Especificaciones Técnicas
Pizarra Colaborativa para Computólogos

Equipo 1

Diego Sebastián Sánchez Correa

Mauro Emiliano Chávez Zamora

Ulises Josué Anaya Pérez

Daniel Linares Gil

Karyme Ivette Azpeitia García

Creado: 15/02/2024
Última vez actualizado: 18/02/2024

Resumen

Se pretende proporcionar una herramienta básica que busca satisfacer las necesidades esenciales para el desarrollo de proyectos relacionados con las ciencias de la computación.

La idea de un editor de texto con *syntax highlighting* es común dentro del ámbito de desarrollo de software, sin embargo, se carece de herramientas de planeación colaborativa donde las ideas se puedan aterrizar de manera general o con detalles tan específicos como los usuarios lo deseen.

Esta aplicación tiene como fin la creación de un ambiente colaborativo y cuya esencia funja como punto de partida para la elaboración de decisiones de diseño a partir de diagramas, control de versiones, edición con dibujo vectorizado y soporte para LaTeX.

Glosario

Contexto

Ejemplos de herramientas de pizarra colaborativa

- Miro
- Mural
- Microsoft Whiteboard
- Figjam
- Jamboard

Ejemplos de uso:

- **Entrevista de programación:** Alice se encuentra entrevistando candidatos para unirse a su equipo como desarrolladores de software. Alice decide utilizar nuestra pizarra colaborativa porque permite bloques de código, dibujar de manera libre y escribir diagramas para explicar la

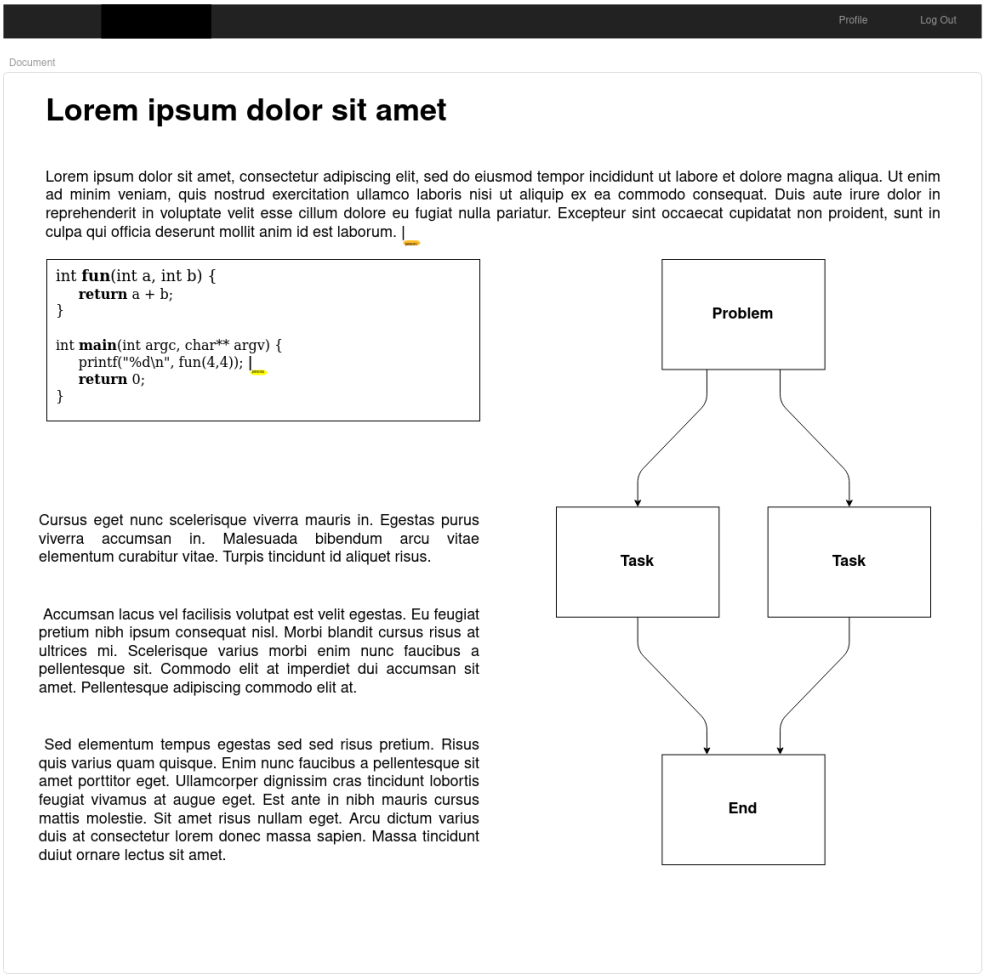
solución. Durante la entrevista, el candidato escribe su solución en el bloque de código y además puede utilizar las funcionalidades de diagramado y dibujo libre para explicar su solución a Alice.

- **Reunión de retrospectiva:** Alice se prepara para la reunión de retrospectiva de su equipo de desarrollo luego de un sprint. Alice crea una nueva pizarra, y la divide en tres columnas:
  - ¿Qué fue bien?
  - ¿Qué no fue bien?
  - ¿Qué se puede mejorar?

  Una vez termina de preparar la pizarra, crea un enlace para compartir la pizarra y lo comparte con los miembros de su equipo. Durante la reunión, los miembros del equipos colaboran añadiendo notas en cada columna. Los miembros del equipo pueden incluir enlaces a los tickets de Jira correspondientes en las notas. Una vez terminan de añadir notas, los miembros del equipo discuten sobre estas.

1. **Integración de Diagramas y Codificación**
   Provee un ambiente donde los desarrolladores pueden crear diagramas y escribir código en el mismo lugar, facilitando con ello la visualización y la implementación simultánea de ideas.

   Desde conceptos matemáticos simples como la incorporación de autómatas, hasta la integración de diagramas de flujo permitirán que los usuarios puedan ser capaces de definir las bases, algoritmos y diseño de un proyecto cuya esencia tenga una naturaleza colaborativa.



- **Planeación de proyectos de software**: al adjuntar código que esté relacionado con diagramas de flujo que representen el problema a resolver y las tareas (o proposiciones) asociadas para su compleción, o que denoten cómo estará organizado las funciones del programa (incluyendo con ello descripciones breves).
- **Editor de notas**: La organización en diagramas ayuda a que funja como una aplicación para tomar notas cómoda y fácil de usar. Además, la incorporación de formatos para código, hace de esta una herramienta esencial para estudiantes de carreras que estén relacionadas con el desarrollo de software.
- **Depuración de código colaborativo**: La visualización colaborativa en diagramas del código escrito trae consigo una depuración rápida y básica para código no compilado que pretende plantear una idea inicial para dar solución a un problema.

2. **Colaboración Concurrente**
   Motivación: El objetivo de la pizarra es fomentar la colaboración y el intercambio de ideas entre usuarios, para lo cual es fundamental que la pizarra permita la colaboración en tiempo real entre los usuarios.

   Objetivo: Permitir que múltiples usuarios trabajen de manera simultanea en la pizarra.

   - **Compartir con enlace:** El usuario debe poder generar un enlace único para compartir la pizarra con otros usuarios.

     Solo los usuarios autenticados deben poder acceder a los enlaces. Si un usuario que no se encuentra autenticado o no posee una cuenta intenta acceder a un enlace, será dirigido al flujo de autenticación/registro y luego a la pizarra.

     El usuario debe poder elegir el nivel de acceso que se concede a los usuarios con el enlace:

     - **Solo lectura:** Permite a los usuarios con el enlace ver la pizarra, pero no realizar cambios.
     - **Edición:** Permite a los usuarios con el enlace ver y editar la pizarra.

     El usuario debe poder revocar el acceso a la pizarra en cualquier momento.
   - **Visualización de la colaboración:** Los usuarios que se encuentren trabajando en una pizarra deben poder visualizar en tiempo real las modificaciones realizadas a la pizarra por otros usuarios.
     - **Indicador de presencia:** Los usuarios deben poder ver en la barra superior los avatares de los usuarios que están conectados a la pizarra.
     - **Cursor en tiempo real:** Los usuarios deben poder ver los cursores de los demás usuarios en la pizarra para indicar dónde están trabajando.

3. **Guardado Automático y Control de Versiones**
   Estas funcionalidades son esenciales para garantizar la seguridad de los datos, facilitar la colaboración y mejorar la productividad de los usuarios. A continuación se describen las especificaciones funcionales.

   - **Guardado Automático**
     El guardado automático protegecontra la pérdida de datos en caso de fallos del sistema, errores del usuario o eventos inesperados.
     - **Frecuencia:** El trabajo del usuario en la pizarra se debe guardar automáticamente con una frecuencia configurable.
       Opciones de frecuencia recomendadas:
       * Cada 5 minutos
       * Cada 10 minutos
       * Cada 15 minutos
       * Personalizado
     - **Ubicación del archivo:** Los archivos guardados automáticamente se deben almacenar en un servidor seguro y accesible para el usuario.
       * Opciones de ubicación recomendadas:
         · Nube privada(por ejemplo, Google Drive, Dropbox)
         · Servidor local
         · Almacenamienro propio de la aplicación

- **Notificaciones:**El usuario debe recibir una notificación cada vez que se guarde automáticamente su trabajo.
    - La notificación indicará: Fecha y hora del guardado, asi como ubicación del archivo guardado.

- **Control de Versiones**
  El control de versiones permite a los usuarios trabajar juntos en la misma pizarra virtual de forma eficiente. Permite rastrear los cambios, deshacer errores y restaurar versiones anteriores.
    - **Historial de versiones:**El usuario debe poder acceder al historial de versiones y ver las diferencias entre cada versión.
        - Opciones para acceder al historial de versiones:
            - Lista de versiones con fecha y hora
            - Miniaturas visuales de cada versión
            - Deslizador para comparar dos versiones
    - **Restauración de versiones:** El usuario debe poder restaurar una versión anterior de la pizarra en cualquier momento.
        - Opciones para restaurar una versión:
            - Seleccionar una versión de la lista de versiones
            - Usar un atajo de teclado
            - Botón de "Restaurar" en la barra de herramientas
    - **Revertir cambios:** El usuario debe poder deshacer los últimos cambios realizados en la pizarra.
        - Opciones para deshacer cambios:
            - Botón de "Deshacer" en la barra de herramientas
            - Atajo de teclado

4. **Pizarra Infinita**
   Motivación: Al momento de trabajar en múltiples pizarras puede ser sencillo que se pierda información relevante durante los cambios de pizarra, si bien lo ideal sería que los temas pudieran agruparse en una pizarra en específico cada uno, hay temas que requerirán más espacio debio a los múltiples elementos que pueden componer una tarea/tema con muchos componentes o pasos.

   Objetivo: Que nuestra aplicación pueda simular una pizarra que nunca se termina conforme las personas van aumentando los elementos que componen su proyecto (notas, inserciones de Latex, etc.) Si bien el generar una pizarra infinita es imposible de manera práctica, deseamos que el espacio brindado sea al menos lo suficientemente grande para que la persona usuaria tenga la sensación de que no estará limitada por las herramientas.

    - **Opciones para implementarla:**
        - Que la aplicación tenga un umbral donde detecte qué tanto algunos elementos se han alejado del centro inicial de la pantalla para irle ofrecienco espacio al usuario conforme incorpora más elementos.
        - Que el usuario pueda solicitar espacio a demanda conforme elige qué sitios del proyecto conviene ampliar en una determinada dirección, de manera que las personas pueda elegir qué tanto van a requerir de la aplicación.

- Una combinación de ambas, donde sólo se amplie aquellos lugares de la pizarra donde se estén agregando elementos que se acerquen al límite establecido hasta el momento.

Como características adicionales, podemos lanzar un aviso cuando una pizarra esté siendo demasiado grande de manera que se sugiera al usuario segmentar el tema para llevar un mejor control del mismo.

5. **Registro de Usuarios**
   - **Objetivo:** Permite que los desarrolladores puedan crear su cuenta con un correo electronico y contrasena, de esta manera podran acceder a su pizarra y a sus datos guardados.
   - **Funcion:** Los usuarios deben ser capaces de poder registrarse en la apliacion web de una manera comoda , incluyendo su nombre de usuario, nombre de la persona , contrasena , segundo factor de autenticacion , ademas de incluir una forma de recuperar su cuenta , en caso de que el usuario pierda su contrasena.
   - **Seguridad:** El sitio debe de ser seguro , ya que se tienen que mantener las tres propiedades de la seguridad:
     - **Confidencilidad:** La informacion solo puede ser vista por aquellos que tienen permisos para verla.
     - **Integridad:** La informacion no debe de ser alterada o modificada de manera deliberada.
     - **Disponibilidad:** La informacion debe de estar disponible cuando se necesite.
     - **Prevencion de inyecciones SQL:** Es fundamenta que el sistema de registro de usuarios este protegido contra inyecciones SQL, ya que en caso de no serlo se verian afectadas las propiedades mencionadas anterioemente.
     - **Almacenamiento de contrasenas seguro con hashing:** El hashing convierte la contraseña en una cadena de caracteres irreconocible mediante un algoritmo de hash. Cuando un usuario intenta iniciar sesión, la contraseña proporcionada se vuelve a hashear y se compara con la versión hasheada almacenada en la base de datos. Esto significa que incluso si la base de datos es comprometida, las contraseñas de los usuarios no estarán expuestas en texto plano.
     - **Limitar intentos de inicio de sesion:** Implementar mecanismos para limitar el número de intentos de inicio de sesión fallidos puede ayudar a prevenir ataques de fuerza bruta y de diccionario.
   - **Experiencia de usuario:**El registro por parte de los usuarios debe ser comodo, intuitivo y facil de usar, ademas de tener buena apariencia.

6. **Soporte de LaTeXy graficación de funciones**
   Integra herramientas que permiten la creación y graficación de fórmulas de LaTeX, facilitando la representación visual de información compleja.

7. **Funcionalidades Avanzadas de Colaboración**
   Facilita la interacción de usuarios a través de funcionalidades como comentarios, menciones y notificaciones en tiempo real; promoviendo un ambiente de trabajo colaborativo y eficiente.

Objetivos fuera del alcance

Objetivos futuros

Suposiciones

Pantallas