gestión de versiones y el trabajo colaborativo del equipo.

3.1. Ramas principales

main (principal)

- Contiene el código en producción.
- Representa el historial oficial de releases.
- Todos los commits deben estar etiquetados con números de versión.
- Solo se actualiza mediante merges desde ramas reléase/develop.

develop (desarrollo)

- Rama de integración para nuevas características.
- Contiene el historial completo del proyecto.
- Base para la creación de ramas feature.
- Refleja el estado actual del desarrollo.

documentacion (desarrollo)

- Rama destinada exclusivamente a la elaboración y mantenimiento de documentación técnica.
- Permite trabajar los entregables documentales sin afectar el código fuente.
- Facilita el seguimiento y control de versiones de documentos.
- Se sincroniza con *develop* cuando se requiere asociar documentación a nuevas funcionalidades.

3.2. Ramas de soporte

feature/ (características)

- Propósito: Desarrollo de nuevas funcionalidades.
- Origen: develop.
- Destino: develop.
- Nomenclatura: feature/nombre-funcionalidad.
- Ciclo de vida: Se elimina después del merge.

4. CONCLUSIONES

El ambiente de desarrollo configurado con FastAPI y React+TypeScript, junto con la implementación de Gitflow Workflow, proporciona una base sólida para el desarrollo colaborativo y estructurado del proyecto. Esta configuración permite:

- **Desarrollo eficiente:** Herramientas modernas y bien documentadas
- **Escalabilidad:** Arquitectura que facilita el crecimiento del proyecto
- **Calidad:** Tipado estático y testing integrado
- **Colaboración:** Flujo de trabajo que soporta equipos de cualquier tamaño
- **Gestión de versiones:** Control claro sobre releases y correcciones

La adopción de estas tecnologías y metodologías posiciona al proyecto para un desarrollo profesional y mantenible a largo plazo.