ESTÁNDAR DE PROCESO DE DESARROLLO 24-25 IS2

Grupo 16

Integrantes:

BECERRA TAPIA ALEJANDRO
GARCIA-CARO BARTOLOME ALVARO
MARIN MITE ALFONSO
MENENDEZ TREJO RODRIGO
PEREZ LOPEZ LUIS

Etiquetado	Modificación	Petición Cambio
EPD_G16_v1.0	Entrega inicial	PC1

ÍNDICE

1. Introducción

Este documento ofrece una guía clara para organizar, llevar a cabo, supervisar y revisar las tareas del desarrollo del proyecto. Su propósito es garantizar que el trabajo se realice con calidad y que los miembros del equipo colaboren de manera efectiva para lograr los objetivos propuestos. Este estándar se aplica a todas las fases del proyecto, comenzando por definir los requisitos y terminando con las evaluaciones finales después de completar el trabajo. Está diseñado para adaptarse a los modelos de desarrollo que utilizaremos (Incremental-Cascada), lo que nos da flexibilidad para ajustarlo a lo que necesite el proyecto. Siguiendo este estándar, aseguramos que el trabajo se haga de forma ordenada y que el resultado final cumpla con lo que espera el cliente.

2. Definición y roles

Para la realización del proyecto se han definido 5 roles, asignados a los 5 integrantes del grupo. Se dividen entre:

- **Líder**: encargado de la coordinación del equipo, organización de reuniones, comunicación interna y externa, y supervisión general del proyecto. Además, gestiona la planificación, estimación, monitorización y la calidad del proyecto.
- **Responsable de desarrollo**: define y supervisa el entorno de desarrollo, incluyendo lenguajes de programación, herramientas y actividades por fase. También garantiza la entrega de los productos correspondientes a cada actividad.
- **Responsable de calidad**: coordina y asegura el cumplimiento del plan de calidad, verificando que los productos y procesos cumplan con los estándares establecidos.
- **Responsable de gestión**: planifica, organiza y monitorea el calendario del proyecto, asegurando que se cumplan los plazos y las estimaciones establecidas.
- **Responsable de soporte**: administra el plan de gestión de configuración software (GCS), supervisa los cambios realizados al proyecto y coordina el seguimiento del plan de GCS.

3. Reuniones y Comunicación

El proceso de reuniones del grupo es muy sencillo: nos intentamos conectar todos los que podemos los sábados y lunes a las 12 para realizar las tareas propuestas y/o solventar dudas de ellas, siguiendo todos los puntos de la agenda proporcionada por el líder. En el enlace adjunto se encuentra la plantilla utilizada.

4. Modelo de Ciclo de Vida

El ciclo de vida adoptado para este estándar combina dos enfoques bien estructurados: **incremental-cascada para avanzar entre ciclos** y **cascada para el desarrollo interno de cada ciclo**. Este modelo garantiza tanto la organización secuencial como la posibilidad de entregar valor de forma incremental a lo largo del proyecto. A continuación, se detalla cómo funciona:

Narración del Ciclo de Vida

1. **Enfoque Incremental-Cascada para Avanzar entre Ciclos**:
    - El desarrollo se organiza en **múltiples ciclos**, cada uno con un conjunto definido de funcionalidades o entregables.
    - Cada ciclo sigue un flujo bien estructurado que abarca desde la planificación hasta la entrega.
    - Una vez que un ciclo termina, los resultados se integran y sirven como base para el siguiente. Este enfoque incremental permite dividir el alcance del proyecto en bloques manejables, entregando valor continuamente y adaptándose a cambios o retroalimentación entre ciclos.
2. **Enfoque Cascada para el Desarrollo Interno de Cada Ciclo**:
    - Dentro de cada ciclo, se utiliza un modelo **cascada tradicional** para guiar el desarrollo. Esto significa que las fases se siguen de forma secuencial, garantizando claridad y control en cada etapa:
        - **Requisitos:** Se recopilan y documentan exhaustivamente las necesidades del ciclo. Estas se consolidan en una Especificación de Requisitos.
        - **Diseño:** Se realizan los diseños de alto y bajo nivel para estructurar el desarrollo.
        - **Codificación:** Los desarrolladores implementan el código según los diseños y estándares predefinidos.
        - **Pruebas Unitarias:** Se evalúa la calidad y funcionalidad de cada componente desarrollado.
        - **Post-Mortem:** Al final del ciclo, se realiza un análisis retrospectivo para identificar lecciones aprendidas y proponer mejoras.
3. **Sinergia entre Ciclos y Cascadas**:
    - Al final de cada ciclo, se realiza una **evaluación integrada** de los productos y procesos para garantizar que están listos para avanzar.
    - Las lecciones aprendidas en el post-mortem de un ciclo alimentan el planeamiento del siguiente, lo que fomenta un proceso iterativo de mejora continua.

Ventajas de esta Combinación

Flexibilidad Incremental: Permite ajustar el enfoque entre ciclos en función de las necesidades emergentes del proyecto.

Control Estricto en Cascada: Dentro de cada ciclo, las fases bien definidas aseguran una mayor calidad y consistencia.

Entrega Continua: A través de ciclos iterativos, se asegura la entrega de valor en cada etapa del proyecto.

Dada esta estructura hemos basado nuestro proyecto debido que al tener el enunciado con los requisitos, funcionalidades y documentación bien detallada a entregar, se han podido planificar las actividades a realizar para el correcto desarrollo del trabajo, dando esto al uso de ciclo de vida cascada. Como el proyecto se divide en dos ciclos, se usa el ciclo de vida incremental ya que los resultados obtenidos se integran y sirven de base para seguir con el desarrollo del proyecto.

5. Entorno de Desarrollo

En el proyecto se han empleado diversas herramientas y tecnologías en el entorno de desarrollo con el objetivo de garantizar la calidad, eficiencia y una continua revisión de las implementaciones realizadas. Estas herramientas han permitido optimizar tanto el trabajo en equipo como la funcionalidad de la aplicación desarrollada.

Herramientas y Tecnologías Implementadas

1. **Firebase**
   Se ha utilizado Firebase como la plataforma principal para la gestión del trabajo y los recursos relacionados con los usuarios y administradores de la aplicación. Esta herramienta nos ha permitido implementar un sistema robusto de autenticación, asegurando tanto la seguridad de los datos como la facilidad de uso para los usuarios finales. Además, Firebase ofrece una base de datos en tiempo real que se emplea para almacenar, organizar y gestionar los datos generados por la aplicación. Este enfoque garantiza una experiencia fluida para los usuarios al manejar actualizaciones en tiempo real de la información almacenada.
2. **Flutter**
   El desarrollo de la aplicación se ha realizado utilizando **Flutter**, un framework que permite crear interfaces de usuario modernas, intuitivas y funcionales. Flutter ha sido fundamental para el desarrollo multiplataforma, ya que con una única base de código es posible desplegar aplicaciones en diferentes plataformas (iOS, Android y Web). Este framework ofrece una amplia gama de widgets y herramientas que facilitan la creación de experiencias visuales y de usuario consistentes.
3. **Visual Studio Code**
   El entorno de desarrollo integrado (IDE) seleccionado ha sido **Visual Studio Code**, el cual ofrece soporte extensivo para Flutter mediante una amplia variedad de extensiones y librerías. Este IDE no solo proporciona herramientas avanzadas de

depuración y documentación, sino que también permite compilar la aplicación de manera sencilla y visualizarla en diferentes formatos, incluido el formato web. Su versatilidad y facilidad de uso han sido determinantes para acelerar el desarrollo y resolver problemas de manera eficiente.

4. **GitHub y GitHub Desktop**
   Para el control de versiones y la colaboración entre los integrantes del equipo, se han utilizado **GitHub** y su herramienta complementaria **GitHub Desktop**. Estas herramientas han permitido gestionar el proyecto de manera organizada, utilizando un flujo de trabajo basado en ramas. Cada integrante ha podido desarrollar nuevas características o solucionar problemas en ramas independientes, evitando conflictos con el trabajo de otros miembros del equipo. Posteriormente, las contribuciones se han integrado al repositorio principal (main) tras un proceso de revisión, asegurando que cada implementación cumple con los requisitos establecidos y está lista para avanzar en el ciclo de desarrollo.

Impacto en el Desarrollo

El uso combinado de estas herramientas ha permitido consolidar un entorno de desarrollo ágil y colaborativo. La integración de Firebase garantiza una gestión segura y eficiente de los datos en tiempo real, mientras que Flutter proporciona una experiencia de desarrollo fluida con resultados modernos y funcionales. Visual Studio Code ha potenciado la productividad, y GitHub ha asegurado la organización y sincronización del trabajo en equipo, reduciendo errores y maximizando la eficiencia.

6. Fases del Proceso de Desarrollo

Desglose de todas las actividades a realizar para el desarrollo del proyecto, ciclo 1

Reunión inicial
Asignación de roles
Resumen ejecutivo
Lectura y comprensión del proyecto
Planificación conjunta de realización de tareas
Realizar estimación de tiempos de tareas
Establecimiento de fechas limite de realización de tareas
Reparto de tareas
Establecimiento de tareas ciclo 1
Realización de requisitos (ERS)
Realización de ficheros logicos internos (ILF)
Realización de ficheros lógicos externos (EIF)
Realización de consultas externas (EQ)
Realización de Entradas externas (EI)
Revisión de requisitos I

Cálculo de puntos de función
Selección de requisitos a desarrollar
Realización de Gestión de Configuración Software (GCS)
Revisión de tareas
Reunion de planificacion
Reunión de diseño
Selección del entorno
Actualización Hoja Valor Ganado
Realizar diseño de alto nivel (DAN)
Realizar diseño de bajo nivel (DBN)
Revisión diseño alto nivel
Revisión diseño de bajo nivel
Creación de la base de la aplicación
Implementación de Creación de Usuarios
Implementación de Modificación de Usuarios
Implementación de Eliminación de Usuarios
implementación de Roles de Usuario
Implementación de Inicio de Sesión
Implementación de Creación de Torneos
Implementación de Emparejamiento de Jugadores
Implementación de Actualización de Resultados
Implementación del Proceso de Inscripción
Implementación de Selección de Jugadores
Implementación de Creación y Configuración de Partidos
Implementación del Registro de Resultados
Implementación de Consulta de Partidos
Inspección de código
Realización de pruebas Unitarias
Realización de pruebas del Sistema
Reunión revisión de código y pruebas
DEMO
Reunión Postmortem
Análisis Postmortem

7. Productos a obtener

Documento de Especificación
Diseño de Bajo Nivel
Diseño de Alto Nivel
Diagrama Gantt
Productos de codigo
Carpeta admin:

1. AccountManagementScreen (archivo: account_managementscreen.dart)
2. AdminScreen (archivo: admin_screen.dart)

Carpeta login:

3. LoginScreen (archivo: login_screen.dart)

Carpeta register:

4. EmailVerificationScreen (archivo: email_verification_screen.dart)
5. RegisterScreen (archivo: register_screen.dart)

Carpeta tournaments:

6. ConfigurarPartidosScreen (archivo: configurarPartidosScreen.dart)
7. CreateTournamentScreen (archivo: create_tournament_screen.dart)
8. EditarSetScreen (archivo: editarSetScreen.dart)
9. PairingsScreen (archivo: pairings_screen.dart)
10. PartidoScreen (archivo: partidoScreen.dart)
11. PlayerSelectionScreen (archivo: player_selection_screen.dart)
12. ResultadosPartidosScreen (archivo: resultados_partidos_screen.dart)
13. SelectTournamentsScreen (archivo: select_tournaments_screen.dart)
14. TournamentManagementScreen (archivo: TournamentManagementScreen.dart)
15. UserTournamentScreen (archivo: user_tournament_screen.dart)

Carpeta users:

16. ProfileScreen (archivo: profile_screen.dart)
17. UserScreen (archivo: user_screen.dart)

Raíz del proyecto:

18. App (archivo: app.dart)
19. FirebaseOptions (archivo: firebase_options.dart)
20. Main (archivo: main.dart)