ACTIVIDAD 5: Estrategia de Control de Calidad y Pruebas Unitarias

La calidad de un proyecto de software depende tanto de un diseño sólido como de un conjunto de prácticas que garanticen que cada componente funciona según lo previsto. Para ello, es fundamental contar con una estrategia de pruebas y control de calidad que abarque desde la verificación de partes específicas del código hasta el modo en que los distintos módulos se comunican entre sí.

Las pruebas unitarias constituyen la base de esta estrategia. Cada clase o método clave del sistema se evalúa de forma aislada para verificar que sus resultados coincidan con lo que se espera. De este modo, se detectan errores desde etapas tempranas y se evita que surjan regresiones cuando se añaden nuevas funcionalidades. Además, es importante que estas pruebas se mantengan independientes y sean rápidas de ejecutar, de modo que puedan correr de forma continua sin ralentizar el desarrollo.

Para mantener una supervisión constante de la calidad, se recomienda configurar un entorno de Integración Continua (CI). Con este enfoque, cada vez que se realiza un cambio en el repositorio, las pruebas unitarias se ejecutan de manera automática. Además, se generan informes de cobertura de código, que muestran hasta qué punto las pruebas están abarcando el comportamiento del sistema. Este mecanismo también puede incluir validaciones de estilo y verificaciones estáticas, asegurando que el código cumpla con los estándares de formato y buenas prácticas.

Aunque las pruebas unitarias son cruciales, no bastan por sí solas para confirmar que el sistema funciona como un todo. Por ello, resulta valioso incluir pruebas de integración, donde se comprueba que servicios como TaskService, PredictionService o ReportService colaboren correctamente. Asimismo, unas pruebas de aceptación resultan útiles para simular la experiencia del usuario o de la capa de presentación al recorrer distintos flujos de trabajo. En caso de que existan componentes externos, como bases de datos o servicios remotos, es común recurrir a mocks y stubs para reproducirlos y no depender de entornos que puedan fallar o cambiar.

La estrategia de control de calidad también se vincula de forma directa con la deuda técnica. Cuando el equipo detecta que ciertas partes del código se están volviendo demasiado complejas, las pruebas facilitan la tarea de refactorizar. Con una suite de pruebas sólida, es posible reestructurar el código con la confianza de no afectar funcionalidades ya existentes. Estas pruebas también documentan de manera implícita

el comportamiento esperado, sirviendo como guía para nuevos desarrolladores y brindando más estabilidad al proyecto.

En conclusión, una estrategia de control de calidad basada en pruebas unitarias, integraciones de CI y otros tipos de validaciones conjuga la solidez de la arquitectura y la mantenibilidad del código. Con la adopción de pruebas en todos los niveles y la práctica de un monitoreo continuo, el equipo asegura que cada incremento de software cumpla los requisitos y preserve la fiabilidad del sistema a largo plazo.