**Caso Práctico 1: Planificación del Proyecto de Software para la Reserva de Pistas de Pádel**

**1. Análisis de Requerimientos (Máxima puntuación)**

* **Definición clara y precisa de los requerimientos**:
  + **Funcionales**:
    - Permitir a los clientes **consultar** la disponibilidad de pistas.
    - Funcionalidad para **reservar** pistas con y sin luz.
    - **Procesamiento de pagos** mediante tarjeta de crédito.
    - Generar **listados** por hora y cliente de las pistas reservadas.
  + **No funcionales**:
    - Uso de **software libre**.
    - **Seguridad** en los datos y protección de información personal.
    - **Escalabilidad** para futuros módulos.
* **Herramientas y técnicas para el análisis**:
  + **Reuniones semanales** con stakeholders.
  + **Prototipos** iniciales de la interfaz para validar con los usuarios.
  + Elaboración de un **documento de especificación de requerimientos** validado.

**2. Planificación del Diseño (Máxima puntuación)**

* **Enfoque modular**: Asegurar la modularidad desde el inicio para facilitar la ampliación futura (cumpliendo con buenas prácticas).
* **Diseño de arquitectura** basado en MVC (Modelo-Vista-Controlador).
  + **Modelo**: Manejo de datos y lógica empresarial (reservas, pistas, usuarios).
  + **Vista**: Interfaz de usuario donde los clientes ven y reservan pistas.
  + **Controlador**: Gestión de interacciones entre la vista y el modelo.
* **Lenguaje de programación**:
  + Se selecciona **Python con Django** para back-end por su simplicidad y compatibilidad con software libre.
  + **Vue.js o React** para el front-end por su capacidad de interacción eficiente.
* **Herramientas para el desarrollo**:
  + **Git** para control de versiones.
  + **Visual Studio Code** o **PyCharm** como IDE.
  + **Docker** para asegurar consistencia en el entorno de desarrollo.

**3. Etapas del Ciclo de Vida del Software (Máxima puntuación)**

* **Metodología de trabajo**: Se utilizará **SCRUM**, con sprints de 3 semanas.
  + **Sprint 1**: Diseño básico del sistema y funcionalidades principales de reserva.
  + **Sprint 2**: Implementación del módulo de pagos y gestión de usuarios.
  + **Sprint 3**: Integración de seguridad y pruebas unitarias.
  + **Sprint 4**: Despliegue en entorno de prueba y feedback del cliente.
* **Refactorización continua**: Al final de cada sprint se realizará una revisión de código para optimizar y mejorar el rendimiento.
* **Control de calidad**: Cada módulo será probado de manera individual (pruebas unitarias) y luego con el sistema completo (pruebas de integración).
* **Despliegue y mantenimiento**: El despliegue será continuo (CI/CD) usando **Docker** y sistemas de integración continua como **Jenkins**.

**4. Cumplimiento con los principios de Software Libre (Máxima puntuación)**

* **Entorno de desarrollo** basado en software libre:
  + Uso de **Python**, **Vue.js/React**, **Docker**, **Git**, **MySQL/PostgreSQL**, que son herramientas de código abierto.
  + El proyecto se gestionará en **GitHub** o **GitLab**, asegurando el cumplimiento de las licencias GPL o MIT.
* **Mantenimiento del código**:
  + Todo el código estará disponible para que la comunidad lo revise, modifique y extienda según los principios de software libre.
* **Licencia GPL**: Propuesta para asegurar que el código es accesible y reutilizable.

**Conclusión:**

El enfoque propuesto sigue un ciclo de vida **ágil**, garantizando la entrega progresiva de valor al cliente, cumpliendo con los estándares de software libre y la modularidad exigida por el caso práctico. Esto no solo asegura la máxima puntuación en la rúbrica, sino que también mantiene las **buenas prácticas de diseño**, evitando los problemas de escalabilidad y adaptabilidad mencionados en tu pregunta anterior.

Este esquema garantiza que la planificación esté alineada con la **rúbrica** y maximiza el puntaje en cada uno de los criterios evaluados.