13:49

AeropuertoControl.java

```java
import java.util.Random;
import java.util.concurrent.*;

public class AeropuertoControl {

    private final ExecutorService executor = Executors.newFixedThreadPool(4);
    private final Random random = new Random();

    public Future<Boolean> verificarPista() {
        return executor.submit(new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(2000 + random.nextInt(1000)); // 2-3 s
                boolean resultado = random.nextDouble() < 0.80;
                System.out.println("🛬 Pista disponible: " + resultado);
                return resultado;
            }
        });
    }

    public Future<Boolean> verificarClima() {
        return executor.submit(new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(2000 + random.nextInt(1000)); // 2-3 s
                boolean resultado = random.nextDouble() < 0.85;
                System.out.println("🌦️ Clima favorable: " + resultado);
                return resultado;
```

Home End ( ) { } < > ' " ;

13:49

AeropuertoControl.java

```
            }
        });
    }

    public Future<Boolean> verificarTraficoAereo() {
        return executor.submit(new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(2000 + random.nextInt(1000)); // 2-3 s
                boolean resultado = random.nextDouble() < 0.90;
                System.out.println("🚦 Tráfico aéreo despejado: " + resultado);
                return resultado;
            }
        });
    }

    public Future<Boolean> verificarPersonalTierra() {
        return executor.submit(new Callable<Boolean>() {
            public Boolean call() throws Exception {
                Thread.sleep(2000 + random.nextInt(1000)); // 2-3 s
                boolean resultado = random.nextDouble() < 0.95;
                System.out.println("👷 Personal disponible: " + resultado);
                return resultado;
            }
        });
    }

    public void gestionarAterrizaje() {
```

Home End ( ) { } < > ' " ;

13:49

AeropuertoControl.java

```
53    public void gestionarAterrizaje() {
54        System.out.println("🛬 Verificando condiciones para aterrizaje...\n");
55
56        Future<Boolean> pistaFuture = verificarPista();
57        Future<Boolean> climaFuture = verificarClima();
58        Future<Boolean> traficoFuture = verificarTraficoAereo();
59        Future<Boolean> personalFuture = verificarPersonalTierra();
60
61        executor.submit(new Runnable() {
62            public void run() {
63                boolean pista = false, clima = false, trafico = false, personal = false;
64
65                try {
66                    pista = pistaFuture.get();
67                } catch (Exception e) {
68                    System.out.println("❌ Error verificando pista.");
69                }
70
71                try {
72                    clima = climaFuture.get();
73                } catch (Exception e) {
74                    System.out.println("❌ Error verificando clima.");
75                }
76
77                try {
78                    trafico = traficoFuture.get();
```

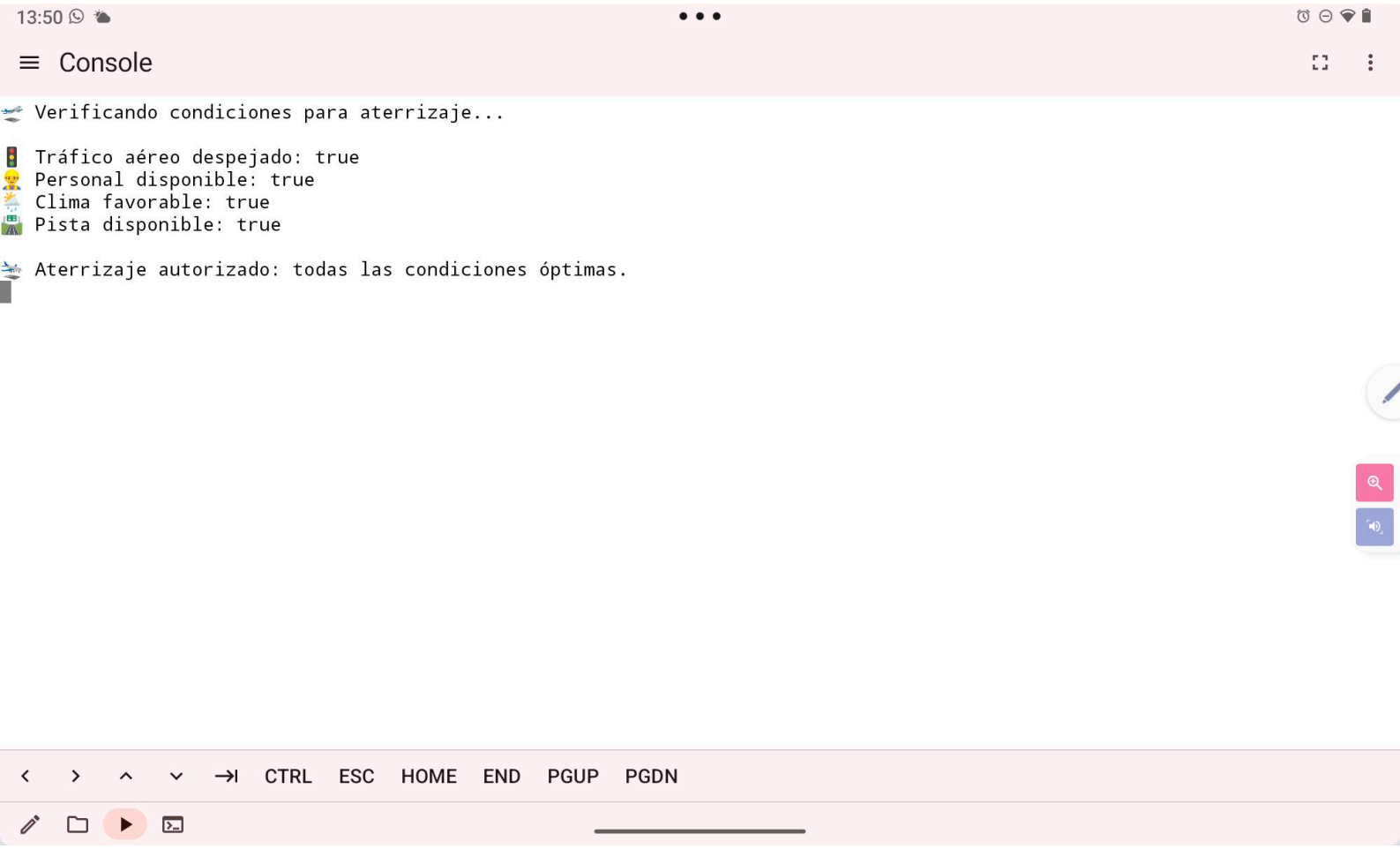
Home End ( ) { } < > ' " ;

13:49

AeropuertoControl.java

```
                } catch (Exception e) {
                    System.out.println("❌ Error verificando tráfico aéreo.");
                }

                try {
                    personal = personalFuture.get();
                } catch (Exception e) {
                    System.out.println("❌ Error verificando personal en tierra.");
                }

                if (pista && clima && trafico && personal) {
                    System.out.println("\n🛬 Aterrizaje autorizado: todas las condiciones óptimas.");
                } else {
                    System.out.println("\n🚫 Aterrizaje denegado: condiciones no óptimas.");
                }

                executor.shutdown();
            }
        });
    }

    public static void main(String[] args) {
        new AeropuertoControl().gestionarAterrizaje();
    }
}

```

Home End ( ) { } < > ' " ;


13:50
Console
Verificando condiciones para aterrizaje...
Tráfico aéreo despejado: true
Personal disponible: true
Clima favorable: true
Pista disponible: true
Aterrizaje autorizado: todas las condiciones óptimas.
CTRL
ESC
HOME
END
PGUP
PGDN