**Características principales de la WebApp:**

1. **Inicio e Interfaz de Usuario:**
   * **Vista General**: Presentación de los diferentes tipos de crímenes (robos, violaciones, homicidios, feminicidios).
   * **Mapa Interactivo (Mapa de Calor)**: Un mapa de calor que muestra las zonas de alto y bajo riesgo, basado en las denuncias. El mapa puede usar colores: verde para zonas tranquilas, amarillo para zonas con riesgo moderado, y rojo para zonas peligrosas.
   * **Gráficos Interactivos**: Gráficos que muestren estadísticas sobre el crimen, como la cantidad de casos denunciados por tipo de crimen, evolución a lo largo del tiempo, distribución geográfica, etc.
2. **Formulario de Denuncia:**  
   El formulario debe ser sencillo, pero lo suficientemente detallado para recolectar datos clave para el análisis y la seguridad. Aquí te dejo un ejemplo de los campos:
   * **Datos del denunciante (opcionalmente anónimos)**:
     + Nombre completo (opcional)
     + Dirección (opcional, para ubicar el crimen)
     + Teléfono de contacto (opcional)
     + Correo electrónico (opcional)
   * **Detalles del incidente**:
     + **Tipo de crimen**: Robos, violaciones, homicidios, feminicidios (desplegable o botones de opción)
     + **Fecha y hora del incidente**
     + **Descripción del crimen**: Breve texto de lo sucedido (campo de texto).
     + **Ubicación del crimen**: Puedes incluir un campo de texto con la dirección o un sistema de geolocalización con un mapa interactivo (tipo Google Maps) para marcar la ubicación exacta.
     + **Descripción de los involucrados**: Información sobre el perpetrador (si está disponible) o cualquier detalle relevante.
     + **¿Hubo víctimas?**: Sí/No. En caso afirmativo, agregar más detalles sobre las víctimas (nombre, edad, tipo de daño, etc.).
3. **Visualización de Datos y Estadísticas:**
   * **Gráficos**: Mostrar estadísticas de los crímenes registrados, desglosados por tipo de crimen, región, y periodo de tiempo. Podría ser con gráficos de barras, líneas, y diagramas de dispersión.
   * **Mapa de calor**: Utilizar herramientas como **Leaflet.js** para visualizar un mapa de calor donde las zonas peligrosas (rojas) se resalten más que las zonas tranquilas (verdes).
   * **Filtros**: Permitir a los usuarios filtrar por tipo de crimen, región, o tiempo (día, semana, mes, año).
4. **Sistema de Autenticación (opcional):**
   * **Cuenta de Usuario**: Opcionalmente, los usuarios pueden crear cuentas para tener acceso a estadísticas personalizadas o a un historial de los crímenes denunciados.
   * **Login**: Login para la policía o autoridades con acceso a datos completos y capacidades administrativas para gestionar reportes.

**Estructura de la Base de Datos (PostgreSQL):**

La base de datos debe ser lo suficientemente flexible como para manejar todos los datos relacionados con los crímenes denunciados. Podrías estructurarla de la siguiente manera:

1. **Tablas principales**:
   * **Denuncias**:
     + id (clave primaria)
     + tipo\_crimen (robos, violaciones, homicidios, feminicidios)
     + descripcion
     + fecha
     + ubicacion (latitud, longitud)
     + victima\_id (relación con una tabla de víctimas)
     + denunciante\_id (relación con la tabla de usuarios)
   * **Víctimas**:
     + id (clave primaria)
     + nombre
     + edad
     + tipo\_victima (agredida, asesinada, etc.)
   * **Usuarios** (si se usa autenticación):
     + id (clave primaria)
     + nombre
     + correo electrónico
     + telefono
     + direccion
   * **Mapas de calor**:
     + zona (relacionado con la ubicación geográfica)
     + nivel\_riesgo (valor de color, 1-5, para determinar el nivel de peligro)
2. **Relaciones**:
   * Una denuncia puede estar asociada a una o más víctimas.
   * Un denunciante (usuario) puede hacer varias denuncias.
   * Cada crimen tiene una ubicación que puede usarse para mostrar en el mapa de calor.

**Tecnologías para el Desarrollo:**

1. **Backend (API)**:
   * **Django (Python)**: Ambas son buenas opciones. Si te sientes cómodo con Python, **Django** es una excelente opción, ya que ofrece muchas funcionalidades "out of the box", como autenticación, administración de la base de datos, y administración de formularios.
     + **Django** podría ser la opción ideal si prefieres un marco de trabajo más estructurado con muchas bibliotecas listas para usar.
   * **API**: Para una API RESTful, puedes usar **Django REST Framework (DRF)**
2. **Base de Datos**:
   * **PostgreSQL** es una excelente elección por su robustez y capacidad de manejar consultas complejas, como las geográficas (utilizando la extensión PostGIS para consultas geoespaciales).
3. **Frontend**:
   * **React.js**: Para la interfaz de usuario, **React** es ideal para construir aplicaciones interactivas con gráficos y mapas. Si decides usar mapas, puedes integrarlo con **Leaflet.js**.
   * **Visualización de Datos**: Utiliza bibliotecas como **D3.js**, **Chart.js** o **Plotly.js** para la visualización interactiva de gráficos.
4. **Mapa de Calor**:
   * **Leaflet.js** Para la visualización del mapa interactivo, donde las zonas de riesgo se marcan según el nivel de criminalidad.