Okay, the user needs a complete Scrum plan for a hospital web project with patient, user, and appointment management, including scheduling as per patient requests. Let me start by breaking down the main components.

First, I should outline the project vision. The primary goal is a web app for managing patients, users, and medical appointments with flexible scheduling. The key stakeholders are hospital admins, medical staff, and patients.

Next, the product backlog. The user mentioned three main modules: patient, user, and appointment management. Each of these will have their own user stories. For patients, CRUD operations are essential. Users need role-based access, so different permission levels for admins, doctors, and receptionists. Appointments should allow patients to choose time slots based on doctor availability. Also, notifications via email or SMS are important for reminders.

I need to consider non-functional requirements too. Security is crucial because it's healthcare data, so HIPAA or similar compliance. Performance-wise, handling high traffic and quick search responses. Usability for different user roles, maybe a responsive design.

Now, structuring the sprints. A typical Scrum approach might use 2-week sprints. Let's plan for 4 sprints. Sprint 1 could focus on setting up the environment, authentication, and basic patient management. Sprint 2 adds user management with roles and permissions. Sprint 3 tackles appointment scheduling, including doctor availability and conflict checks. The final sprint would handle notifications, reports, and final touches like UI polish and testing.

The team roles are standard: Product Owner, Scrum Master, Developers, QA, UX/UI Designer. Tools like Jira or Trello for tracking, GitHub for code, Slack for communication, and Zoom for meetings.

Including a Definition of Done is important—each task needs testing, code review, documentation, and deployment to staging. Risks like time estimation errors or changing requirements should be noted, along with mitigation strategies.

Wait, did I miss anything? Let me check. The user wanted scheduling based on patient requests, so ensuring that the appointment system allows patients to select preferred times, but also checks doctor availability. Maybe a calendar UI for patients to pick slots. Also, notifications for both patients and doctors. Security aspects like data encryption and access controls. Oh, and reporting for appointments, maybe daily or weekly summaries.

I think that covers the main points. Each sprint builds on the previous, with testing integrated throughout. Including a risk assessment shows foresight. The timeline of 8 weeks seems reasonable for a basic but functional system. Post-launch support is also a good addition for any unforeseen issues.

Aquí tienes una planificación Scrum completa para el proyecto web del hospital, estructurada en diferentes componentes:

### 1. \*\*Visión del Proyecto\*\*

\*\*Objetivo\*\*: Desarrollar una aplicación web para gestionar pacientes, usuarios y citas médicas en un hospital, permitiendo a los pacientes solicitar horarios específicos y a los administradores gestionar la disponibilidad de médicos.

\*\*Stakeholders principales\*\*:

- Administradores del hospital

- Personal médico

- Pacientes

- Equipo de desarrollo

---

### 2. \*\*Product Backlog (Lista de Requerimientos Prioritizados)\*\*

#### \*\*Módulo 1: Mantenedor de Pacientes\*\*

- \*\*User Stories\*\*:

1. Como administrador, quiero crear, editar y eliminar pacientes (CRUD).

2. Como paciente, quiero registrarme con mi información personal (nombre, RUT, contacto, historial médico).

3. Como médico, quiero buscar pacientes por nombre o RUT.

#### \*\*Módulo 2: Mantenedor de Usuarios\*\*

- \*\*User Stories\*\*:

4. Como administrador, quiero crear usuarios con roles (médico, recepcionista, administrador).

5. Como usuario, quiero iniciar sesión con autenticación segura (JWT/OAuth).

6. Como administrador, quiero asignar permisos por rol (ej: médicos solo ven sus citas).

#### \*\*Módulo 3: Mantenedor de Citas Médicas\*\*

- \*\*User Stories\*\*:

7. Como paciente, quiero solicitar una cita eligiendo especialidad, médico y horario disponible.

8. Como recepcionista, quiero confirmar/cancelar citas y enviar notificaciones por correo/SMS.

9. Como médico, quiero ver mi agenda diaria con detalles de pacientes.

#### \*\*Requerimientos No Funcionales\*\*:

- Seguridad: Cumplir con HIPAA (o normativa local) para datos médicos.

- Rendimiento: Soporte para 1000 usuarios concurrentes.

- Usabilidad: Interfaz responsive para móvil y desktop.

---

### 3. \*\*Sprint Planning (Ejemplo para 4 Sprints de 2 semanas)\*\*

#### \*\*Sprint 1: Configuración y Módulo de Pacientes\*\*

- \*\*Objetivo\*\*: Entorno técnico listo + CRUD de pacientes.

- \*\*Tareas\*\*:

- Configurar repositorio (Git), base de datos (PostgreSQL), y API REST.

- Desarrollo de formulario de registro de pacientes (frontend y backend).

- Implementar búsqueda de pacientes por RUT.

#### \*\*Sprint 2: Módulo de Usuarios y Autenticación\*\*

- \*\*Objetivo\*\*: Sistema de roles y permisos funcional.

- \*\*Tareas\*\*:

- Desarrollo de login con JWT.

- Crear interfaz para asignar roles (admin).

- Integrar permisos (ej: recepcionista no puede eliminar pacientes).

#### \*\*Sprint 3: Sistema de Citas Médicas\*\*

- \*\*Objetivo\*\*: Agendamiento flexible de citas.

- \*\*Tareas\*\*:

- Calendario interactivo para selección de horarios.

- Validación de disponibilidad de médicos.

- Notificaciones automáticas (ej: correo de confirmación).

#### \*\*Sprint 4: Mejoras y Pruebas Finales\*\*

- \*\*Objetivo\*\*: Pulir funcionalidades y asegurar calidad.

- \*\*Tareas\*\*:

- Optimizar rendimiento de búsquedas.

- Pruebas de estrés y seguridad.

- Documentación técnica y manual de usuario.

---

### 4. \*\*Equipo Scrum\*\*

- \*\*Product Owner\*\*: Representante del hospital (prioriza el backlog).

- \*\*Scrum Master\*\*: Facilitador del equipo técnico.

- \*\*Equipo de Desarrollo\*\*:

- 2 Backend Developers (Node.js/Python)

- 2 Frontend Developers (React/Angular)

- 1 QA Engineer

- 1 UX/UI Designer

---

### 5. \*\*Herramientas\*\*

- \*\*Gestión de Proyectos\*\*: Jira, Trello o ClickUp.

- \*\*Desarrollo\*\*: GitHub/GitLab, Postman (API testing).

- \*\*Comunicación\*\*: Slack, Zoom para dailies.

- \*\*Base de Datos\*\*: PostgreSQL con cifrado de datos sensibles.

---

### 6. \*\*Entregables por Sprint\*\*

- \*\*Al final de cada Sprint\*\*:

- Demostración funcional al cliente.

- Retrospectiva para mejorar procesos.

- Actualización del backlog según feedback.

---

### 7. \*\*Definición de "Done" (DoD)\*\*

Cada tarea se considera terminada cuando:

- El código pasa pruebas unitarias y de integración.

- Está revisado por otro desarrollador (code review).

- La documentación está actualizada.

- Se despliega en entorno de staging.

---

### 8. \*\*Riesgos y Mitigación\*\*

- \*\*Riesgo 1\*\*: Cambios en requerimientos legales (ej: normas de privacidad).

- \*Mitigación\*: Reuniones semanales con stakeholders para validar.

- \*\*Riesgo 2\*\*: Conflictos de horarios por sobrecupos médicos.

- \*Mitigación\*: Implementar lógica de validación en tiempo real.

---

### 9. \*\*Cronograma Estimado\*\*

- \*\*Duración Total\*\*: 8 semanas (4 sprints).

- \*\*Lanzamiento\*\*: MVP listo en 2 meses, con soporte post-lanzamiento para ajustes.

---

### 10. \*\*Ejemplo de Tablero Scrum (Sprint 1)\*\*

| \*\*Tarea\*\* | \*\*Responsable\*\* | \*\*Estado\*\* |

|-----------|------------------|------------|

| Configurar API REST | Backend Dev | ✔️ Done |

| Diseñar formulario de pacientes | UX Designer | ⏳ In Progress |

| Pruebas de registro | QA Engineer | 🕒 To Do |