README

Backend Plantilla - Node.js + Express

Plantilla de backend con Node.js y Express para consumir APIs externas. Ideal para comenzar proyectos que requieran actuar como intermediario entre el frontend y APIs de terceros.

📋 Características

- ✅ Servidor Express configurado
- ✅ CORS habilitado
- ✅ Ejemplos de peticiones a API externa
- ✅ Manejo de errores
- ✅ Variables de entorno
- ✅ Estructura lista para expandir

🚀 Instalación

1. **Instalar dependencias:**

```
npm install
```

2. **Configurar variables de entorno (opcional):**

```
cp .env.example .env
```

Edita el archivo `.env` con tus configuraciones.

3. **Iniciar el servidor:**

   - **Modo desarrollo** (con recarga automática):

README

- **Modo producción:**

```
npm start
```

El servidor estará disponible en `http://localhost:3000`

📍 Endpoints Disponibles

`GET /`

Página principal con información sobre los endpoints disponibles.

`GET /api/ejemplo`

Ejemplo básico que consulta una API externa y devuelve los datos.

Respuesta:

```
{
  "success": true,
  "data": {
    "userId": 1,
    "id": 1,
    "title": "...",
    "body": "..."
  }
}
```

`GET /api/usuarios`

Obtiene una lista de usuarios de ejemplo.

`GET /api/usuario/:id`

Obtiene un usuario específico por ID.

Ejemplo: `GET /api/usuario/1`

README

Obtiene posts, opcionalmente filtrados por userId.

🔧 Cómo Adaptar para la API de Aemet

1. **Obtener API Key de Aemet:**
   - Regístrate en OpenData Aemet
   - Obtén tu clave API
2. **Agregar la clave al archivo `.env`:**

```
AEMET_API_KEY=tu_clave_aqui
```

3. **Crear un nuevo endpoint en `server.js`:**

README

```
    const { provincia } = req.params;
    const API_KEY = process.env.AEMET_API_KEY;

    // Ejemplo: Predicción por provincia
    const response = await fetch(
      `https://opendata.aemet.es/opendata/api/prediccion/provincia/${provincia}?
        api_key=${API_KEY}`
    );

    if (!response.ok) {
      throw new Error(`Error HTTP: ${response.status}`);
    }

    const data = await response.json();

    // La API de Aemet suele devolver una URL donde están los datos
    // Hay que hacer una segunda petición
    if (data.datos) {
      const datosResponse = await fetch(data.datos);
      const meteorologia = await datosResponse.json();

      res.json({
        success: true,
        data: meteorologia
      });
    } else {
      res.json(data);
    }

  } catch (error) {
    console.error('Error:', error.message);
    res.status(500).json({
      success: false,
      error: 'Error al obtener datos meteorológicos'
    });
  }
});
```

README

```
backend-plantilla/
├── node_modules/       # Dependencias (generado)
├── .env                # Variables de entorno (crear desde .env.example)
├── .env.example        # Ejemplo de variables de entorno
├── .gitignore          # Archivos ignorados por git
├── package.json        # Configuración y dependencias del proyecto
├── README.md           # Este archivo
└── server.js           # Servidor principal
```

Tecnologías Utilizadas

- **Node.js**: Entorno de ejecución de JavaScript
- **Express**: Framework web minimalista
- **CORS**: Middleware para habilitar CORS
- **dotenv**: Gestión de variables de entorno
- **nodemon**: Recarga automática en desarrollo

Ejercicios para Alumnos

1. Crear un nuevo endpoint que consulte otra API pública
2. Implementar un endpoint POST que envíe datos a una API
3. Agregar validación de parámetros
4. Implementar caché de respuestas
5. Crear endpoints para la API de Aemet:
    - Predicción por municipio
    - Estado del tiempo actual
    - Alertas meteorológicas

Recursos Útiles

- Documentación Express
- API Aemet OpenData
- Fetch API

README

- Los ejemplos actuales usan JSONPlaceholder como API de prueba
- Recuerda no subir el archivo `.env` a repositorios públicos
- Para producción, considera agregar rate limiting y más validaciones

---

¡Listo para comenzar a desarrollar! 🎉