2025

Libro de Visitas

JEAN ANDRADE, DAVID TOBAR

Descripción del Proyecto

La miniaplicación “Libro de Visitas” permite a los usuarios registrarse e ingresar mensajes breves. Cada mensaje se almacenará con el nombre del usuario visitante, su contenido, fecha y hora. La información se vincula a una tabla de usuarios y otra de mensajes, estableciendo una relación uno a muchos.

CI/CD al Proyecto “Libro de Visitas”

Relación con CI/CD:

- **Integración Continua (CI):** Permite integrar y verificar automáticamente cada cambio en el código, como el registro de visitantes (nombre y correo) y el envío de mensajes (contenido limitado a 300 caracteres).
- **Entrega Continua (CD):** Toma ese código probado y lo despliega automáticamente en un entorno web accesible para pruebas y luego para producción.
- Juntas, CI y CD aseguran que el sistema esté siempre actualizado, funcional y sin errores.

Etapas de CD:

1. **Validación automática** del código cuando se realiza un cambio o actualización.
2. **Empaquetado del proyecto**, listo para despliegue, usando herramientas como GitHub Actions.
3. **Despliegue en entorno de pruebas** para simular el uso real del sistema.

4. **Verificación funcional**, asegurando que los visitantes se registren y que los mensajes se guarden y muestren correctamente.
5. **Despliegue final** al entorno real, permitiendo el acceso público.

Beneficios:

- Se reducen los errores al validar cada parte del sistema antes de publicarlo.
- El despliegue es automático, ahorrando tiempo y esfuerzo manual.
- Se recibe retroalimentación rápida ante fallos o mejoras.
- Facilita la colaboración entre desarrolladores con procesos definidos.
- Aporta valor continuo al usuario con nuevas funciones o mejoras disponibles de inmediato.

Herramientas utilizadas

GitHub:

Se utilizó para llevar el control de versiones del proyecto, permitiendo registrar los cambios en el código, colaborar de forma ordenada y mantener una copia centralizada y segura del repositorio.

GitHub Actions:

Automatizó el proceso de CI/CD. Cada vez que se actualiza el código, GitHub Actions ejecuta tareas como validación, pruebas y despliegue automático. Esto ayuda a mantener el sistema funcional sin necesidad de procesos manuales.

Node.js:

Fue el entorno de ejecución para el backend. Permite crear el servidor y manejar las funcionalidades principales del sistema (registro de usuarios y mensajes), usando JavaScript de forma eficiente.

MySQL Workbench:

Se utilizó para diseñar, administrar y consultar la base de datos. Facilitó la creación de tablas y relaciones necesarias para guardar la información de los visitantes y sus mensajes.

Capturas de uso de GitHub Actions

```yaml
name: Fullstack CI

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  backend:
    name: Backend - Node.js
    runs-on: ubuntu-latest

    steps:
      - name: Checkout código    "código": Unknown word.
        uses: actions/checkout@v3

      - name: Instalar Node.js    "Instalar": Unknown word.
        uses: actions/setup-node@v3
        with:
          node-version: '20'

      - name: Instalar dependencias Backend    "Instalar": Unknown word.
        run: |
          cd Backend
```

Cuando se ejecuta (push/pull_request a main).

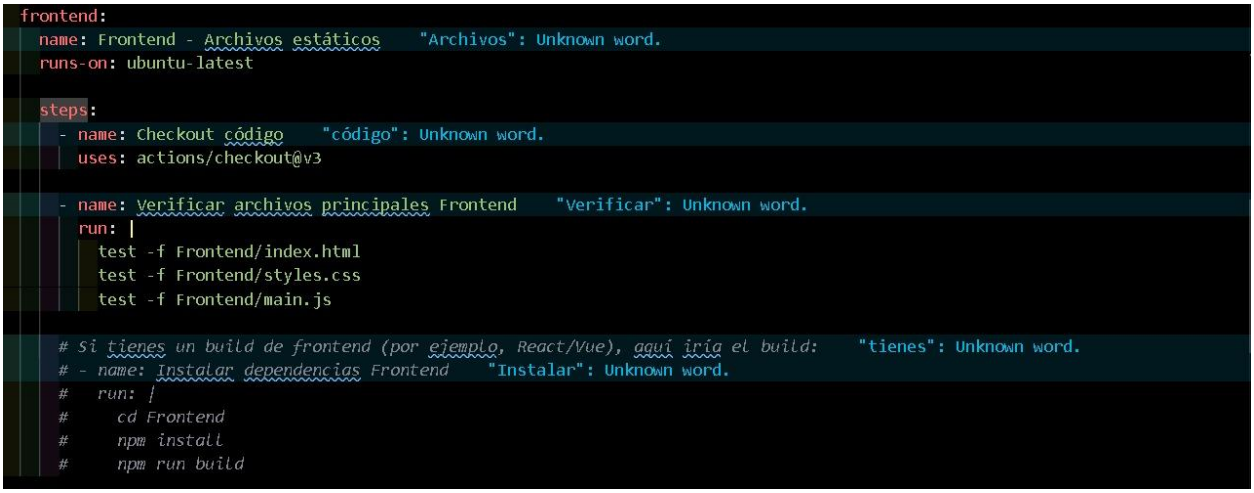
Jobs: Define los trabajos a ejecutar.

backend:Descarga el código.

Instala Node.js.

Instala dependencias del backend.

(Opcional) Ejecuta pruebas.Verifica que existan archivos clave.

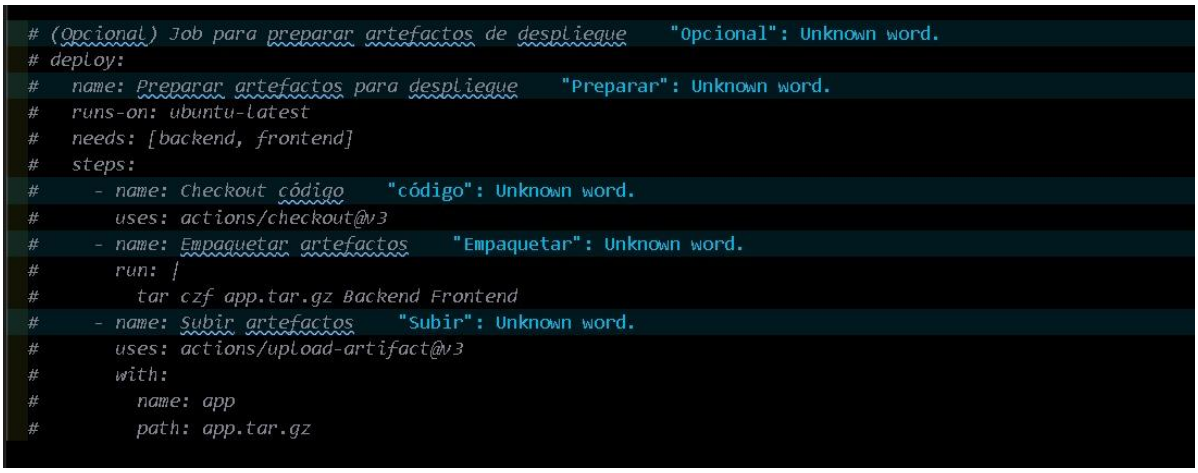


```yaml
frontend:
  name: Frontend - Archivos estáticos    "Archivos": Unknown word.
  runs-on: ubuntu-latest

  steps:
    - name: Checkout código    "código": Unknown word.
      uses: actions/checkout@v3

    - name: Verificar archivos principales Frontend    "Verificar": Unknown word.
      run: |
        test -f Frontend/index.html
        test -f Frontend/styles.css
        test -f Frontend/main.js

    # Si tienes un build de frontend (por ejemplo, React/Vue), aquí iría el build:    "tienes": Unknown word.
    # - name: Instalar dependencias Frontend    "Instalar": Unknown word.
    #   run: |
    #     cd Frontend
    #     npm install
    #     npm run build
```

frontend:Descarga el código.

Verifica que existan los archivos principales del frontend.

(Opcional) Instala dependencias y hace build si usas frameworks modernos.



```yaml
# (Opcional) Job para preparar artefactos de despliegue    "Opcional": Unknown word.
# deploy:
#   name: Preparar artefactos para despliegue    "Preparar": Unknown word.
#   runs-on: ubuntu-latest
#   needs: [backend, frontend]
#   steps:
#     - name: Checkout código    "código": Unknown word.
#       uses: actions/checkout@v3
#     - name: Empaquetar artefactos    "Empaquetar": Unknown word.
#       run: |
#         tar czf app.tar.gz Backend Frontend
#     - name: Subir artefactos    "Subir": Unknown word.
#       uses: actions/upload-artifact@v3
#       with:
#         name: app
#         path: app.tar.gz
```

deploy:Empaqueta y sube los artefactos para despliegue.