### **Título del Proyecto: Social Structure - Implementación de estructuras no lineales**

### **1. Resumen Ejecutivo**

El proyecto "Social Structure" es una red social en desarrollo en el contexto de un curso de estructuras de datos. Su propósito es aplicar y fortalecer habilidades en el manejo de estructuras avanzadas de datos, como árboles AVL, ABB y B. En esta segunda fase, se busca reemplazar las estructuras de datos empleadas en la primera fase de consola con estructuras más robustas y eficientes que permitan realizar operaciones como la búsqueda, inserción, y eliminación de manera óptima.

El proyecto implica la implementación de un sistema de usuarios mediante un árbol AVL, la gestión de publicaciones con un árbol ABB, y la administración de comentarios con un árbol B. Al finalizar, los estudiantes habrán reforzado tanto sus conocimientos técnicos en estructuras de datos avanzadas como sus habilidades en resolución de problemas y trabajo en equipo.

### **3. Objetivos del Aprendizaje**

#### **3.1 Objetivo General**

Implementar una red social funcional en C++ que utilice estructuras avanzadas de datos, como árboles AVL, ABB y B, para la optimización de operaciones comunes en aplicaciones de redes sociales, tales como la gestión de usuarios, publicaciones y comentarios.

#### **3.2 Objetivos Específicos**

1. **Desarrollar la habilidad de implementar un árbol AVL** para almacenar y gestionar eficientemente los usuarios de la red social, garantizando el balance de la estructura.
2. **Implementar un árbol ABB** para las publicaciones, optimizando las operaciones de búsqueda y acceso a los datos.
3. **Implementar un árbol B** para almacenar y manejar comentarios en las publicaciones, garantizando el acceso rápido y la eficiencia en el manejo de datos.
4. **Refinar las habilidades de depuración y testing en C++**, asegurando la fiabilidad de cada estructura de datos implementada.
5. **Integrar las estructuras de datos en una aplicación completa**, enlazando las funcionalidades de la primera fase con las nuevas estructuras optimizadas.

### **4. Enunciado del Proyecto**

El proyecto consiste en el diseño y desarrollo de una aplicación de escritorio en C++ que permita gestionar relaciones de amistad entre usuarios y proporcionar recomendaciones basadas en esas relaciones. El problema a resolver es la falta de una herramienta eficiente que facilite la administración de conexiones sociales, permitiendo a los usuarios agregar, eliminar y visualizar amigos de manera sencilla, así como recibir sugerencias para expandir su red de contactos.

**Alcance del Proyecto**

Esta fase del proyecto requiere la transformación de la aplicación de consola básica en una estructura optimizada que permita una experiencia de usuario más eficiente. Cada sección de la red social estará respaldada por una estructura de datos avanzada que permitirá realizar operaciones de alta complejidad de forma eficiente. El alcance incluye la implementación de:

* Un árbol AVL para la gestión de usuarios (alta, baja y modificaciones).
* Un árbol ABB para gestionar las publicaciones y facilitar búsquedas de publicaciones específicas.
* Un árbol B para la administración de los comentarios asociados a cada publicación.

**Funcionalidades Principales:**

* Creación, eliminación y visualización de relaciones de amistad.
* Ver publicaciones y realizar comentarios
* Filtrar publicaciones por fecha
* Reportes del administrador y usuario

**4.3 Requerimientos técnicos**Los estudiantes deberán utilizar C++ como lenguaje de programación principal para el desarrollo del proyecto. Se espera que integren Qt como plataforma para crear una interfaz gráfica de usuario (GUI), lo que permitirá una experiencia de usuario intuitiva y efectiva. Reemplazando estructuras utilizadas anteriormente por árboles(ABB, AVL y B) para mejorar la eficiencia de búsqueda e inserción de datos.

Adicionalmente, se recomienda el uso de herramientas de control de versiones como Git para mantener un historial del desarrollo del proyecto y facilitar la documentación técnica.

**4.4 Entregables**Los estudiantes deberán entregar los siguientes elementos en un repositorio de GitHub:

* **Prototipo funcional de la aplicación:** Se debe entregar un archivo ejecutable (por ejemplo, un archivo .exe) del sistema desarrollado, junto con el código fuente.
* **Informe Técnico:** Un informe de no más de 10 páginas que describa el proceso de desarrollo, los retos encontrados y cómo se resolvieron.
* **Documentación del usuario:** Manual de uso del sistema con capturas de pantalla.

### **5. Metodología**

1. **Análisis de requisitos**: Revisión de los requisitos y planificación de la implementación de las estructuras de datos.
2. **Diseño de la solución**: Definición del diseño de las estructuras, detallando el tipo de operaciones y su justificación.
3. **Desarrollo e implementación**: Codificación y pruebas unitarias de cada estructura de datos.
4. **Integración**: Conexión de cada estructura a la lógica de la aplicación de red social.
5. **Pruebas y ajuste final**: Validación de las operaciones y ajuste de errores encontrados.
6. **Documentación**: Redacción del informe técnico y la documentación de cada estructura.

### **6. Desarrollo de Habilidades Blandas**

Para complementar el desarrollo técnico, esta sección se centra en las habilidades blandas que los estudiantes deberán mejorar a lo largo del proyecto individual.

**6.1 Proyectos Individuales**

Los proyectos individuales permiten a los estudiantes desarrollar autonomía y responsabilidad sobre su propio trabajo. En este tipo de proyectos, cada estudiante debe gestionar su tiempo, investigar, planificar y ejecutar todas las fases del proyecto de manera independiente.

**6.1.1 Autogestión del Tiempo**

Los estudiantes deben crear un cronograma personal para cumplir con los plazos establecidos. Esto les ayuda a mejorar su disciplina y capacidad de priorización, habilidades esenciales en cualquier entorno profesional.

**6.1.2 Responsabilidad y Compromiso**

En un proyecto individual, el estudiante asume la totalidad de las responsabilidades, desde la investigación hasta la entrega final. Esto fomenta el sentido de compromiso y permite una mayor personalización en la solución del problema planteado.

**6.1.3 Resolución de Problemas**

Trabajar de manera independiente impulsa a los estudiantes a buscar soluciones de forma creativa y a enfrentarse a los desafíos sin depender del apoyo constante de otros. Esto fortalece su capacidad para resolver problemas de manera autónoma.

**6.1.4 Reflexión Personal**

Al concluir el proyecto, el estudiante realiza una autoevaluación, reflexionando sobre sus decisiones, lo aprendido y las áreas en las que podría mejorar. Esta práctica promueve el desarrollo continuo y el autoaprendizaje.

### **7. Cronograma**

| Tipo | Fecha Inicio | Fecha Fin |
| --- | --- | --- |
| Asignación de Proyecto | Semana 5 | Semana 5 |
| Elaboración | Semana 5 | Semana 9 |
| Calificación | Semana 10 | Semana 10 |

Este **cronograma** es un ejemplo general y puede ser ajustado según las necesidades y el contexto del proyecto.

### **8. Evaluación**

### **Resumen de Puntuaciones**

| **Área** | **Puntos Totales** | **Puntos Obtenidos** |
| --- | --- | --- |
| Aplicación | 17 |  |
| Usuarios | 11 |  |
| Solicitudes de amistad | 05 |  |
| Publicaciones | 14 |  |
| Comentarios | 16 |  |
| Reportes | 27 |  |
| Documentación | 10 |  |
| **Total** | **100** |  |

### **Detalle de la Calificaciòn**

| **Criterio** | **Descripción** | **Puntos Máximos** | **Puntuación Obtenida** |
| --- | --- | --- | --- |
| **1. Aplicación** |  | **17** |  |
| 1.1 Funcionalidad | Grado en que la aplicación cumple con los requisitos básicos y funciona correctamente. |  |  |
| 1.2 Interfaz de Usuario | Calidad de la interfaz de usuario, facilidad de uso, y experiencia general del usuario. |  |  |
|  |  |  |  |
| **2. Usuarios** |  | **11** |  |
| 2.1 Gestión de Usuarios | Implementación de funcionalidades de la fase anterior y haber utilizado el árbol AVL para almacenar los usuarios. |  |  |
| 2.2 Experiencia de Usuario | Calidad y efectividad en la interacción del usuario con el sistema. |  |  |
|  |  |  |  |
| **3. Solicitudes de amistad** |  | **05** |  |
| 3.1 Gestión de relaciones | Implementación de pila, lista y de matriz dispersa. |  |  |
|  |  |  |  |
| **4. Publicaciones** |  | **14** |  |
| 4.1 Implementación | Arbol BST para el manejo de publicaciones |  |  |
| 4.2 Funcionalidades | Crear y eliminar una publicación. Opción de agregar imagen al crear una publicación |  |  |
|  |  |  |  |
| **4. Comentarios** |  | **16** |  |
| 4.1 Funcionalidad | Crear comentarios por publicación. |  |  |
| 4.2 Implementación | Utilizar un árbol B para almacenar los datos de los comentarios. |  |  |
|  |  |  |  |
| **5. Reportes** |  | **27** |  |
| 5.1 Generación de Reportes | Funcionalidad y efectividad en la generación de reportes de actividad o uso dentro de la aplicación. |  |  |
| 5.2 Precisión y Claridad | Claridad en la presentación de los datos y precisión en la información presentada en los reportes. |  |  |
|  |  |  |  |
| **7. Documentación** |  | **10** |  |
| 7.1 Manual de Usuario | Claridad y detalle en el manual de usuario, con guías de uso y resolución de problemas comunes. |  |  |
| 7.2 Manual Técnico | Calidad y detalle en la documentación técnica, incluyendo diagramas y explicación de la arquitectura. |  |  |