### 1. ****Definir el Entorno de Desarrollo****

* **Backend**: Node.js con Express (o FastAPI si prefieres Python). https://nodejs.org/es
* **Base de Datos**: PostgreSQL (ideal para relaciones complejas).
* **Frontend**: React.js para crear una interfaz dinámica.
* **Autenticación**: Puedes usar OAuth2 con bibliotecas como Passport.js para Node.js.

### 2. ****Instalación de Herramientas****

Asegúrate de tener instalado lo siguiente:

* Node.js
* PostgreSQL
* Un editor de código (Visual Studio Code es una buena opción)
* Docker y Kubernetes (opcional, para la escalabilidad)

### 3. ****Estructura del Proyecto****

Crea la estructura de carpetas para tu proyecto:

Code

/military\_enrollment\_system  
/backend  
/models  
/routes  
/controllers  
/middleware  
/config  
/frontend  
/src  
/components  
/pages  
/services

### 4. ****Base de Datos****

Define tu base de datos. Aquí tienes un esquema básico para la tabla militares:

Code

CREATE TABLE militares (  
id SERIAL PRIMARY KEY,  
email VARCHAR(255) UNIQUE NOT NULL,  
nombres\_completos VARCHAR(255) NOT NULL,  
cedula VARCHAR(50) UNIQUE NOT NULL,  
telefono VARCHAR(15),  
edad INT,  
genero VARCHAR(10),  
grado VARCHAR(100),  
clasificacion VARCHAR(100),  
categoria VARCHAR(100),  
arma\_servicio VARCHAR(100),  
zodi VARCHAR(100),  
miliciano BOOLEAN,  
discapacidad VARCHAR(100),  
tipo\_res VARCHAR(50),  
create\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
matricula VARCHAR(255),  
observaciones TEXT  
);

### 5. ****Backend - API REST****

Configura tu servidor Express:

Code

const express = require('express');  
const app = express();  
const bodyParser = require('body-parser');  
  
app.use(bodyParser.json());  
  
// Rutas  
app.post('/api/inscripcion', (req, res) => {  
// Lógica para inscribir a un militar  
});  
  
app.listen(3000, () => {  
console.log('Servidor en ejecución en http://localhost:3000');  
});

### 6. ****Frontend - Formulario de Inscripción****

Crea un formulario en React:

Code

import React, { useState } from 'react';  
  
const InscripcionForm = () => {  
const [formData, setFormData] = useState({  
email: '',  
nombres\_completos: '',  
cedula: '',  
// Agrega los demás campos  
});  
  
const handleChange = (e) => {  
setFormData({ ...formData, [e.target.name]: e.target.value });  
};  
  
const handleSubmit = (e) => {  
e.preventDefault();  
// Lógica para enviar datos al backend  
};  
  
return (  
<form onSubmit={handleSubmit}>  
<input type="text" name="email" onChange={handleChange} placeholder="Email" required />  
<input type="text" name="nombres\_completos" onChange={handleChange} placeholder="Nombres Completos" required />  
<input type="text" name="cedula" onChange={handleChange} placeholder="Cédula" required />  
{/\* Agrega los demás campos \*/}  
<button type="submit">Inscribirse</button>  
</form>  
);  
};  
  
export default InscripcionForm;

### 7. ****Motor de Asignación Automática****

Implementa la lógica de asignación en el backend. Esta lógica se ejecutará después de que el militar se inscriba. Aquí es donde verificarás la disponibilidad en las UCRES y asignarás la matrícula.

### 8. ****Integración Biométrica****

Para la autenticación biométrica, investiga APIs como AWS Rekognition para reconocimiento facial y SDKs para lectores de huellas dactilares.

### 9. ****Pruebas y Seguridad****

Realiza pruebas en cada módulo. Asegúrate de aplicar medidas de seguridad, como validación de entrada y cifrado de datos sensibles.

### 10. ****Despliegue****

Considera usar Docker para contenerizar tu aplicación y Kubernetes para la orquestación, asegurando así que tu sistema sea escalable.

### 11. ****Documentación****

Finalmente, documenta tu proyecto, tanto la API como la lógica del sistema. Esto te ayudará a ti y a otros a entender el funcionamiento.