

17-01-2018

Manual Técnico

Proyecto Comunidad Organizada

Philip Arias Ares
Manrique J. Durán Vásquez
Valeria Garro Abarca
TECNOLOGICO DE COSTA RICA

CONTENIDO	
Introducción	4
Alcance	4
Información para el uso de la documentación	4
Referencias	4
Glosario	5
Requerimientos Técnicos	5
Requerimientos Mínimos de Hardware	5
Requerimientos Mínimos de Software	5
Herramientas Utilizadas	5
Framework	5
Programación WEB	5
Bases de Datos	6
Servidores	6
Control de Versionamiento	6
Diccionario de datos	6
Diagrama relacional	6
Tablas y vistas	8
roles	8
genders	8
people	8
users	9
password_resets	9
cat_report	9
cat_weapon	10
cat_transportation	10
countries	10
provinces	10
cantons	11
districts	11
states	11
community_groups	11

sub_cat_report	12
reports	12
cat_evidence	13
evidence	13
comments	13
likes	14
report_alert	14
security_reports	14
communities	15
communities_by_groups	15
users_by_community_groups	15
victims	16
perpetrators	16
cat_request_state	16
community_requests	16
community_admins	17
admins_by_communities	17
community_group_requests	17
communities_by_group_request	18
Seeders	18
Roles	18
Genders	18
People	18
Users	18
Catreport	19
SubCatReport	19
Countries	19
Provinces	19
Cantons	19
Districts	19
CatWeapon	19
CatTransportation	19

CatEvidence	19
States	19
CatRequestState	20
Políticas de respaldo	20
Archivos	20
base de datos	20
Enrutamiento	20
Descripción de interfaces con otros sistemas	28
Google maps api	28
tawk to api	28
Instalación Y configuración	28
Requisitos pre-instalación	28
Proceso de instalación	28
Proceso de Configuración	28

1. INTRODUCCIÓN

El presente Manual Técnico tiene como propósito contar con una guía clara y específica que garantice la óptima operación y desarrollo de las actividades que se pueden realizar en el sistema de Comunidades Organizadas.

Este manual comprende en forma ordenada la explicación de cómo se compone el sistema y las herramientas utilizadas para su desarrollo.

Este Manual está sujeto a actualización en la medida que se presenten variaciones en el sistema.

1.1. ALCANCE

En el presente documento se plantea la arquitectura, diseño y herramientas para la elaboración del del proyecto Comunidad Organizada.

Se desglosan las pruebas a realizar, las métricas y criterios de aceptación o rechazo a utilizar, así como, el análisis de resultados.

2. INFORMACIÓN PARA EL USO DE LA DOCUMENTACIÓN

2.1. REFERENCIAS

Titulo del Documento	Referencia
Standard IEEE 830 - 1998	IEEE
Estándares de programación en Laravel	[1] Estándares de programación en Laravel. (2018). https://styde.net/estandares-de-programacion-en-laravel/
Documento de especificación de Requerimientos	Documento Interno.
Documento de Arquitectura de Software	Documento Interno.

2.2. GLOSARIO

ISO: Organización Internacional para la Estandarización.

IEC: International Electrotechnical Commission.

IEEE: Institute of Electrical and Electronics Engineers.

OTS: Organizational Test Strategy for Traditional Ltd.

3. REQUERIMIENTOS TÉCNICOS

3.1. REQUERIMIENTOS MÍNIMOS DE HARDWARE

- **Procesador:** Unicore
- **RAM:** 4GB
- **Disco duro:** 500GB

3.2. REQUERIMIENTOS MÍNIMOS DE SOFTWARE

- **Sistema Operativo:** Windows/Linux
- **Navegador:** Firefox/Google Chrome/Microsoft Edge

4. HERRAMIENTAS UTILIZADAS

4.1. FRAMEWORK

Laravel: Laravel es un Framework Open Source para PHP. El objetivo de Laravel es el de ser un framework que permita el uso de una sintaxis refinada y expresiva para crear código de forma sencilla, evitando el “código espagueti” y permitiendo multitud de funcionalidades. Aprovecha todo lo bueno de otros frameworks y utiliza las características de las últimas versiones de PHP. Laravel se basa en el patrón de arquitectura de software llamado MVC (Modelo Vista Controlador).

4.2. PROGRAMACIÓN WEB

PHP: Es un Lenguaje de Programación para trabajar páginas WEB ofreciendo la ventaja de mezclarse con HTML y generar contenido dinámico. Las ejecuciones son realizadas en el Servidor y el cliente es el encargado de recibir los resultados de la ejecución. Permite conexión con varios tipos de Bases de Datos como: MySQL, Oracle, SQL Server, etc. permitiendo aplicaciones robustas sobre la WEB.

Javascript: Lenguaje de programación interpretado. Se utiliza principalmente del lado del cliente implementado como parte de un navegador web, permitiendo mejoras en la interfaz de usuario y páginas web. Se define como lenguaje Orientado a Objetos, imperativo y basado en prototipos. Con ayuda de AJAX es utilizado para enviar y recibir información del servidor.

AJAX: Es un acrónimo de *Asynchronous JavaScript And XML*. Es una técnica de desarrollo web para desarrollar aplicaciones interactivas. Esta herramienta ayuda a que mientras se navega en la aplicación se mantenga una comunicación asincrónica con el servidor.

HTML: Es un acrónimo de **HyperText Markup Language**. Es un lenguaje utilizado para elaboración de páginas web.

CSS: Es un acrónimo de **Cascade Style Sheets**. Es un lenguaje de diseño gráfico para definir el formato y estructura de un documento escrito en lenguaje de marcado como HTML.

4.3. BASES DE DATOS

MySQL: Es un sistema de Bases de Datos relacional desarrollado por MySQL AB para después de un tipo ser comprado por Oracle. Ofrece ventajas tales como fácil adaptación a diferentes entornos de desarrollo, interacción con lenguajes de programación como PHP, Javascript. También se puede utilizar en distintos Sistemas Operativos.

4.4. SERVIDORES

Apache

4.5. CONTROL DE VERSIONAMIENTO

Github

5. DICCIONARIO DE DATOS

5.1. DIAGRAMA RELACIONAL

5.2. TABLAS Y VISTAS

Todas las tablas dentro de la base de datos son generados por medio de migrations. Este esquema brinda beneficios de agregar o editar el modelo de la base y sus relaciones de manera más sencilla y potencialmente automatizada. Todas los migrations pueden accederse desde el directorio: '/database/migrations/'.

La estructura que llevan toma importancia en dos niveles. El nombre está definido por una fecha y luego el nombre: [fecha]_create_[nombre de la tabla]_table.php. El contenido del archivo consta de un método up y otro down. Dentro del up se define la estructura de la tabla (nombres y tipos de columnas) y las llaves foránea. El down contiene el nombre de la misma tabla. A la hora de refrescar la base de datos se llama el método down para borrar la tabla y luego el up para volverla a crear.

Es importante que las tablas que vayan a tener llaves foráneas tengan una fecha luego de la de donde pertenece esa llave. La base de datos es generada secuencialmente iniciando con las fechas más antiguas.

A continuación una descripción de cada una de las tablas.

5.2.1. ROLES

- **Nombre:** roles
- **Descripción:** Describe el rol de un usuario; Administrador, Administrador de Comunidades, Usuario Regular.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.2. GENDERS

- **Nombre:** genders
- **Descripción:** Describe el género de un usuario; Masculino, Femenino, Otro.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.3. PEOPLE

- **Nombre:** people
- **Descripción:** Describe los atributos de la persona relacionada a la cuenta de un usuario.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (varchar, 255) last_name
 - ❑ (varchar, 255) second_last_name
 - ❑ (varchar, 9) official_id
 - ❑ (u_int) gender_id
 - ❑ (bool) foreigner

- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) gender_id -> (int) id : [genders](#)

5.2.4. USERS

- **Nombre:** users
- **Descripción:** Describe la cuenta de un usuario dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) role_id
 - ❑ (u_int) person_id
 - ❑ (varchar, 128) email
 - ❑ (timestamp) email_verified_at
 - ❑ (varchar, 255) password
 - ❑ (text) avatar_path
 - ❑ (timestamp) deleted_at
 - ❑ (varchar, 100) remember_token
 - ❑ (timestamp) created_at
 - ❑ (timestamp) updated_at
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) person_id -> (int) id : [people](#)
 - ❑ (u_int) role_id -> (int) id : [roles](#)

5.2.5. PASSWORD_RESETS

- **Nombre:** password_resets
- **Descripción:** Almacena los registros de las solicitudes de cambios de contraseñas.
- **Atributos:**
 - ❑ (varchar, 128) email
 - ❑ (varchar, 255) token
 - ❑ (timestamp) created_at
- **Llaves primarias:**
 - ❑ (varchar) email
- **Llaves foráneas:** N/A

5.2.6. CAT_REPORT

- **Nombre:** cat_report
- **Descripción:** Catálogo de los tipos de reportes dentro del sistema; Noticia, seguridad, servicio.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.7. CAT_WEAPON

- **Nombre:** cat_weapon
- **Descripción:** Catálogo de los tipos de armas dentro de un reporte de seguridad.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.8. CAT_TRANSPORTATION

- **Nombre:** cat_transportation
- **Descripción:** Catálogo de medios de transporte relacionados a un reporte de seguridad.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.9. COUNTRIES

- **Nombre:** countries
- **Descripción:** Catálogo de países dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.10. PROVINCES

- **Nombre:** provinces
- **Descripción:** Catálogo de provincias dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) country_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) country_id-> (int) id : [countries](#)

5.2.11. CANTONS

- **Nombre:** cantons
- **Descripción:** Catálogo de cantones dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) province_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) province_id-> (int) id : [provinces](#)

5.2.12. DISTRICTS

- **Nombre:** districts
- **Descripción:** Catálogo de distritos dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) canton_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) canton_id-> (int) id : [cantons](#)

5.2.13. STATES

- **Nombre:** states
- **Descripción:** Catálogo de estados de un reporte.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.14. COMMUNITY_GROUPS

- **Nombre:** community_groups
- **Descripción:** Describe los nombres de los grupo de comunidades
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.15. SUB_CAT_REPORT

- **Nombre:** sub_cat_report
- **Descripción:** Catálogo de los subtipos de cada tipo de reporte dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) cat_report_id
 - ❑ (varchar, 255) name
 - ❑ (text) multimedia_path
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) cat_report_id-> (int) id : [cat_report](#)

5.2.16. REPORTS

- **Nombre:** reports
- **Descripción:** Describe las características generales dentro de cada reporte en el sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (text) title
 - ❑ (text) description
 - ❑ (u_int) community_group_id
 - ❑ (u_int) sub_cat_report_id
 - ❑ (u_int) user_id
 - ❑ (u_int) state_id
 - ❑ (timestamp) created_at
 - ❑ (timestamp) updated_at
 - ❑ (date) date
 - ❑ (time) time
 - ❑ (decimal, 11, 8) longitud
 - ❑ (decimal, 10, 8) latitud
 - ❑ (bool) active
 - ❑ (bool) news
 - ❑ (timestamp) deleted_at
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) community_group_id-> (int) id : [community_groups](#)
 - ❑ (u_int) sub_cat_report_id-> (int) id : [sub_cat_report](#)
 - ❑ (u_int) user_id-> (int) id : [users](#)
 - ❑ (u_int) state_id-> (int) id : [states](#)

5.2.17. CAT_EVIDENCE

- **Nombre:** cat_evidence
- **Descripción:** Catálogo de los tipos de evidencia para cada evidencia dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.18. EVIDENCE

- **Nombre:** evidence
- **Descripción:** Describe las características de cada evidencia relacionada a un reporte dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) report_id
 - ❑ (u_int) cat_evidence_id
 - ❑ (text) multimedia_path
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) report_id -> (int) id : [reports](#)
 - ❑ (u_int) cat_evidence_id -> (int) id : [cat_evidence](#)

5.2.19. COMMENTS

- **Nombre:** comments
- **Descripción:** Contiene los comentarios que el usuario ha hecho en una publicación.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) report_id
 - ❑ (u_int) user_id
 - ❑ (text) comment
 - ❑ (timestamp) create_at
 - ❑ (timestamp) updated_at
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) report_id -> (int) id : [reports](#)
 - ❑ (u_int) user_id -> (int) id : [users](#)

5.2.20. LIKES

- **Nombre:** likes
- **Descripción:** Contiene los agradecimientos que los usuarios han dado a las diferentes publicaciones.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) report_id
 - ❑ (u_int) user_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) report_id -> (int) id : [reports](#)
 - ❑ (u_int) user_id -> (int) id : [users](#)

5.2.21. REPORT_ALERT

- **Nombre:** report_alert
- **Descripción:** Contiene las alertas hechas por usuarios de algún reporte.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) report_id
 - ❑ (u_int) user_id
 - ❑ (text) reason
 - ❑ (timestamp) deleted_at
 - ❑ (timestamp) created_at
 - ❑ (timestamp) updated_at
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) report_id -> (int) id : [reports](#)
 - ❑ (u_int) user_id -> (int) id : [users](#)

5.2.22. SECURITY_REPORTS

- **Nombre:** security_reports
- **Descripción:** Describe las características específicas dentro de un reporte de seguridad.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) report_id
 - ❑ (u_int) cat_weapon_id
 - ❑ (u_int) cat_transportation_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) report_id -> (int) id : [reports](#)
 - ❑ (u_int) cat_weapon_id -> (int) id : [cat_weapon](#)
 - ❑ (u_int) cat_transportation_id -> (int) id : [cat_transportation](#)

5.2.23. COMMUNITIES

- **Nombre:** communities
- **Descripción:** Describe las comunidades dentro del sistema.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) district_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) district_id -> (int) id : [districts](#)

5.2.24. COMMUNITIES_BY_GROUPS

- **Nombre:** communities_by_groups
- **Descripción:** Contiene a las asociaciones de comunidades con grupos.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) community_id
 - ❑ (u_int) community_group_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) community_id -> (int) id : [communities](#)
 - ❑ (u_int) community_group_id -> (int) id : [community_groups](#)

5.2.25. USERS_BY_COMMUNITY_GROUPS

- **Nombre:** users_by_community_groups
- **Descripción:** Contiene las asociaciones de usuarios con grupos de comunidades. Sirve para mantener un registro de que grupos de comunidades el usuario sigue.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) user_id
 - ❑ (u_int) community_group_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) user_id -> (int) id : [users](#)
 - ❑ (u_int) community_group_id -> (int) id : [community_groups](#)

5.2.26. VICTIMS

- **Nombre:** victims
- **Descripción:** Describe las características de una víctima dentro de un reporte de seguridad.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) security_report_id
 - ❑ (u_int) gender_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) security_report_id -> (int) id : [security_reports](#)
 - ❑ (u_int) gender_id -> (int) id : [genders](#)

5.2.27. PERPETRATORS

- **Nombre:** perpetrators
- **Descripción:** Describe a los perpetradores dentro de un reporte de seguridad.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) security_report_id
 - ❑ (u_int) gender_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) security_report_id -> (int) id : [security_reports](#)
 - ❑ (u_int) gender_id -> (int) id : [genders](#)

5.2.28. CAT_REQUEST_STATE

- **Nombre:** cat_request_state
- **Descripción:** Catálogo de estados para las solicitudes de comunidades y grupo de comunidades.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (bool) active
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:** N/A

5.2.29. COMMUNITY_REQUESTS

- **Nombre:** community_requests
- **Descripción:** Contiene las solicitudes de comunidades realizadas por usuarios.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) user_id
 - ❑ (u_int) district_id
 - ❑ (u_int) cat_request_state_id

- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) user_id -> (int) id : [users](#)
 - ❑ (u_int) district_id -> (int) id : [districts](#)
 - ❑ (u_int) cat_request_state_id -> (int) id : [cat_request_state](#)

5.2.30. COMMUNITY_ADMINS

- **Nombre:** community_admins
- **Descripción:** Contiene las solicitudes de comunidades realizadas por usuarios.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) user_id
 - ❑ (u_int) active_community_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) user_id -> (int) id : [users](#)
 - ❑ (u_int) active_community_id -> (int) id : [communities](#)

5.2.31. ADMINS_BY_COMMUNITIES

- **Nombre:** admins_by_communities
- **Descripción:** Contiene las solicitudes de comunidades realizadas por usuarios.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) community_admin_id
 - ❑ (u_int) community_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) community_admin_id -> (int) id : [community_admins](#)
 - ❑ (u_int) community_id -> (int) id : [communities](#)

5.2.32. COMMUNITY_GROUP_REQUESTS

- **Nombre:** community_group_requests
- **Descripción:** Contiene las solicitudes de grupos de comunidades realizadas por usuarios.
- **Atributos:**
 - ❑ (int) id
 - ❑ (varchar, 255) name
 - ❑ (u_int) user_id
 - ❑ (u_int) cat_request_state_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) user_id -> (int) id : [users](#)

❑ (u_int) cat_request_state_id -> (int) id : [cat_request_state](#)

5.2.33. COMMUNITIES_BY_GROUP_REQUEST

- **Nombre:** community_by_group_request
- **Descripción:** Asocia comunidades con un grupo una solicitud de un grupo de comunidades.
- **Atributos:**
 - ❑ (int) id
 - ❑ (u_int) communitiy_group_request_id
 - ❑ (u_int) community_id
- **Llaves primarias:**
 - ❑ (int) id
- **Llaves foráneas:**
 - ❑ (u_int) communitiy_group_request_id -> (int) id : [community_group_requests](#)
 - ❑ (u_int) community_id -> (int) id : [communities](#)

5.3. SEEDERS

El framework de laravel provee una manera sencilla de llenar la base de datos. Esto se hace por medio de clases seeders. Por defecto la clase DatabaseSeeder realiza las llamadas a los otros seeders definidos. Por convención las clases de seeders tienen el siguiente formato en su nombre; [nombre]TableSeeder. Todas los seeds pueden accederse desde el directorio: '/database/seeds/'.

A continuación una descripción de todos los seeders definidos.

5.3.1. ROLES

- **Nombre:** RolesTableSeeder
- **Descripción:** Inserta los roles principales de usuarios al sistema.
- **Dependencias:** N/A

5.3.2. GENDERS

- **Nombre:** GenderTableSeeder
- **Descripción:** Inserta los géneros principales de personas al sistema.
- **Dependencias:** N/A

5.3.3. PEOPLE

- **Nombre:** PeopleTableSeeder
- **Descripción:** Crear el registro de la persona para la cuenta 'Super usuario' o 'Adminsitrador global'
- **Dependencias:**
 - ❑ Registro de un género (En este caso masculino) en la tabla de géneros.

5.3.4. USERS

- **Nombre:** UsersTableSeeder
- **Descripción:** Crea el registro del administrador global.
- **Dependencias:**
 - ❑ Registro de la persona para la cuenta de administrador global en la tabla de personas.

- ❑ Registro del rol administrador en la tabla de roles.

5.3.5. CATREPORT

- **Nombre:** CatReportTableSeeder
- **Descripción:** Inserta los tipos de reporte principales.
- **Dependencias:** N/A

5.3.6. SUBCATREPORT

- **Nombre:** SubCatReportTableSeeder
- **Descripción:** Inserta los subtipos para cada tipo de reporte.
- **Dependencias:**
 - ❑ Registro de servicio en el catálogo de tipos de reporte.
 - ❑ Registro de seguridad en el catálogo de tipos de reporte.

5.3.7. COUNTRIES

- **Nombre:** CountriesTableSeeder
- **Descripción:** Registra a Costa Rica dentro de la tabla de países.
- **Dependencias:** N/A

5.3.8. PROVINCES

- **Nombre:** ProvincesTableSeeder
- **Descripción:** Registra las provincias de Costa Rica.
- **Dependencias:**
 - ❑ Registro de costa rica dentro de la tabla de países.

5.3.9. CANTONS

- **Nombre:** CantonsTableSeeder
- **Descripción:** Registra los cantones dentro de cada provincia en Costa Rica.
- **Dependencias:**
 - ❑ Modelo de Provincias.
 - ❑ Registro de las provincias de Costa Rica en la tabla de provincias.

5.3.10. DISTRICTS

- **Nombre:** DistrictTableSeeder
- **Descripción:** Registra los distritos dentro de cada cantón en Costa Rica.
- **Dependencias:**
 - ❑ Modelo de Cantones.
 - ❑ Registro de los cantones de Costa Rica en la tabla de cantones.

5.3.11. CATWEAPON

- **Nombre:** CatWeaponTableSeeder
- **Descripción:** Registra los tipos de armas dentro de la tabla de cat_weapon.
- **Dependencias:** N/A

5.3.12. CATTRANSPORTATION

- **Nombre:** CatTransportationTableSeeder
- **Descripción:** Registra los medios de transporte dentro de la tabla de cat_transportation.
- **Dependencias:** N/A

5.3.13. CATEVIDENCE

- **Nombre:** CatEvidenceTableSeeder
- **Descripción:** Registra los tipos de evidencia dentro de la tabla de cat_evidence.
- **Dependencias:** N/A

5.3.14. STATES

- **Nombre:** StatesTableSeeder
- **Descripción:** Registra los estados de reportes dentro de la tabla de states.
- **Dependencias:** N/A

5.3.15. CATREQUESTSTATE

- **Nombre:** CatRequestStateTableSeeder
- **Descripción:** Registra los estados de las solicitudes de comunidades y grupos de comunidades en la tabla de cat_request_state.
- **Dependencias:** N/A

6. POLÍTICAS DE RESPALDO

6.1. ARCHIVOS

6.2. BASE DE DATOS

7. ENRUTAMIENTO

El enrutamiento define el direccionamiento y sus acciones relacionadas. Principalmente las acciones están ligadas a un controlador que maneja las solicitudes y define el comportamiento de la aplicación. La lista completa de rutas se puede acceder directamente en el archivo '/routes/web.php' o utilizando el comando de consola: 'php artisan route:list'. Generalmente el sistema se apega a rutas RESTful, pero existen excepciones. Todas las rutas se describirán a continuación.

Método	URI	Acción
GET HEAD	/	App\Http\Controllers\HomeController@index
PATCH	active-community	App\Http\Controllers\ActiveCommunityController@update
POST	add-comment	App\Http\Controllers\CommentController@store
GET	administracion	App\Http\Controllers\AdministrationHomeController@index
POST	administracion/administradores	App\Http\Controllers\AdministrationUsersController@store
GET	administracion/administradores	App\Http\Controllers\AdministrationUsersController@index

GET	administracion/administradores/agregar	App\Http\Controllers\AdministrationUsersController@create
POST	administracion/comunidades/comunidad	App\Http\Controllers\AdministrationCommunityController@store
GET	administracion/comunidades/comunidad	App\Http\Controllers\AdministrationCommunityController@index
GET	administracion/comunidades/comunidad/agregar	App\Http\Controllers\AdministrationCommunityController@create
POST	administracion/comunidades/comunidad/filtrar	App\Http\Controllers\AdministrationCommunityController@show
PATCH	administracion/comunidades/comunidad/{community}	App\Http\Controllers\AdministrationCommunityController@update
GET	administracion/comunidades/comunidad/{community}	App\Http\Controllers\AdministrationCommunityController@edit
GET	administracion/comunidades/grupos	App\Http\Controllers\AdministrationCommunityGroupController@index
POST	administracion/comunidades/grupos	App\Http\Controllers\AdministrationCommunityGroupController@store
GET	administracion/comunidades/grupos/agregar	App\Http\Controllers\AdministrationCommunityGroupController@create
POST	administracion/comunidades/grupos/filtrar	App\Http\Controllers\AdministrationCommunityGroupController@show
PATCH	administracion/comunidades/grupos/{community_group}	App\Http\Controllers\AdministrationCommunityGroupController@update
GET	administracion/comunidades/grupos/{community_group}	App\Http\Controllers\AdministrationCommunityGroupController@edit
GET	administracion/estados	App\Http\Controllers\AdministrationStateController@index
POST	administracion/estados	App\Http\Controllers\AdministrationStateController@store
PATCH	administracion/estados/activo/{state}	App\Http\Controllers\AdministrationStateController@toggle

GET	administracion/estados/agregar	App\Http\Controllers\AdministrationStateController@create
PATCH	administracion/estados/{state}	App\Http\Controllers\AdministrationStateController@update
GET	administracion/estados/{state}	App\Http\Controllers\AdministrationStateController@edit
POST	administracion/evidencias	App\Http\Controllers\AdministrationEvidenceController@store
GET	administracion/evidencias	App\Http\Controllers\AdministrationEvidenceController@index
PATCH	administracion/evidencias/activo/{evidence}	App\Http\Controllers\AdministrationEvidenceController@toggle
GET	administracion/evidencias/agregar	App\Http\Controllers\AdministrationEvidenceController@create
GET	administracion/evidencias/{evidence}	App\Http\Controllers\AdministrationEvidenceController@edit
PATCH	administracion/evidencias/{evidence}	App\Http\Controllers\AdministrationEvidenceController@update
POST	administracion/generos	App\Http\Controllers\AdministrationGenderController@store
GET	administracion/generos	App\Http\Controllers\AdministrationGenderController@index
GET	administracion/generos/agregar	App\Http\Controllers\AdministrationGenderController@create
GET	administracion/generos/{gender}	App\Http\Controllers\AdministrationGenderController@edit
PATCH	administracion/generos/{gender}	App\Http\Controllers\AdministrationGenderController@update
GET	administracion/home	App\Http\Controllers\AdministrationHomeController@index
POST	administracion/login	App\Http\Controllers\AdministrationSessionsController@store
GET	administracion/login	App\Http\Controllers\AdministrationSessionsController@create
GET	administracion/logout	App\Http\Controllers\AdministrationSessionsController@destroy
GET	administracion/publicaciones	App\Http\Controllers\AdministrationPublicationController@index

PATCH	administracion/publicaciones/activo/{report}	App\Http\Controllers\AdministrationPublicationController@toggle
GET	administracion/publicaciones/editar/{report}	App\Http\Controllers\AdministrationPublicationController@edit
PATCH	administracion/publicaciones/{report}	App\Http\Controllers\AdministrationPublicationController@update
DELETE	administracion/publicaciones/{report}	App\Http\Controllers\AdministrationPublicationController@destroy
GET	administracion/publicaciones/{report}	App\Http\Controllers\AdministrationPublicationController@show
GET	administracion/reportes	App\Http\Controllers\AdministrationReportController@index
PATCH	administracion/reportes/activo/{report}	App\Http\Controllers\AdministrationPublicationController@toggle
DELETE	administracion/reportes/alertas/{reportAlert}	App\Http\Controllers\AdministrationReportController@destroy
GET	administracion/reportes/ignorar/{report}	App\Http\Controllers\AdministrationPublicationController@ignore
GET	administracion/reportes/{report}	App\Http\Controllers\AdministrationReportController@show
POST	administracion/roles	App\Http\Controllers\AdministrationRoleController@store
GET	administracion/roles	App\Http\Controllers\AdministrationRoleController@index
GET	administracion/roles/agregar	App\Http\Controllers\AdministrationRoleController@create
POST	administracion/roles/filtrar	App\Http\Controllers\AdministrationRoleController@show
POST	administracion/roles/usuarios/{user}	App\Http\Controllers\AdministrationRoleController@updateUser
GET	administracion/roles/usuarios/{user}	App\Http\Controllers\AdministrationRoleController@editUser
GET	administracion/roles/{role}	App\Http\Controllers\AdministrationRoleController@edit

PATCH	administracion/roles/{role}	App\Http\Controllers\AdministrationRoleController@update
GET	administracion/seguridad	App\Http\Controllers\AdministrationSecurityController@index
POST	administracion/seguridad/armas	App\Http\Controllers\AdministrationWeaponController@store
PATCH	administracion/seguridad/armas/activo/{catWeapon}	App\Http\Controllers\AdministrationWeaponController@toggle
GET	administracion/seguridad/armas/agregar	App\Http\Controllers\AdministrationWeaponController@create
PATCH	administracion/seguridad/armas/{catWeapon}	App\Http\Controllers\AdministrationWeaponController@update
GET	administracion/seguridad/armas/{catWeapon}	App\Http\Controllers\AdministrationWeaponController@edit
POST	administracion/seguridad/categorias	App\Http\Controllers\AdministrationSecurityController@store
PATCH	administracion/seguridad/categorias/activo/{subCatReport}	App\Http\Controllers\AdministrationSecurityController@toggle
GET	administracion/seguridad/categorias/agregar	App\Http\Controllers\AdministrationSecurityController@create
PATCH	administracion/seguridad/categorias/{subCatReport}	App\Http\Controllers\AdministrationSecurityController@update
POST	administracion/seguridad/transportes	App\Http\Controllers\AdministrationTransportationController@store
PATCH	administracion/seguridad/transportes/activo/{catTransportation}	App\Http\Controllers\AdministrationTransportationController@toggle
GET	administracion/seguridad/transportes/agregar	App\Http\Controllers\AdministrationTransportationController@create
PATCH	administracion/seguridad/transportes/{catTransportation}	App\Http\Controllers\AdministrationTransportationController@update
GET	administracion/seguridad/transportes/{catTransportation}	App\Http\Controllers\AdministrationTransportationController@edit
GET	administracion/seguridad/{subCatReport}	App\Http\Controllers\AdministrationSecurityController@edit
POST	administracion/servicio	App\Http\Controllers\AdministrationServiceController@store

GET	administracion/servicio	App\Http\Controllers\AdministrationServiceController@index
PATCH	administracion/servicio/activo/{subCatReport}	App\Http\Controllers\AdministrationServiceController@toggle
GET	administracion/servicio/agregar	App\Http\Controllers\AdministrationServiceController@create
PATCH	administracion/servicio/{subCatReport}	App\Http\Controllers\AdministrationServiceController@update
GET	administracion/servicio/{subCatReport}	App\Http\Controllers\AdministrationServiceController@edit
GET	administracion/solicitudes	App\Http\Controllers\AdministrationRequestController@index
POST	administracion/solicitudes/comunidad/agregar/{communityRequest}	App\Http\Controllers\AdministrationCommunityRequestController@store
DELETE	administracion/solicitudes/comunidad/{communityRequest}	App\Http\Controllers\AdministrationCommunityRequestController@destroy
POST	administracion/solicitudes/grupo/agregar/{communityGroupRequest}	App\Http\Controllers\AdministrationGroupRequestController@store
DELETE	administracion/solicitudes/grupo/{communityGroupRequest}	App\Http\Controllers\AdministrationGroupRequestController@destroy
POST	busqueda	App\Http\Controllers\SearchController@show
GET	busqueda	App\Http\Controllers\SearchController@index
ALL	cantones	App\Http\Controllers\CantonController@show
ALL	comunidad	App\Http\Controllers\CommunitiesController@show
GET	comunidades	App\Http\Controllers\CommunitiesController@index
GET	comunidades/solicitar-comunidad	App\Http\Controllers\CommunitiesController@create
POST	comunidades/solicitar-comunidad	App\Http\Controllers\CommunitiesController@store
GET	comunidades/solicitar-grupo	App\Http\Controllers\GroupController@create

POST	comunidades/solicitar-grupo	App\Http\Controllers\GroupController@store
POST	comunidades/solicitar-grupo/filtrar	App\Http\Controllers\GroupController@fetchCommunitiesByDistrict
POST	dejar-grupo	App\Http\Controllers\GroupController@unfollow
ALL	distritos	App\Http\Controllers\DistrictController@show
GET	favoritas	App\Http\Controllers\FavoriteController@index
POST	favoritas/comunidades	App\Http\Controllers\FavoriteController@show
ALL	generos	App\Http\Controllers\GenderController@show
POST	grupo-comunidades	App\Http\Controllers\GroupController@communities
ALL	grupos	App\Http\Controllers\GroupController@show
GET	home	App\Http\Controllers\HomeController@index
POST	home	App\Http\Controllers\HomeController@show
GET	home/recientes	App\Http\Controllers\HomeController@showRecent
GET	index	App\Http\Controllers\HomeController@index
GET	informacion	Closure
ALL	like	App\Http\Controllers\LikeController@store
POST	login	App\Http\Controllers\Auth\LoginController@login
GET	login	App\Http\Controllers\Auth\LoginController@showLoginForm
GET	logout	App\Http\Controllers\SessionsController@destroy
POST	logout	App\Http\Controllers\Auth\LoginController@logout

GET	mis-grupos	App\Http\Controllers\GroupController@userCommunitiesIndex
POST	noticia	App\Http\Controllers\NewsController@store
GET	noticia	Closure
GET	noticia/agregar	App\Http\Controllers\NewsController@create
PATCH	noticia/{report}	App\Http\Controllers\NewsController@update
GET	obtener-grupos	App\Http\Controllers\GroupController@userCommunitiesShow
POST	password/email	App\Http\Controllers\Auth\ForgotPasswordController@sendResetLinkEmail
GET	password/reset	App\Http\Controllers\Auth\ForgotPasswordController@showLinkRequestForm
POST	password/reset	App\Http\Controllers\Auth\ResetPasswordController@reset
GET	password/reset/{token}	App\Http\Controllers\Auth\ResetPasswordController@showResetForm
ALL	provincias	App\Http\Controllers\ProvinceController@index
POST	register	App\Http\Controllers\Auth\RegisterController@register
GET	register	App\Http\Controllers\Auth\RegisterController@showRegistrationForm
POST	remove-comment	App\Http\Controllers\CommentController@destroy
POST	reportar/{report}	App\Http\Controllers\ReportAlertController@store
GET	reportar/{report}	App\Http\Controllers\ReportAlertController@create
GET	reporte/editar/{report}	App\Http\Controllers\ReportController@edit
GET	reporte/{report}	App\Http\Controllers\ReportController@show

POST	seguir-grupo	App\Http\Controllers\GroupController@follow
POST	seguridad	App\Http\Controllers\SecurityReportController@store
GET	seguridad	Closure
GET	seguridad/agregar	App\Http\Controllers\SecurityReportController@create
PATCH	seguridad/{report}	App\Http\Controllers\SecurityReportController@update
POST	servicio	App\Http\Controllers\ServiceReportController@store
GET	servicio	Closure
GET	servicio/agregar	App\Http\Controllers\ServiceReportController@create
PATCH	servicio/{report}	App\Http\Controllers\ServiceReportController@update
GET	statistics	App\Http\Controllers\StatisticsController@index
GET	statistics/cr_map	App\Http\Controllers\StatisticsController@reports_per_province
POST	statistics/genero	App\Http\Controllers\StatisticsController@statisticsBySexIncident
GET	statistics/genero	App\Http\Controllers\StatisticsController@statisticsBySex
POST	statistics/securityBar	App\Http\Controllers\StatisticsController@securityByDate
GET	statistics/securityBar	App\Http\Controllers\StatisticsController@securityBar
POST	statistics/serviceBar	App\Http\Controllers\StatisticsController@serviceByDate
GET	statistics/serviceBar	App\Http\Controllers\StatisticsController@serviceBar
POST	statistics/tiempo	App\Http\Controllers\StatisticsController@chartByTimeFilters
GET	statistics/tiempo	App\Http\Controllers\StatisticsController@chartByTime

GET	terminos-y-condiciones	Closure
ALL	unlike	App\Http\Controllers\LikeController@destroy
POST	update-comment	App\Http\Controllers\CommentController@update
GET	user	App\Http\Controllers\UserController@index
GET	user/{user}	App\Http\Controllers\UserController@edit
PATCH	user/{user}	App\Http\Controllers\UserController@update

8. DESCRIPCIÓN DE INTERFACES CON OTROS SISTEMAS

8.1. GOOGLE MAPS API

- **Nombre:**
- **Descripción:**
- **Formas de Comunicación:**
- **Cómo utilizar:**

8.2. TAWK TO API

9. INSTALACIÓN Y CONFIGURACIÓN

9.1. REQUISITOS PRE-INSTALACIÓN

```
// clonar el repo del git

// Si es local -> php, mysql, composer, laravel

// composer install

// composer update

// composer dump-autoload
```

9.2. PROCESO DE INSTALACIÓN

```
// CREATE DATABASE ComunidadesOrganizadas;
```

```
// php artisan migrate:fresh --seed
```

```
// En caso de que esté refrescando la base -> borrar folders en /public/evidence y  
/public/users/
```

```
// php artisan laraveles:install-lang
```

9.3. PROCESO DE CONFIGURACIÓN

```
// renombrar el .env.copy a .env
```

```
// php artisan key:generate
```

```
// Pegar llave en el APP_KEY
```

```
// Configurar el APP_URL
```

```
// Configurar nombre de la base y usuario y contraseña, puerto, host
```

```
// DB_DATABASE, DB_USER, DB_PASSWORD, DB_PORT, DB_HOST
```

```
// php artisan cache:clear
```