Plantilla para Generar un Proyecto en Ingeniería de Software

Título del Proyecto:

Aplicación web para préstamos de implementos deportivos(WAPED)

Descripción:

En la sociedad actual, muchas instituciones enfrentan carencias tanto en términos de recursos como de instalaciones deportivas. Esto ha llevado a la falta de desarrollo en el ámbito deportivo dentro de estas instituciones, convirtiéndose en un problema social. Con el objetivo de abordar esta situación, se propone la creación de una aplicación web para administrar el préstamo y control de equipos deportivos en la Institución Educativa Distrital Mayor de Barranquilla y del Caribe. Esta iniciativa está dirigida a instituciones cercanas con recursos limitados que requieran estos elementos para fomentar la actividad deportiva.

Objetivos del Proyecto:

1.      Realizar un estudio de la población que incluya personas con recursos y personas sin recursos que deseen utilizar los servicios de préstamo de implementos deportivos.

2.      Recolectar los requisitos funcionales y no funcionales del sistema de gestión de préstamos de equipamiento y espacios deportivos, para construir los diagramas UML (Lenguaje Unificado Modelado de datos), utilizando herramientas case.

3.      Implementar un diseño detallado del software de gestión de préstamos de equipamiento y espacios deportivos, incluyendo la arquitectura del sistema, la interfaz de usuario, las funciones principales y la integración de características específicas.

4.      Construir el software de gestión de préstamos de equipamiento y espacios deportivos, siguiendo el diseño previamente establecido, garantizando la funcionalidad completa de todas las características especificadas.

Requerimientos:

Requerimientos Funcionales:

* El sistema tiene que llevar a cabo operaciones CRUD y asignar funciones (permisos) a los usuarios según su rol.

Desripcion:

La aplicación debe permitir al administrador crear la información esencial del usuario, como nombre, apellido, curso, correo, contraseña y rol. Además, el administrador debe tener la capacidad de consultar, editar y eliminar usuarios en la base de datos. El sistema debe proporcionar un control detallado de los permisos para cada usuario, especialmente para los administradores.

* El sistema debe implementar operaciones CRUD para los elementos y espacios deportivos

Descripcion:

La aplicación debe posibilitar la creación de nuevos elementos deportivos, junto con la opción de crear o elegir una categoría y subcategoría para cada elemento. Se deben ingresar los datos respectivos, como número, estado, horas disponibles e imagen. Además, la aplicación debe permitir consultar, editar y eliminar los elementos deportivos almacenados en la base de datos.

* El sistema debe llevar a cabo, aprobar y cancelar préstamos de artículos deportivos.

Descripcion:

La plataforma debe posibilitar a los estudiantes solicitar un préstamo de artículos deportivos, eligiendo el artículo y la hora deseados. Además, se espera que la aplicación permita al administrador aprobar o rechazar la solicitud de préstamo.

* El sistema debe registrar las devoluciones de artículos deportivos realizadas por los usuarios.

Descripcion:

La aplicación debe brindar al administrador la capacidad de revisar el préstamo, verificar el estado del artículo deportivo, ingresar la información necesaria y guardar los datos correspondientes.

Requerimientos No Funcionales:

* Seguridad

Los datos personales y financieros de los usuarios son protegidos de forma segura para evitar el acceso no autorizado, el acceso a la aplicación será realizado por el mismo usuario. Cada estudiante y administrador debe tener un usuario y una contraseña de acceso a la aplicación que le permite iniciar sesión y hacer uso de ella.

* Mantenimiento

El software sera actualizado regularmente para garantizar la seguridad y la funcionalidad óptima.

* Rendimiento

La plataforma rápida y eficiente que garantiza una buena experiencia de usuario, maneja un alto volumen de usuarios sin afectar su rendimiento y esta disponible y accesible en todo momento, sin interrupciones no programadas.

Planificación del Proyecto:

Fases del Proyecto

[Fase 1: Planificación y Análisis]

Descripción de la fase: En esta fase, se realizará una planificación detallada del proyecto y se llevará a cabo un análisis exhaustivo de los requisitos del sistema.

Actividades:

* Definir el alcance del proyecto.
* Establecer los objetivos y entregables del proyecto.
* Identificar los interesados y establecer comunicación con ellos.
* Recolectar y analizar los requisitos funcionales y no funcionales del sistema.
* Realizar un estudio de factibilidad técnica y económica.
* Elaborar un plan de trabajo y asignar recursos.
* Crear un documento de especificación de requisitos del software.

[Fase 2: Diseño del Sistema]

Descripción de la fase: En esta fase, se diseñará la arquitectura del sistema y se elaborará el diseño detallado del software.

Actividades:

* Diseñar la interfaz de usuario (UI) y la experiencia de usuario (UX).
* Elaborar diagramas UML para modelar el sistema.
* Diseñar la base de datos y definir el modelo de datos.
* Detallar las funciones principales del sistema y su integración.
* Revisar y validar el diseño con los interesados.

[Fase 3: Implementación y Desarrollo]

Descripción de la fase: En esta fase, se llevará a cabo la codificación del software y la implementación de las funcionalidades especificadas.

Actividades:

* Configurar el entorno de desarrollo.
* Desarrollar los módulos y componentes del sistema.
* Realizar pruebas unitarias para verificar el funcionamiento de cada componente.
* Integrar los diferentes módulos y realizar pruebas de integración.
* Implementar medidas de seguridad y protección de datos.
* Documentar el código y preparar manuales de usuario y administrador.

[Fase 4: Pruebas y Validación]

Descripción de la fase: En esta fase, se realizarán pruebas exhaustivas para validar el funcionamiento del sistema y garantizar su calidad.

Actividades:

* Planificar y diseñar casos de prueba para cubrir todos los escenarios posibles.
* Ejecutar pruebas de unidad, integración, sistema y aceptación.
* Registrar y corregir los errores identificados durante las pruebas.
* Realizar pruebas de rendimiento y seguridad.
* Validar el cumplimiento de los requisitos del cliente.

[Fase 5: Implementación y Mantenimiento]

Descripción de la fase: En esta fase, se realizará la implementación final del sistema y se establecerá un plan de mantenimiento para futuras actualizaciones y mejoras.

Actividades:

* Desplegar el sistema en un entorno de producción.
* Capacitar a los usuarios finales y al personal de soporte.
* Monitorear el rendimiento y la estabilidad del sistema en producción.
* Recopilar comentarios y sugerencias de los usuarios para futuras mejoras.
* Programar y realizar actualizaciones periódicas del software.
* Brindar soporte técnico y resolver problemas reportados por los usuarios.

Recursos:

Personal:

* Desarrolladores
* Administrador de base de datos
* Tester/QA

Herramientas:

* Entorno de desarrollo integrado (IDE): Visual Studio Code, IntelliJ IDEA, etc.
* Frameworks y bibliotecas: React.js, Node.js, Express.js, MongoDB, etc.
* Herramientas de gestión de proyectos: Jira, Trello, Asana, etc.
* Sistema de control de versiones: GitHub, Bitbucket, etc.

Equipos:

* Computadoras portátiles/desktops para cada miembro del equipo con especificaciones adecuadas para el desarrollo de software.
* Conexión a internet de alta velocidad.
* Servidor de prueba para despliegue y pruebas del sistema.

Cronograma:

* Fase 1: Planificación y Análisis
* Duración: 4 semanas (1 mes)
* Fechas: [Fecha de inicio] - [Fecha de finalización]
* Fase 2: Diseño del Sistema
* Duración: 4 semanas (1 mes)
* Fechas: [Fecha de inicio] - [Fecha de finalización]
* Fase 3: Implementación y Desarrollo
* Duración: 1 semanas (0.25 meses)
* Fechas: [Fecha de inicio] - [Fecha de finalización]
* Fase 4: Pruebas y Validación
* Duración: 2 semanas (0.5 mes)
* Fechas: [Fecha de inicio] - [Fecha de finalización]
* Fase 5: Implementación y Mantenimiento
* Duración: Indefinido (mantenimiento continuo)
* Fechas: A partir de la finalización de la fase 4

[Fecha de inicio del proyecto: DD/MM/AAAA]

[Fecha de finalización del proyecto: DD/MM/AAAA]

Metodología:

El modelo en cascada es un enfoque secuencial y lineal para el desarrollo de software, donde las fases del proyecto se llevan a cabo de manera secuencial y cada fase debe completarse antes de pasar a la siguiente.

Aplicación en este Proyecto:

* Se llevará a cabo una planificación exhaustiva al inicio del proyecto para definir claramente los requisitos y objetivos del sistema.
* Cada fase del proyecto se completará antes de pasar a la siguiente, siguiendo una secuencia predefinida de actividades.
* Se asignará un período de tiempo específico para cada fase, con entregables claros y fechas de finalización establecidas.
* Se realizarán revisiones y controles de calidad en cada fase para garantizar que los entregables cumplan con los estándares y requisitos definidos.
* Se realizará una fase de pruebas exhaustiva al final del desarrollo para verificar el funcionamiento del sistema y corregir cualquier error identificado.
* Una vez que se complete el desarrollo y las pruebas, se procederá a la fase de implementación y mantenimiento del sistema en producción.

Roles y Responsabilidades:

**Diseñador de Experiencia de Usuario (User Experience Designer):** Este rol se encarga de diseñar la experiencia del usuario (UX) del software. Esto puede incluir diseñar la interfaz de usuario, realizar pruebas de usabilidad y asegurarse de que el software sea intuitivo y fácil de usar.

**Desarrollador (Developer):** Los desarrolladores son responsables de escribir y mantener el código del software. Dependiendo del tamaño del equipo, puede haber diferentes niveles de desarrolladores, como desarrolladores senior, desarrolladores intermedios y desarrolladores junior.

**Administrador de Base de Datos (Database Administrator):** Este rol es responsable de diseñar, implementar y mantener la base de datos del sistema. Esto puede incluir la creación y gestión de esquemas de bases de datos, el monitoreo del rendimiento de la base de datos y la realización de copias de seguridad y restauraciones de datos.

**Control de Calidad (Quality Assurance):** Este rol es responsable de probar el software para identificar errores y asegurarse de que cumpla con los estándares de calidad establecidos. El equipo de control de calidad puede incluir probadores manuales, probadores automatizados y analistas de control de calidad.

**Soporte Técnico (Technical Support):** Este rol es responsable de proporcionar soporte técnico a los usuarios del software. Esto puede incluir la resolución de problemas, la respuesta a preguntas técnicas y la gestión de incidencias y solicitudes de los usuarios.

Comunicación:

1. **Reuniones Regulares:**
   * **Reuniones Diarias de Pie (Daily Stand-ups):** Estas reuniones cortas (generalmente de no más de 15 minutos) se llevan a cabo diariamente para que cada miembro del equipo pueda informar sobre lo que ha hecho el día anterior, lo que planea hacer hoy y si hay algún impedimento que le impida avanzar en su trabajo.
   * **Reuniones de Planificación (Sprint Planning Meetings):** Estas reuniones, que suelen tener lugar al comienzo de cada sprint, permiten al equipo de desarrollo y al propietario del producto revisar y priorizar las tareas para el próximo sprint.
   * **Reuniones de Revisión (Sprint Review Meetings):** Al final de cada sprint, el equipo presenta los resultados de su trabajo al propietario del producto y otras partes interesadas y recibe retroalimentación.
   * **Reuniones de Retrospectiva (Sprint Retrospective Meetings):** Después de la revisión del sprint, el equipo se reúne para reflexionar sobre lo que funcionó bien durante el sprint y lo que podría mejorarse en el próximo sprint.
   * **Reuniones Ad-Hoc:** Además de las reuniones planificadas, puede ser necesario organizar reuniones adicionales según las necesidades del proyecto.
2. **Herramientas de Colaboración:**
   * **Plataformas de Gestión de Proyectos:** Herramienta GitHub.
   * **Comunicación en Línea:** Herramienta de mensajería instantánea Slack.
   * **Videoconferencias:** Plataforma Zoom.
   * **Documentación Compartida:** Herramienta Google Drive.
   * **Control de Versiones:** Herramientas como Git y GitHub.
3. **Gestión de Cambios y Comunicación con Partes Interesadas:**
   * **Reuniones de Revisión de Requisitos:** Antes de comenzar el desarrollo, es importante reunirse con las partes interesadas para comprender completamente los requisitos del proyecto.
   * **Demostraciones del Producto:** A lo largo del proyecto, es útil organizar demostraciones del producto para las partes interesadas para que puedan ver el progreso y proporcionar retroalimentación.
   * **Comunicación Regular:** Mantener a las partes interesadas informadas sobre el progreso del proyecto y cualquier cambio en los requisitos a través de correos electrónicos, boletines informativos u otros medios.
   * **Gestión de Cambios:** Si surgen cambios en los requisitos del proyecto, es importante gestionar estos cambios adecuadamente y comunicarlos a todas las partes interesadas.

Control de Calidad:

Plan de pruebas unitarios sobre los modelos que conforman la aplicación

Registros y consulta Tiempo ejecución Resultado

|  |  |  |
| --- | --- | --- |
| Registros y consulta | Tiempo ejecución | Resultado |
| Usuario | 2.8s | Exitoso |
| Curso | 1.90s | Exitoso |
| Elemento deportivo | 2.21s | Exitoso |
| Estado elemento | 1.97s | Exitoso |
| Subcategoría | 1.94s | Exitoso |
| Categoría | 1.87s | Exitoso |
| Préstamo | 2.02s | Exitoso |
| Devolución | 1.84s | Exitoso |

Gestión de Riesgos:

[Identificación de posibles riesgos del proyecto y planificación de estrategias para mitigarlos o enfrentarlos en caso de que ocurran.]

Presupuesto:

El presupuesto para el desarrollo de este proyecto es alrededor de 30.000.000 millones de pesos.

Aprobación del Proyecto:

[Firma y fecha de aprobación por parte del cliente o la dirección del proyecto.]