Actividad 5: Estrategia de Control de Calidad y Pruebas Unitarias

Garantizar la calidad de nuestro software va a ser esencial para mantener las características principales que estamos buscando, su escalabilidad y mantenibilidad. Para ello usaremos una estrategia de pruebas unitarias, y pruebas de integración y validaciones automatizadas.

Pruebas Unitarias:

Las pruebas unitarias verificaran el correcto funcionamiento de los componentes individuales del código. Usaremos Junit como framework de pruebas. Un ejemplo de una prueba unitaria para la clase *FabricaTareas* sería el siguiente:

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;
import plataforma.tareas.*;

class FabricaTareasTest {
   @Test
   void testCrearTarea() {
      FabricaTareas fabrica = new FabricaTareasSimples();
      Tarea tarea = fabrica.crearTarea("Diseñar UI");
      assertNotNull(tarea);
      assertEquals("Pendiente", tarea.getEstado());
   }
}
```

Pruebas de Integración:

Las pruebas de integración validan la comunicación entre los módulos del programa, lo que es especialmente importante en la integración con herramientas externas mediante el patrón Adapter. Un ejemplo de una prueba de integración seria simular respuestas de los servicios de terceros como Trello o Google Drive y verificar que los módulos de *integración* obtienen los datos correctamente y no

lanzan excepciones. Las pruebas de integración estarán integradas en el pipeline CI/CD mediante GitHub Actions.

Con estas medidas, veremos varios beneficios. Reduciremos la deuda técnica, ya que podremos detectar los errores desde las primeras fases del desarrollo, minimizando problemas a largo plazo. Cada modificación en el código se validará automáticamente, lo cual facilitará el mantenimiento de este y asegurará su calidad.