Obligatorio 2025 DDA – Ingeniería de Software

Objetivo General:

Desarrollar una versión escalable del juego **Tutti-Frutti**.

Etapas del Proyecto:

1) Versión para un jugador:

Desarrollar una versión básica del juego **Tutti-Frutti** que permita a un jugador jugar en solitario.

- **Setup:** el jugador elige las **categorías** (p.ej., Animal, País, Color…). El sistema **sortea una letra**. Se fija un **tiempo máximo** (ej. 60 s)
- **Pantalla de juego:** inputs de texto (uno por categoría), **timer** visible y dos botones:
    - **Tutti Frutti**: habilitado **solo si todos los campos están completos** y empiezan con la letra sorteada. Al presionarlo, **termina la partida** y se envían las respuestas a validación.
    - **Me rindo**: cierra la partida **sin completar** y envía lo que haya (vacíos cuentan como 0).
- **Cierre automático:** si se acaba el tiempo, se cierra igual y se valida lo escrito.
- **Validación (Juez IA):** chequea letra inicial, existencia y pertenencia a categoría. Muestra **resultado** por categoría (válida/duplicada no aplica en single, inválida/vacía) y **puntaje total**.
- **Post-partida:** mostrar puntaje y opción de **reintentar** (nuevas categorías o misma configuración).

2) Versión mujtijugador

Lobby: un jugador crea **sala** (elige categorías, tiempo). Otros se **unen**. Cuando todos están listos, el servidor **sortea la letra** y arranca el **timer centralizado**.

Pantalla de juego (cada jugador): mismos inputs + timer + botones:

- **Tutti Frutti**: habilitado solo si completó **todo válido por letra**. Al presionarlo, **finaliza la partida para todos** (se da una gracia mínima opcional, p.ej. 5–10 s, o final inmediato según regla que definas).

- **Me rindo**: ese jugador **cierra su participación** (sus faltantes quedan en blanco). La partida **sigue** para los demás hasta que alguien haga Tutti Frutti o se termine el tiempo.

Desconexiones: si un jugador se cae, su fila queda como **no respondida**; la sala continúa.

Validación (Juez IA): letra, existencia, pertenencia; además **duplicados entre jugadores** (válida-única vs válida-duplicada). Puntajes por jugador.

NOTAS DE IMPLEMENTACIÓN

- Debe haber una configuración del sistema en base a parámetros de juego (duración de partida, tiempo de gracia tras Tutti Frutti, categorías activas, puntos por palabra válida única, puntos por palabra válida duplicada)

- El botón **Tutti Frutti** debe estar deshabilitado hasta que **todas las celdas** cumplan mínimos (no vacías y empiezan por la letra).
- **Me rindo** está siempre disponible.
  El **timer** manda: al cero, se envía lo que haya.
  El **Juez** debe ser intercambiable (Strategy) y registrar motivos (logs) de cada veredicto.

Requisitos Generales:

Realizar un análisis completo del sistema, considerando las etapas del proyecto. Incluyendo la investigación de las tecnologías necesarias para desarrollarlo en su totalidad (Sockets, RMI, Frameworks, etc)

La documentación debe incluir diagramas, descripción de la lógica del juego, diseño de la interfaz y estructura de datos. Se complementará la documentación del proyecto con la requerida en la materia Ingeniería de Software.

Aunque se espera un análisis completo del sistema, para todas las etapas, entendemos que, debido a restricciones de tiempo o complejidad técnica, es posible que no logren el 100% de los requerimientos. Sin embargo, esperamos ver un progreso significativo y coherente hacia ese objetivo.

El código fuente debe estar bien organizado, comentado y seguir los principios SOLID y GRASP.

Se valorará especialmente el uso de patrones de diseño vistos en clase, de pruebas unitarias y de integración para garantizar la calidad del software.

Tareas a realizar para: Ingeniería de Software

Un documento que contenga secciones con detalle de:

Planificación:
Documento de Project (ProjectLibre, Microsoft Project, o cualquier otro sistema de gestión de proyectos), con planificación, tareas y sus dependencias, recursos, costos y otra información que considere relevante para su solución.
Detalle de ciclo de vida y metodología seleccionada. Reflejar cómo aplicaron la metodología y que actividades realizaron, ejemplo: reuniones de análisis de requerimientos.
Detalle de metodología de estimación y resultados obtenidos de su aplicación.

Análisis de requerimientos:
Documento ESRE con todos los requerimientos y su completo detalle (los requerimientos deben cumplir con las características de requerimientos).
Casos de uso para cada uno de los requerimientos del sistema.

Análisis de la solución:
Diagramas (dominio completo, y por otro lado podrá seleccionar realizar todos los de actividad o secuencia) para todas las funcionalidades del sistema.

Calidad y Gestión de la configuración:
SCM con detalle de cómo realiza el manejo de versionado del sistema y de la documentación generada.
Plan de pruebas del sistema y detalle de su ejecución con evidencias de sus resultados (las evidencias pueden ir en documento anexo).

Potenciales herramientas para la mejora de la calidad:
Análisis de posibilidad de utilización y beneficios/debilidades que percibe para las herramientas:

1- Selenium
2- SonarQube
3- Jira

Entregas:

Primera Entrega (Análisis de todo el proyecto): Hasta el 1/11
Entrega Final: XXX