Programación Didáctica del módulo "Proyecto Intermodular"

CFGS Desarrollo de Aplicaciones Multiplataforma (DAM) - Comunitat Valenciana

1. Contextualización

Esta programación está diseñada para ser implementada en un grupo heterogéneo de estudiantes, de segundo curso de DAM, que trabajarán en equipos de **4 o 5 miembros**, fomentando la colaboración, la comunicación y la responsabilidad compartida.

En este marco, el rol del profesorado evoluciona desde el de un instructor tradicional al de un **"facilitador"** La función principal del docente será guiar a los equipos, proveer de las herramientas metodológicas y técnicas necesarias, practicar la escucha activa y actuar como mentor para ayudar a los alumnos a superar los desafíos y alcanzar sus objetivos de forma autónoma.

Alineación con los Objetivos de Desarrollo Sostenible (ODS)

Un pilar fundamental del nuevo enfoque pedagógico, y un requisito explícito de esta programación, es el anclaje de todos los proyectos en un marco ético de impacto global: los **Objetivos de Desarrollo Sostenible (ODS)** de las Naciones Unidas Aunque en esta programación no se define un proyecto concreto, se establece como condición indispensable que cada equipo, en su fase de ideación, identifique un desafío social, medioambiental o económico real y alinee su propuesta de solución con uno o varios de los ODS.

2. Objetivos Generales del Proyecto

Al inicio del curso, cada equipo deberá definir los objetivos específicos de su proyecto utilizando la metodología S.M.A.R.T. Como guía, los objetivos generales de este módulo son:

- **S - Específico (Specific):** Desarrollar una solución software multiplataforma completa que incluya una aplicación de escritorio (WPF con C# y arquitectura MVVM), una aplicación móvil (Android con Kotlin, Jetpack Compose y arquitectura MVI) y un servicio backend (API REST con Java, Spring Boot y Spring Security). El sistema deberá incluir comunicación asíncrona (RabbitMQ) y estar gestionado mediante un flujo de trabajo profesional (Git, GitHub Projects, Markdown).
- **M - Medible (Measurable):** Implementar el 100% de los casos de uso mínimos definidos en la fase de diseño. Todos los artefactos del proyecto (código, documentación, diario de trabajo) deben estar actualizados semanalmente en el repositorio de GitHub.
- **A - Alcanzable (Achievable):** El alcance tecnológico es ambicioso pero factible en la temporalización asignada de **100 horas lectivas** (3h/semana durante 32-33 semanas) y aproximadamente **64 horas de trabajo autónomo** (2h/semana), utilizando las tecnologías del currículo de 2º de DAM.
- **R - Relevante (Relevant):** El proyecto exige la aplicación e integración práctica de las competencias clave de todos los módulos técnicos y transversales de 2º de DAM, simulando un entorno de desarrollo profesional real.
- **T - Delimitado en el Tiempo (Time-bound):** El proyecto se desarrollará íntegramente a lo largo de **16 quincenas (32 semanas)**, con una entrega final y presentación en la última quincena.

3. Metodología, Organización y Secuenciación de Contenidos

- Metodología de Trabajo Ágil (SCRUM)
  El proyecto se **planteará por retos**, siguiendo el marco de trabajo ágil **SCRUM**, por ser el estándar de facto en la industria del desarrollo de software y por su adecuación a los principios pedagógicos de flexibilidad, entrega iterativa y mejora continua.
- **Sprints:** Cada quincena lectiva constituirá un **Sprint** de dos semanas de duración. El proyecto se completará en un total de **16 sprints**.
  nt.
- **Roles:** El profesor actuará como **Product Owner**, definiendo los requisitos generales y el marco de trabajo. Los equipos de alumnos constituirán el **Development Team**. Para fomentar el liderazgo, el rol de **Scrum Master** podrá ser rotativo entre los miembros del equipo en cada sprint.

4. Herramientas y Entorno de Trabajo Colaborativo

- **GitHub:** Será la plataforma central para el control de versiones del código fuente. Se exigirá un uso correcto de un flujo de trabajo de ramas (ej. GitFlow simplificado), la realización de **Pull Requests** para la integración de código y la revisión de código entre compañeros como parte del proceso de calidad.
- **GitHub Projects:** Se utilizará para la gestión del **Product Backlog** y los **Sprint Backlogs**. Cada equipo mantendrá un tablero Kanban para visualizar el estado de las tareas (To Do, In Progress, Done), proporcionando transparencia sobre el progreso del proyecto.
- **Documentación en Markdown:** Toda la documentación del proyecto (definición, análisis, diseño, manuales, memoria) se generará en formato **Markdown** y se alojará en el propio repositorio de GitHub. Se recomienda el uso de extensiones como **Markdown Preview Enhanced** para la inclusión de diagramas (PlantUML), fórmulas y otros elementos enriquecidos.
- **Diario de Trabajo Individual:** Cada equipo deberá mantener un fichero a modo de diario de trabajo denominado `DIARIO.md` en la raíz del repositorio. En este diario, se registrará semanalmente las tareas realizadas, el tiempo dedicado, los problemas encontrados, las soluciones aplicadas y una reflexión sobre su aprendizaje. Rellenando una tabla en Markdown donde cada fila represente una **memoria semanal** de trabajo por alumno con las siguientes columnas:
  - **Alumno:** Nombre del alumno.
  - **Tareas Realizadas:** Descripción breve de las tareas completadas.
  - **Problemas Encontrados:** Descripción de los problemas o bloqueos.
  - **Soluciones Aplicadas:** Cómo se resolvieron los problemas.
  - **Reflexión:** Reflexión personal sobre el aprendizaje y la mejora continua.
    Ejemplo:

Semana	Alumno	Tareas Realizadas	Problemas Encontrados	Soluciones Aplicadas	Reflexión
1	Juan Pérez	Definición del proyecto,	Dificultad con GitHub Projects	Asistencia a un taller de GitHub	Aprendí a gestionar mejor el flujo

Semana	Alumno	Tareas Realizadas	Problemas Encontrados	Soluciones Aplicadas	Reflexión
		creación de repos			de trabajo en equipo.
1	María García	Investigación sobre ODS y definición de caso de uso	Falta de claridad en los ODS	Consulta a documentación oficial y foros	Mejoré mi comprensión de los ODS y su aplicación práctica.
2	María García	Diseño del modelo de dominio en Java	Confusión con PlantUML	Consulta a documentación y foros	Mejoré mi comprensión de UML y su aplicación práctica.
2	Juan Pérez	Implementación de clases Java y diagrama de clases	Problemas con la transpilación a C#	Asistencia a un taller de IA generativa	Aprendí a utilizar herramientas de IA para mejorar el código.

Este documento es un instrumento fundamental para la evaluación individual y **solo se tendrá en cuenta si se ha realizado un commit por semana** y se ha actualizado correctamente.

5. Secuenciación de Actividades y Contenidos Tecnológicos

La secuencia sigue un **flujo lógico de retos** que abarca desde la formación inicial en herramientas y metodologías, pasando por el diseño e implementación de las distintas capas del sistema, hasta la integración final y la entrega del proyecto.

- **Actividad 1: Presentación:** Presentación del módulo y presentación del grupo utilizando dinámicas de grupo para fomentar la cohesión y el trabajo en equipo.
- **Actividad 2: Formación de equipos** Formación de equipos con técnica **H.A.D.A.** (Hacedor, Analista, Divergente, Armonizador).
- **Actividad 3: Taller de Git y GitHub** Instalación VSCode, Creación de la organización y primer repositorio en GitHub. Definición de flujos de trabajo con ramas y Pull Requests.
- **Actividad 4: Definición del proyecto y ODS**. Cada equipo selecciona un desafío social y define su proyecto alineado con uno o varios ODS a partir de una plantilla usando una LLM (Gemini, ChatGPT, Perplexity, DeepSeek. etc.).
- **Actividad 5: Gestión Ágil con SCRUM** Repaso ceremonias SCRUM y configuración de GitHub Projects para la gestión del Product Backlog y los Sprint Backlogs. Definen un Backlog inicial y planifican el primer sprint.
- **Actividad 6: Diseño del Modelo de Dominio (Sprint 1)** Sprint Planning 1 y uso de IA generativa para ayudar en el diseño del modelo de dominio. Diseño del modelo de dominio en **C#** y creación del diagrama de clases con **PlantUML**. Documentación en `DISEÑO.md`.
- **Actividad 7: Implementación del Modelo de Dominio (Sprint 2)** Sprint Planning 2 y transpilación de las clases Java a **Java** y **Kotlin** usando IA generativa. Creación de Mocks de datos e imágenes para pruebas con IA.
- **Actividad 8: Prototipado UI/UX (Sprint 3)** Sprint Planning 3 y prototipado de UI/UX para Android (Material 3) y Escritorio (XAML) con **Figma/Penpot** y herramientas de IA como Google Stitch AI o Lovable.
- **Actividad 9: Frontend Escritorio (Sprint 4)** Sprint Planning 4. Implementación de la UI de la aplicación de admin con **WPF**.
- **Actividad 10: Frontend Escritorio (Sprint 5)** Sprint Planning 5. Definición arquitectura **MVVM** vinculando VM a los datos mock.
- **Actividad 11: Frontend Móvil (Sprint 6)** Sprint Planning 6. Implementación de la UI de la aplicación de cliente con **Android (Jetpack Compose)**.
- **Actividad 12: Frontend Móvil (Sprint 7)** Sprint Planning 7. Definición arquitectura **MVI** vinculando estados a los datos mock.
- **Actividad 13: Backend - API REST (Sprint 8)** Sprint Planning 8. Implementación de API REST con **Spring Boot** y Java. Definición de entidades y sus relaciones con a partir del modelo y generación base de datos H2 en fichero siguiendo el esquem Entity-First.
- **Actividad 14: Backend - API REST (Sprint 9)** Sprint Planning 9. Implementación de la lógica de negocio en los servicios de la API.
- **Actividad 15: Backend - Seguridad (Sprint 10)** Sprint Planning 10. Implementación de autenticación y autorización con **Spring Security** y **JWT**. Gestión de usuarios y claves hasheadas en BD.

- **Actividad 16: Backend - Seguridad (Sprint 11)** Sprint Planning 11. Protección de endpoints por roles (JWT).
- **Actividad 17: Integración Full-Stack (Sprint 12)** Sprint Planning 12. Sustitución de datos mock por el consumo real de la API REST en las aplicaciones de Escritorio (C#) y Android (Kotlin).
- **Actividad 18: Avanzado y Entrega Final (Sprint 13)** Sprint Planning 13. Elaboración de la memoria final, vídeo y preparación de la presentación.

6. Planificación y Temporalización Detallada (16 Quincenas)

Sprint	Semanas	Fechas	Objetivo Principal del Sprint	Entregables Clave
	1-2	08 sep 19 sep	Fundación y Metodología	Creación de equipos con técnicas H.A.D.A.
	3-4	22 sep 03 oct	Git y GitHub y Organización Definición del Proyecto y ODS	Documento de casos de uso y objetivos S.M.A.R.T. (MD).
	5-6	06 oct 17 oct	GitHub Project y Scrum	Documentación SCRUM en MD utilizando GitHub Projects.
1	7-8	20 oct 31 oct	IA y Modelo de Dominio (C#)	Diagrama de clases (PlantUML). Clases del modelo en C#.
2	9-10	03 nov 14 nov	Transpilación y Mocks con IA	Refinamientos, Clases en Kotlin y Java, ficheros de mocks de datos.
3	11-12 (**1º Ev**)	17 nov 05 dic	Prototipado UI/UX	Prototipos de UI/UX en Figma/Penpot.
4	13-14	08 dic 19 dic	Frontend Escritorio (WPF+MVVM)	Aplicación WPF con vistas y ViewModels, usando datos mock.
5	15-16	07 ene 16 ene	Frontend Escritorio (Funcional)	Lógica de negocio y navegación implementada en la app de escritorio.
6	17-18	19 ene 30 ene	Frontend Móvil (Android+MVI)	App Android con pantallas y estados (Compose), usando datos mock.
7	19-20	02 feb 13 feb	Frontend Móvil (Funcional)	Lógica de negocio y navegación implementada en la app móvil.
8	(**2º Ev**) 21-22	23 feb 06 mar	Backend: API REST (CRUD)	Proyecto Spring Boot con endpoints CRUD funcionales y BD H2.
9	23-24	09 mar 20 mar	Backend: API REST (Lógica)	Lógica de negocio implementada en los servicios de la API.
FE	**FE**	**FE**	**FE**	**FE**
10	25-26	23 mar 24 abr	Backend: Seguridad (JWT)	API con endpoints de login y registro.

Sprint	Semanas	Fechas	Objetivo Principal del Sprint	Entregables Clave
11	27-28	27 abr 08 may	Backend: Autorización	API con endpoints protegidos por roles (JWT).
12	29-30	11 may 22 may	Integración Full-Stack	Apps WPF y Android conectadas a la API REST segura.
13	31-32	25 may 05 jun	Avanzado y Entrega Final	(Opcional) RabbitMQ/CI/CD. Memoria final, vídeo y presentación.

7. Sistema de Evaluación Integrada

La evaluación del módulo será **continua, formativa e integradora**, combinando la valoración del producto final con el seguimiento del proceso y las contribuciones individuales.

Se utilizarán rúbricas detalladas con una escala de **1 (Insuficiente) a 5 (Excelente)** para los entregables clave.

Ejemplo de Rúbrica: Entrega Sprint 10 - Seguridad del Backend

Criterio de Evaluación	1 (Insuficiente)	3 (Suficiente)	5 (Excelente)	Tipo
Funcionalidad de la Seguridad	La seguridad no funciona, es inexistente o se puede eludir fácilmente.	La autenticación básica con JWT funciona, pero la autorización por roles es incompleta o presenta fallos.	Sistema de seguridad robusto con autenticación JWT y autorización por roles granular y correctamente implementada.	Grupal
Calidad del Código Backend	Código desorganizado, con malas prácticas, difícil de leer y mantener.	El código es funcional pero podría ser más eficiente y seguir mejor los patrones de diseño (capa de servicio, DTOs).	Código limpio, bien estructurado (capas bien definidas), eficiente, documentado y que sigue las mejores prácticas.	Grupal
Aportaciones al Repositorio	Commits inexistentes, muy esporádicos o con mensajes inútiles.	Realiza commits funcionales pero con mensajes poco descriptivos. Participa	Commits regulares, atómicos y con mensajes claros. Participa activamente en la revisión de	Individual

Criterio de Evaluación	1 (Insuficiente)	3 (Suficiente)	5 (Excelente)	Tipo
	No participa en PRs.	mínimamente en las revisiones.	código de sus compañeros.	
Registro en Diario de Trabajo	El diario está vacío, incompleto o las entradas son superficiales.	El diario se actualiza, pero con poca regularidad. Las descripciones son breves y poco reflexivas.	El diario se actualiza semanalmente con detalle, explicando tareas, problemas, soluciones y reflexiones profundas.	Individual

Escala de Valoración Final del Proyecto

La calificación final del módulo se calculará ponderando diferentes bloques para equilibrar la valoración grupal e individual.

Bloque de Evaluación	Ponderación	Criterios de Evaluación (Individuales y Grupales)	RA del Módulo Relacionados
Calidad del Producto y Proceso (Grupal)	**60%**	**(40%) Calidad Técnica:** Cumplimiento de requisitos, robustez, correcta implementación de tecnologías, calidad del código. **(20%) Gestión y Documentación:** Correcta aplicación de SCRUM, calidad de la documentación, memoria final.	RA2, RA3
Desempeño y Contribución Individual (Individual)	**40%**	**(15%) Contribución Técnica:** Calidad y cantidad de las aportaciones al repositorio (commits, PRs). **(15%) Diario de Trabajo:** Regularidad, detalle y profundidad de las reflexiones. **(10%) Participación y Defensa:** Colaboración activa, participación en ceremonias y calidad de la defensa individual.	RA1, RA4