

# DOCUMENTO DE PLANIFICACIÓN

## Proyecto CMS - Escuela de Baile Urbano

### 1. ENTORNO DE TRABAJO

#### 1.1 Herramientas Colaborativas

##### Discord

Uso: Videollamadas, reuniones, compartir pantalla y **Comunicación diaria/Chat General**.

Justificación:

- Llamadas sin límite de tiempo.
- Compartir pantalla.
- Canales organizados por temas.
- Todos los miembros ya lo usan habitualmente.

Casos de uso específicos:

- Reuniones de planning y retrospectivas.
- Sesiones de trabajo conjunto.
- Demostración del WordPress al equipo.
- Resolución de problemas técnicos en tiempo real.
- **Chat general para comunicación diaria y dudas (Sustituye a Teams).**

##### GitHub

Control de versiones, repositorio oficial y **almacenamiento centralizado de todos los archivos**.

Justificación:

- Control de versiones profesional.
- Almacenamiento ordenado de documentos y **todos los entregables finales**.
- Organización clara por entregas.

- Historial completo de cambios (*commits*).
- **Vinculación directa a archivos locales para facilitar *commit* de documentación.**

Estructura del repositorio:

cms-escuela-baile/

- README.md (descripción del proyecto)
- Sprint-1/
  - 01-Requisitos.pdf
  - 02-Analisis.pdf
  - 03-Diseño.pdf
  - 04-Implementacion.pdf
  - 05-Planificacion.pdf
- docs/
  - guia-entrevista.md
  - backlog.md
  - manual-instalacion.md
- assets/
  - mockups/
  - diagramas/
  - capturas-wordpress/

Política de subida:

- Se sube la versión FINAL de cada documento.
- **Los borradores y *screenshots* de progreso se almacenan también en GitHub (en carpetas de trabajo).**

## 1.2 Flujo de Trabajo Centralizado

### WordPress en Servidor Compartido con Acceso Remoto

Todo el desarrollo de WordPress se realizará en un servidor web único compartido, accesible por URL.

Razones:

1. Permite a **todos los miembros del equipo trabajar remotamente** en el CMS al mismo tiempo.
2. Configuración única y consistente, sin necesidad de entornos locales.
3. Facilita la demostración del trabajo al profesor a través de un enlace web.
4. No hay conflictos de versiones de *plugins* o base de datos.

Proceso:

1. Instalación inicial: Un miembro (Juan) instala WordPress en el **Servidor Compartido**.
2. Sesiones de trabajo:
  - Reuniones por Discord compartiendo pantalla para visualización.
  - Cada miembro accede al CMS desde su equipo.
3. Backups regulares:
  - Export de base de datos (.sql) → Subir a **GitHub** (carpeta de backups).
  - *Screenshots* de progreso → Subir a **GitHub/assets/capturas-wordpress/**.

## Traspaso de Información entre Miembros

Ejemplo práctico:

1. Adri crea *mockups* en Balsamiq → Subir .png.
2. Juan los descarga → Implementa las páginas en WordPress (Servidor) → Hace *screenshots*.
3. David documenta en Word → Sube borrador → Revisa equipo.
4. Versión final → Se sube a GitHub como PDF .

## 1.3 Comunicación según Tipo de Tarea

Tipo de comunicación	Herramienta	Ejemplo
Reuniones planificación	Discord (videollamada)	Sprint Planning
Trabajo conjunto	Discord (compartir pantalla)	Configurar <i>plugins</i> juntos
Chat diario	Discord (chat)	"He terminado los <i>mockups</i> "
Almacenar borradores	GitHub (Archivos/Issues)	Versiones de trabajo de requisitos
Seguimiento de tareas	GitHub (Issues) / Taiga	Ver qué está pendiente
Versión final oficial	GitHub	PDF final para el profesor
Historial de cambios	GitHub ( <i>commits</i> )	"Actualizado diagrama de clases"

## 1.4 Herramientas de Desarrollo

### MAMP

Uso: Servidor local para pruebas (**solo para componentes**). El desarrollo principal es en el servidor compartido.

Componentes (Base de la arquitectura del servidor compartido):

- Apache (servidor web)
- MySQL (base de datos)
- PHP (lenguaje de WordPress)

### Balsamiq/Figma

Uso: Creación de *mockups*.

Proceso:

1. Diseñar *mockups*.
2. Exportar como PNG.
3. Subir a **GitHub**.

### Lucichart

Uso: Diagramas de clases y arquitectura.

Proceso:

1. Crear diagrama *online*.
2. Exportar como PNG y XML.
3. Subir a **GitHub**.

## 2. METODOLOGÍA SCRUM

Software utilizado: Taiga

Justificación: Software especializado en la utilización de SCRUM, *backlogs*, *burndown chart*. Gratuito y permite exportar el contenido.

### 2.1 Adaptación al Proyecto

- Scrum Master: Facilita reuniones.
- Product Owner: Prioriza *backlog*.
- Developers: Todos los miembros participan en desarrollo y documentación.

## Sprint Planning

- Cuándo: Inicio de cada *sprint*.
- Dónde: Discord (videollamada).
- Objetivo: Seleccionar tareas del *Product Backlog* para el *Sprint Backlog*.

## Daily Standup (adaptado)

- Dónde: **Discord (Chat General)/Whatsapp.**
- Formato: Cada uno escribe:
  - ¿Qué hice hoy?
  - ¿Qué haré mañana?
  - ¿Tengo impedimentos?

## Sprint Review

- Cuándo: Día de entrega o día de antes.
- Dónde: Laboratorio (presencial) o Discord.
- Objetivo: Demostrar trabajo al profesor (cada Lunes en horario lectivo).

## Sprint Retrospective

- Cuándo: Después de cada *sprint*.
- Dónde: Discord (videollamada).
- Objetivo: ¿Qué funcionó? ¿Qué mejorar?

# 3. PRODUCT BACKLOG

Lista completa de tareas del proyecto, ordenadas por prioridad.

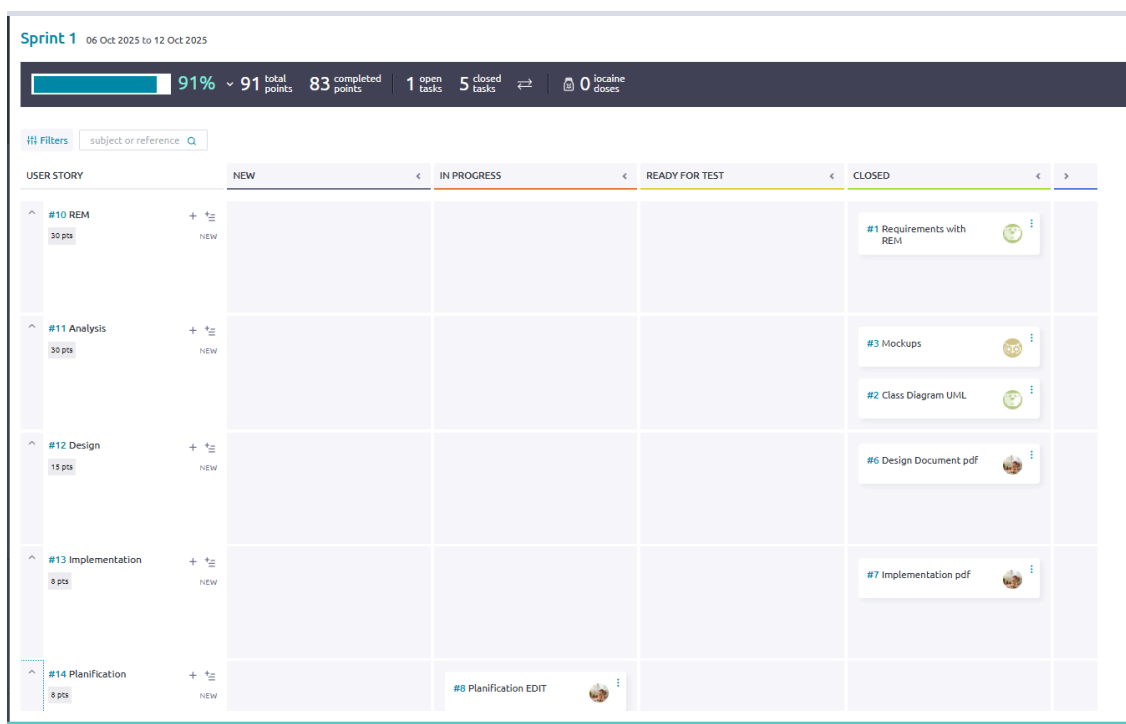
### Leyenda de Estados

- New - No iniciada
- In progress - Alguien trabajando
- Ready for test - revisión
- Closed - Terminada y revisada
- Needs info - poca información para realizar la tarea



## 4. SPRINT 1 BACKLOG

Tareas seleccionadas para completar en el primer *sprint*. Objetivo del Sprint 1: Documentar requisitos, análisis, diseño, instalar WordPress e informar sobre planificación.



# 5. BURNDOWN CHART - SPRINT 1

