

# Documentación de API - DynaDoc

---

**Base URL:** http://localhost:5116

**Formato:** JSON

Guía de integración para Frontend (cURL y JavaScript).

## 1. Autenticación

---

### Registro de Usuario

Crea una cuenta. **Guarda el ID retornado.**

**POST**     **/api/auth/register**

#### CURL

---

```
curl -X POST http://localhost:5116/api/auth/register \-H "Content-Type: application/json" \-d '{"Username": "testuser", "Password": "mypassword"}
```

#### JAVASCRIPT (FETCH)

---

```
const registerUser = async () => {  
  const res = await fetch('http://localhost:5116/api/auth/register', {  
    method: 'POST',  
    headers: { 'Content-Type': 'application/json' },  
    body: JSON.stringify({ Username: "testuser", Password: "mypassword" })  
  });  
  const data = await res.json();  
  console.log(data);  
};
```

## Iniciar Sesión

**POST**    **/api/auth/login**

### CURL

---

```
curl -X POST http://localhost:5116/api/auth/login \-H "Content-Type: application/json" \-d '{"Username": "testuser", "Password": "mypassword"}
```

### JAVASCRIPT (FETCH)

---

```
const login = async () => {  
  const res = await fetch('http://localhost:5116/api/auth/login', {  
    method: 'POST',  
    headers: { 'Content-Type': 'application/json' },  
    body: JSON.stringify({ Username: "testuser", Password: "mypassword" })  
  });  
  const data = await res.json();  
  return data; // Guarda el token o ID si aplica  
};
```

## 2. Gestión de Currículums (CRUD)

---

## Crear un nuevo CV

**POST**    **/api/cv**

### CURL

---

```
curl -X POST http://localhost:5116/api/cv \ -H "Content-Type: application/json" \ -d '{
"templateId": 1, "userId": 1, "nombre": "Maria", ... }'
```

### JAVASCRIPT (FETCH)

---

```
const createCv = async (cvData) => {
  const res = await fetch('http://localhost:5116/api/cv', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(cvData)
  });
  if (!res.ok) console.error("Error creando CV");
  return await res.json();
};

// Ejemplo de objeto cvData:
// { templateId: 1, userId: 1, nombre: "Maria", educacion: [], ... }
```

## Obtener todos los CVs

**GET**    **/api/Cv**

### JAVASCRIPT (FETCH)

---

```
const getAllCvs = async () => {
  const res = await fetch('http://localhost:5116/api/Cv');
  const data = await res.json();
  console.log(data);
};
```

## Obtener un CV por ID

**GET**    **/api/Cv/{id}**

### JAVASCRIPT (FETCH)

---

```
const getCvById = async (id) => {  
  const res = await fetch(`http://localhost:5116/api/Cv/${id}`);  
  const data = await res.json();  
  return data;  
};
```

## Actualizar un CV

El ID en la URL debe coincidir con el ID en el cuerpo del JSON.

**PUT**    **/api/Cv/{id}**

### JAVASCRIPT (FETCH)

---

```
const updateCv = async (id, updatedData) => {  
  const res = await fetch(`http://localhost:5116/api/Cv/${id}`, {  
    method: 'PUT',  
    headers: { 'Content-Type': 'application/json' },  
    body: JSON.stringify(updatedData)  
  });  
  
  if (res.status === 204) console.log("Actualizado con éxito");  
};
```

## Eliminar un CV

**DELETE**    **/api/Cv/{id}**

JAVASCRIPT (FETCH)

---

```
const deleteCv = async (id) => {  
  await fetch(`http://localhost:5116/api/Cv/${id}`, {  
    method: 'DELETE'  
  });  
};
```

## 3. Plantillas y PDF

---

### Listar Plantillas

**GET**    **/api/pdf/list**

JAVASCRIPT (FETCH)

---

```
const getTemplates = async () => {  
  const res = await fetch('http://localhost:5116/api/pdf/list');  
  return await res.json();  
};
```

## Generar y Descargar PDF

Este endpoint devuelve un archivo binario (Blob). Requiere un manejo especial en JS.

**POST**     **/api/pdf/generate/{id}**

### CURL

---

```
curl -X POST http://localhost:5116/api/pdf/generate/1 --output cv.pdf
```

### JAVASCRIPT (DESCARGA DE ARCHIVO)

---

```
const downloadPdf = async (cvId) => {
  try {
    const res = await fetch(`http://localhost:5116/api/pdf/generate/${cvId}`,
      method: 'POST'
    );

    if (res.ok) {
      // 1. Convertir respuesta a Blob
      const blob = await res.blob();

      // 2. Crear URL temporal
      const url = window.URL.createObjectURL(blob);

      // 3. Crear enlace invisible y forzar click
      const a = document.createElement('a');
      a.href = url;
      a.download = `cv_${cvId}.pdf`;
      document.body.appendChild(a);
      a.click();

      // 4. Limpiar
      a.remove();
      window.URL.revokeObjectURL(url);
    }
  } catch (err) {
    console.error("Error descargando PDF", err);
  }
};
```

---

Generado para el equipo de desarrollo de DynaDoc