HERRAMIENTAS DE DESARROLLO

Trabajo final

Remigio Huarcaya Almeyda

Docente

2025

Proyecto Final

Objetivo

Diseñar y construir una aplicación web con Node.js con el framework React y el gestor de base de datos MySQL usando el flujo de trabajo colaborativo en GitHub.

Planteamiento del proyecto

Sistema de Gestión de Recursos Educativos

El proyecto consiste en desarrollar una aplicación web que permita a docentes de la UTP gestionar recursos educativos digitales.

La plataforma servirá como repositorio centralizado donde los docentes pueden crear, compartir, organizar y acceder a diversos materiales educativos como documentos, videos, enlaces, y ejercicios. Los estudiantes son usuarios que pueden descargar los materiales.

Funcionalidades Principales

1. Gestión de Usuarios

- **Registro y autenticación**: Programa de login con roles diferenciados (administrador, docente, estudiante)
- **Perfiles personalizables**: Información básica, áreas de interés, historial de actividad
- **Gestión de permisos**: Control granular sobre qué usuarios pueden crear, editar o eliminar recursos

2. Gestión de Recursos Educativos

- **Creación de recursos**: Formularios para subir o enlazar diferentes tipos de materiales
- **Categorización**: Organización por curso.
- **Búsqueda avanzada**: Filtros múltiples y búsqueda por texto completo

4. Panel Administrativo

- **Estadísticas de uso**: Visualización de datos sobre recursos más populares
- **Moderación de contenido**: Revisión y aprobación de recursos subidos
- **Gestión de usuarios**: Administración de cuentas y roles

Requisitos Técnicos

Frontend

- Interfaces responsivas y accesibles
- Formularios validados del lado del cliente
- Previsualización de recursos cuando sea posible

Backend (Node.js)

- API RESTful para todas las operaciones
- Autenticación mediante JWT o sesiones
- Validación exhaustiva de datos de entrada
- Gestión de archivos y almacenamiento seguro

Base de Datos (MySQL)

- Diseño normalizado de tablas para usuarios, recursos, categorías, etc.
- Optimización para búsquedas frecuentes
- Integridad referencial mediante llaves foráneas
- Construir procedimiento almacenados pertinentes

Consideraciones de Desarrollo

- La aplicación debe ser escalable para manejar potencialmente miles de recursos
- Debe implementar buenas prácticas de seguridad (prevención de SQL injection, XSS, etc.)
- El código debe estar bien documentado siguiendo estándares JSDoc

- Las interfaces deben ser intuitivas y accesibles según estándares WCAG

Entregable Final

Una aplicación web completamente funcional donde los docentes puedan:

1. Registrarse y gestionar su perfil
2. Subir y categorizar recursos educativos
3. Buscar y filtrar recursos existentes
4. Organizar recursos en colecciones temáticas
5. Administrar permisos y accesos (si tienen rol administrativo)

Desarrollar actividades de GitHub en el desarrollo del proyecto

Se debe utilizar el GitHub durante el desarrollo del proyecto, y registrar las evidencias de las siguientes actividades:

1. **Gestión de Repositorios**

   Elaborar la configuración Inicial

   - **Creación del repositorio**: Los alumnos deben crear un repositorio con la estructura básica del proyecto.
   - **README.md**: Documentación detallada con badges de estado del proyecto, instrucciones de instalación, y descripción del proyecto.
   - **.gitignore**: Configuración adecuada para Node.js, evitando subir node_modules, archivos .env, y logs.

   Realizar la colaboración del repositorio

   - **Fork del repositorio**: Un alumno puede hacer fork del repositorio inicial y el otro colaborar mediante invitación directa.
   - **Configuración de colaboradores**: Gestión de permisos para cada miembro del equipo.

2. **Flujo de Trabajo con Ramas (Branching)**

   Plantear estrategia de Branching y buenas prácticas

- **Implementación de GitFlow**: Estructura de ramas:
    - main: Código en producción
    - develop: Rama de integración
    - feature/*: Para nuevas funcionalidades
    - bugfix/*: Para corrección de errores
    - release/*: Para preparación de versiones

Ejecutar comandos y operaciones

- **Creación de ramas**: git checkout -b feature/nombre-funcionalidad
- **Cambio entre ramas**: git checkout [nombre-rama]
- **Actualización de ramas**: git pull origin develop
- **Visualización de ramas**: git branch -a y git log --graph --oneline --all

3. **Pull Requests y Code Reviews**

Con respecto a la creación y gestión

- **Apertura de PR**: Crear pull requests desde ramas de características hacia develop.
- **Plantillas de PR**: Implementar plantillas con campos para descripción, tipo de cambio, y testing realizado.
- **Draft PRs**: Utilizar PRs en modo borrador para trabajo en progreso.

Desarrollar Procesos de Revisión

- **Asignación de revisores**: Designar al compañero como revisor obligatorio.
- **Comentarios en línea**: Proporcionar feedback específico sobre el código.
- **Discusiones y resolución**: Utilizar las conversaciones dentro del PR para aclarar dudas.

- **Aprobación y rechazo**: Establecer criterios claros para la aprobación de cambios.

4. **Issues y control de proyecto usando Trello**

   Gestión de Issues

   - **Creación estructurada**: Usar plantillas para reportar bugs, solicitar funcionalidades o documentar tareas.
   - **Etiquetas (labels)**: Clasificar issues por tipo (bug, enhancement, documentation).
   - **Hitos (milestones)**: Agrupar issues en entregables específicos (v1.0, Autenticación, API REST).
   - **Asignaciones**: Distribuir tareas entre los miembros del equipo.

   Construir el tablero de Proyecto

   - **Configuración de proyecto**: Crear un tablero en Trello con columnas (Pendiente, En Progreso, Revisión, Completado).
   - **Automatizaciones**: Configurar movimiento automático de tarjetas según el estado de issues y PRs.
   - **Seguimiento de progreso**: Utilizar la vista de milestone para visualizar el avance del proyecto.

5. **GitHub Actions (CI/CD)**

   Integración Continua

   La Integración Continua (CI) es la práctica de integrar los cambios de código de varios desarrolladores en un repositorio central de forma frecuente y automatizada. Cada vez que un desarrollador realiza un cambio en el código y lo sube al repositorio, se desencadena automáticamente una serie de pruebas y compilaciones para verificar que el nuevo código sea compatible con el resto del proyecto y no introduzca errores.

   - **Configuración de workflows**: Crear archivo .github/workflows/ci.yml para:

- Ejecución automática de tests
- Análisis de código con ESLint
- Verificación de seguridad con dependabot

Despliegue Continuo

El despliegue continuo (CD), en el contexto de CI/CD, es la automatización de la publicación de cambios de código a producción después de que pasan pruebas y verificaciones automatizadas. Es la última etapa de un proceso de CI/CD, que incluye la integración continua, la entrega continua y el despliegue continuo

- **Configuración de CD**: Automatizar el despliegue a un entorno de staging al fusionar en develop.
- **Badges de estado**: Mostrar el estado de los workflows en el README.md.

6. **Gestión de Versiones y Releases**

   Elaborar Tags y Releases

   - **Etiquetado semántico**: Usar git tag siguiendo SemVer (ej: v1.0.0).
   - **Notas de lanzamiento**: Documentar cambios importantes para cada release.
   - **Assets**: Incluir archivos compilados o documentación en los releases.

7. **Documentación**

   Utilizar gitHub Pages

   - **Configuración**: Habilitar GitHub Pages para alojar documentación.
   - **JSDoc**: Generar documentación automática del código y publicarla.

8. **Seguridad y Calidad**

   Debe considerar herramientas de Análisis:

- **Dependabot**: Configuración para alertas de dependencias vulnerables.
- **CodeQL**: Análisis automático de seguridad del código.

Establecer protección de Ramas

- **Branch protection rules**: Configurar protecciones para main y develop:
  - Requerir aprobaciones de PR
  - Requerir checks de status (tests pasados)
  - Prohibir push directo

9. **Colaboración Avanzada**

Establecer discusiones

- **Foros de debate**: Utilizar **Discussions** para decisiones de diseño o arquitectura.
- **Anuncios**: Comunicar actualizaciones importantes del proyecto.

Gists

- **Compartir snippets**: Utilizar Gists para compartir fragmentos de código útiles.

10. **Resolución de Conflictos**

Aplicar técnica de manejo de conflicto

- **Identificación**: Reconocer conflictos en archivos.
- **Resolución manual**: Editar archivos para combinar cambios.
- **Herramientas**: Usar herramientas como git mergetool.
- **Prevención**: Estrategias para minimizar conflictos mediante comunicación efectiva.

Recurso

Aplicación básica de NodeJs con MySQL

ANEXO 01

Documentación de Github

https://docs.github.com/es

Issue

Un "issue" (o "incidencia") es una herramienta para rastrear, planificar y colaborar en el desarrollo de un proyecto. Sirve para registrar errores, sugerir mejoras, discutir ideas o solicitar tareas, facilitando la comunicación y el seguimiento del trabajo entre el equipo.

https://docs.github.com/es/issues/tracking-your-work-with-issues/about-issues

Git Actions

Con GitHub Actions se automatiza, personaliza y ejecuta tus flujos de trabajo de desarrollo de software directamente en tu repositorio. Puedes descubrir, crear y compartir acciones para realizar cualquier trabajo que quieras, incluido CI/CD, y combinar acciones en un flujo de trabajo completamente personalizado

https://docs.github.com/es/actions

Documentación GitHub Discussions

GitHub Discussions es un foro de comunicación colaborativa para la comunidad que circunda un proyecto interno o de código abierto. Los miembros de la comunidad pueden hacer preguntas y proporcionar respuestas, compartir actualizaciones, tener conversaciones abiertas y dar seguimiento a las decisiones que afectan la forma de trabajar de la misma.

Git mergetool

Git mergetool es una herramienta integrada en Git que facilita la resolución de conflictos en archivos que han sido modificados por múltiples ramas al mismo tiempo.

Permite comparar y fusionar cambios de manera visual, utilizando una interfaz gráfica que ayuda a resolver conflictos de manera más intuitiva que la línea de comandos estándar de Git. Al ejecutar git mergetool, Git utiliza una herramienta de comparación y fusión configurada en tu sistema (como Meld, KDiff3, o

cualquier otra herramienta compatible) para ayudarte a manejar los conflictos de manera más eficiente.

Dependabot

Es una herramienta de GitHub que automatiza la actualización de dependencias de software, tanto para seguridad como para versiones más recientes

Gist

Un gist es una forma que se tiene para poder compartir código usando GitHub. Cuando tenemos un código que **no es lo suficientemente grande** para crear un repositorio, entonces creamos un **gist**. Aunque gist funciona básicamente cómo un repositorio, ya que se le puede hacer un fork o clonarlo. Así mismo se puede editar y las personas pueden comentar el gist.

https://gist.github.com/renatojobal/cebf28c1f7d941f18aa7129074c5ac2d

CodeQL

CodeQL es un motor de análisis de código estático que utiliza un lenguaje de consulta (CodeQL) para analizar el código fuente en busca de posibles vulnerabilidades de seguridad y otros problemas.

Se trata de una herramienta desarrollada por GitHub, que permite a los desarrolladores realizar un análisis de código más profundo y preciso que los métodos tradicionales de revisión de código

ANEXO 02

JSDoc es una herramienta de documentación para JavaScript que permite agregar comentarios tipados y estructurados a tu código. Similar a JavaDoc para Java, **JSDoc** no solo ayuda a documentar tu código, sino que también mejora la experiencia de desarrollo con autocompletado e información de tipos en editores modernos como Visual Studio Code.

¿Por qué usar JSDoc?

- **Mejora la mantenibilidad**: Facilita entender el código meses después
- **Autocompletado inteligente**: Los IDEs pueden proporcionar sugerencias más precisas
- **Documentación automática**: Genera documentación HTML a partir de comentarios
- **Validación de tipos**: Proporciona verificación de tipos sin necesidad de TypeScript
- **Compatibilidad**: Funciona con JavaScript vanilla y con frameworks modernos