INFORME FINAL PROYECTO

CONTEO CIUDADANO

RAFAEL ORTIZ VERGARA

Código 7501810003

TUTOR

JOHN ARRIETA

UNIVERSIDAD DE CARTAGENA

PROGRAMA DE INGENIERÍA DE SOFTWARE

DESARROLLO DE SOFTWARE WEB

CARTAGENA D.T. y C.

JULIO 2020

TABLA DE CONTENIDO

INTRODUCCIÓN

El presente escrito busca presentar un informe acerca de la puesta en práctica de todo lo aprendido en las asignaturas Desarrollo web a través de un proyecto real en el que se pueda demostrar las habilidades adquiridas por medio de lo relacionado con el diseño e implementación de bases de datos para aplicaciones de software.

TÍTULO

Diseño, Implementación y conexión de Conteo Ciudadano App utilizando las tecnologías del PERN stack: PostgreSQL + Express + React + NodeJS.

JUSTIFICACIÓN

La razón u objetivos de realizar este proyecto es con el único fin de comprender la naturaleza en ámbitos de desarrollo web, implementar protocolos de acciones que permitan realizar conexiones con servidores web, realizar aplicativos full stack que interactúen el front con el backend, implementando estrategias que permitan la viabilidad y confiabilidad del aplicativo desarrollado durante el proceso de aprendizaje y futuros diseños laborales.

OBJETIVOS

- Comprender desde un punto de vista práctico los conceptos explicados en las unidades didácticas teóricas.
- Disponer de un modelo de referencia para emprender proyectos de Desarrollo Web
- Adquirir el criterio suficiente para identificar las actividades clave y tomar decisiones en un proyecto que implique el uso de Desarrollo Web.
- Diseñar una app que controle el censo de los ciudadanos de una población
- Mostrar los resultados obtenidos durante la elaboración del proyecto.
- Aprender la forma de usar algunas librería para realizar reportes PDF.

DESCRIPCIÓN DEL PROBLEMA

Para realizar el diseño de la Base de Datos del censo de población se dispone de la siguiente especificación de requerimientos:

- Se almacenarán datos de personas y datos de lugares.
- Cada persona tiene un nombre, apellidos, fecha de nacimiento, lugar de nacimiento, edad, estatura, sexo y nivel de estudios. Para los hombres también interesa conocer su situación militar. Para las personas mayores de 16 años también se debe considerar el DNI.
- Cada lugar corresponde a un municipio, teniendo como datos propios el nombre y el código (número secuencial del municipio según orden alfabético en cada provincia). Por ejemplo, Ciudad Real capital es el municipio “13034”, ya que el código de la provincia es el “13”.
- Todas las personas están censadas (población de derecho) en un municipio en una dirección (calle y número). Interesa conocer la fecha en que se registró dicha situación.
- Además, cada persona es residente (población de hecho) en un municipio en una dirección o en un país extranjero. En ambos casos queremos conocer también la fecha de inicio.
- Una persona puede ser residente en el mismo municipio.
- De cada municipio y de cada provincia interesa conocer su población de derecho (censada) y de hecho (residente realmente).

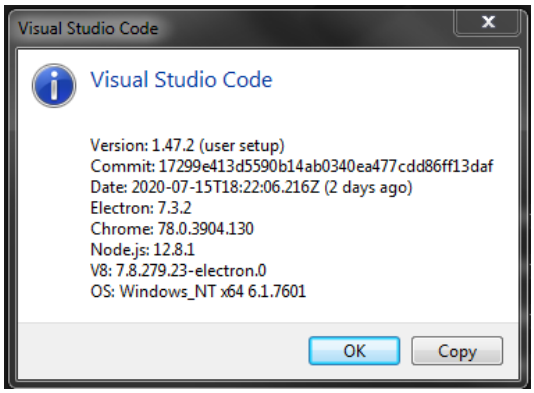
Se pide:

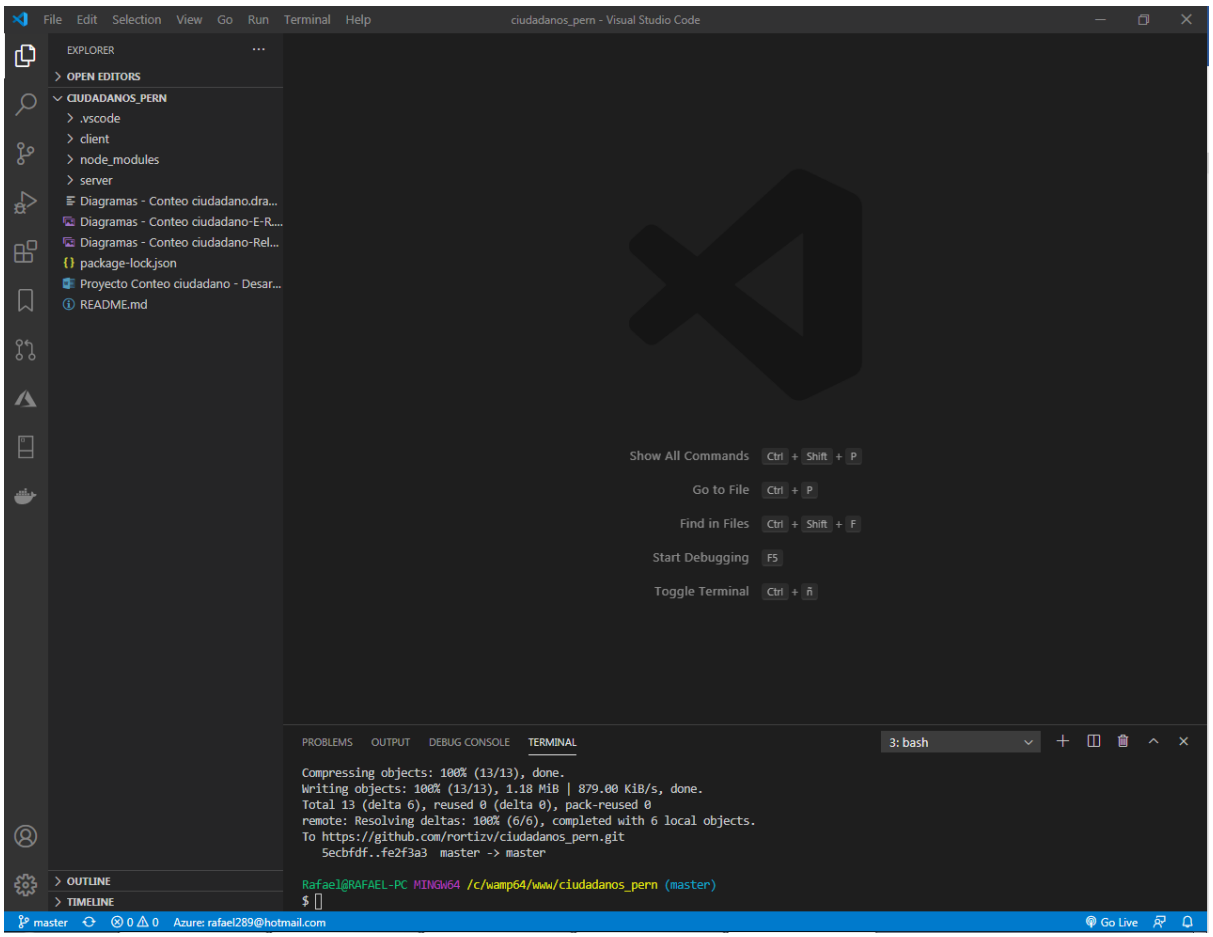
A. Diseñar el sistema de bases de datos para el caso anterior.
B. Proponer una forma lo más sencilla posible de transformar el anterior diagrama para que la dirección (calle y no) donde está censada una persona, nos permita conocer el distrito al que pertenece dentro del municipio. El distrito es una subdivisión del municipio compuesta por un conjunto de manzanas (una manzana no puede dividirse para pertenecer a dos distritos). Una manzana es un conjunto de casas físicamente unidas, separadas de las demás por calles. Cada lado de la manzana incluye una serie de números consecutivos (pares o impares) de una calle.

DESARROLLO

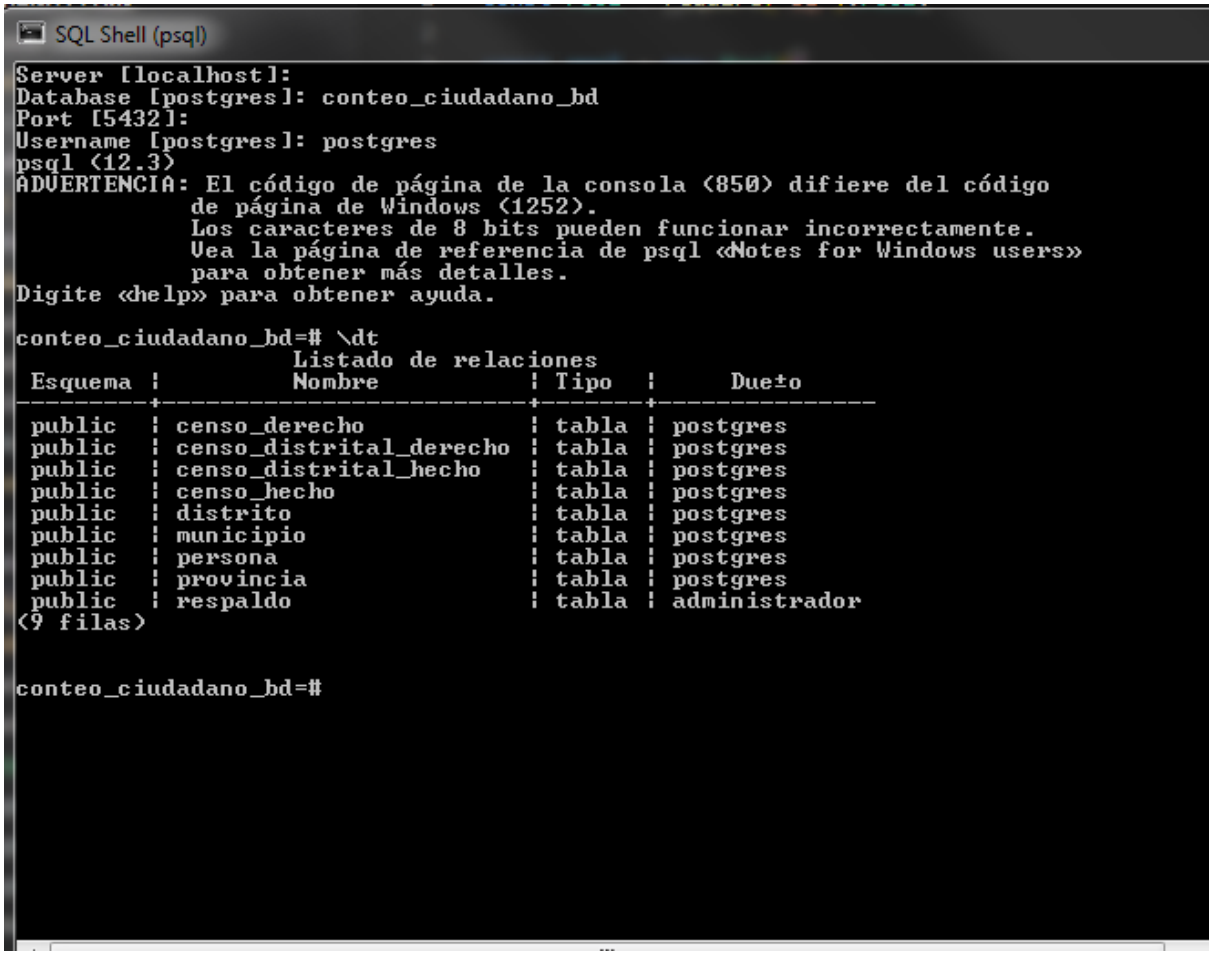
Descripción del entorno de trabajo

1. Editor de código: Visual Studio Code v1.47.2





2. Motor de BD: Postgre SQL 12



ENLACE DE UBICACIÓN DE REPOSITORIO CON EL CÓDIGO EN GITHUB

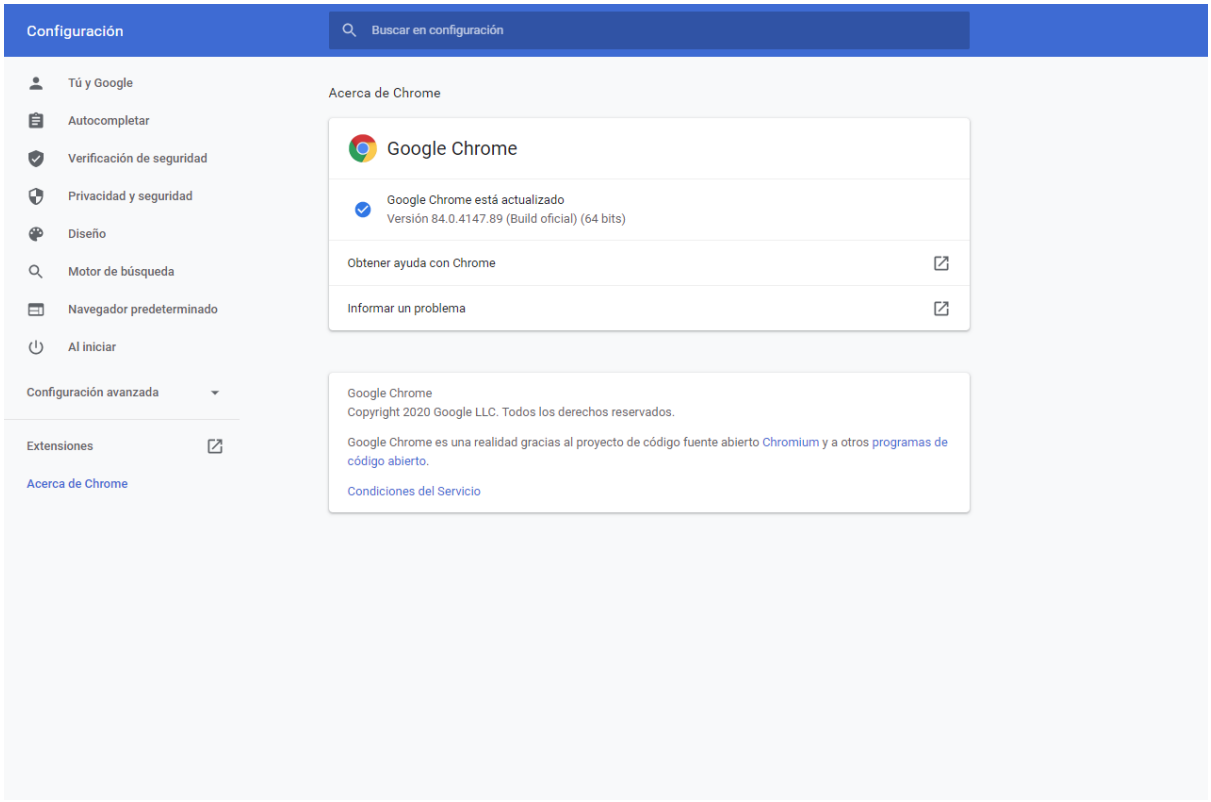
https://github.com/rortizv/ciudadanos_pern

ENLACE DE VIDEO DEMONSTRATIVO EN YOUTUBE

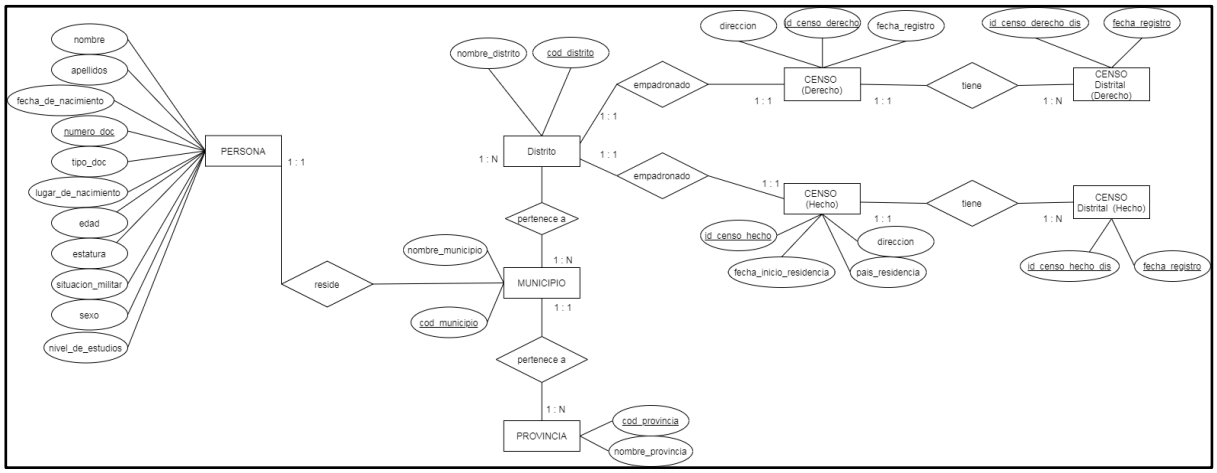
https://youtu.be/xej7WZ4JYoE

Navegadores sobre el cual hicieron las pruebas

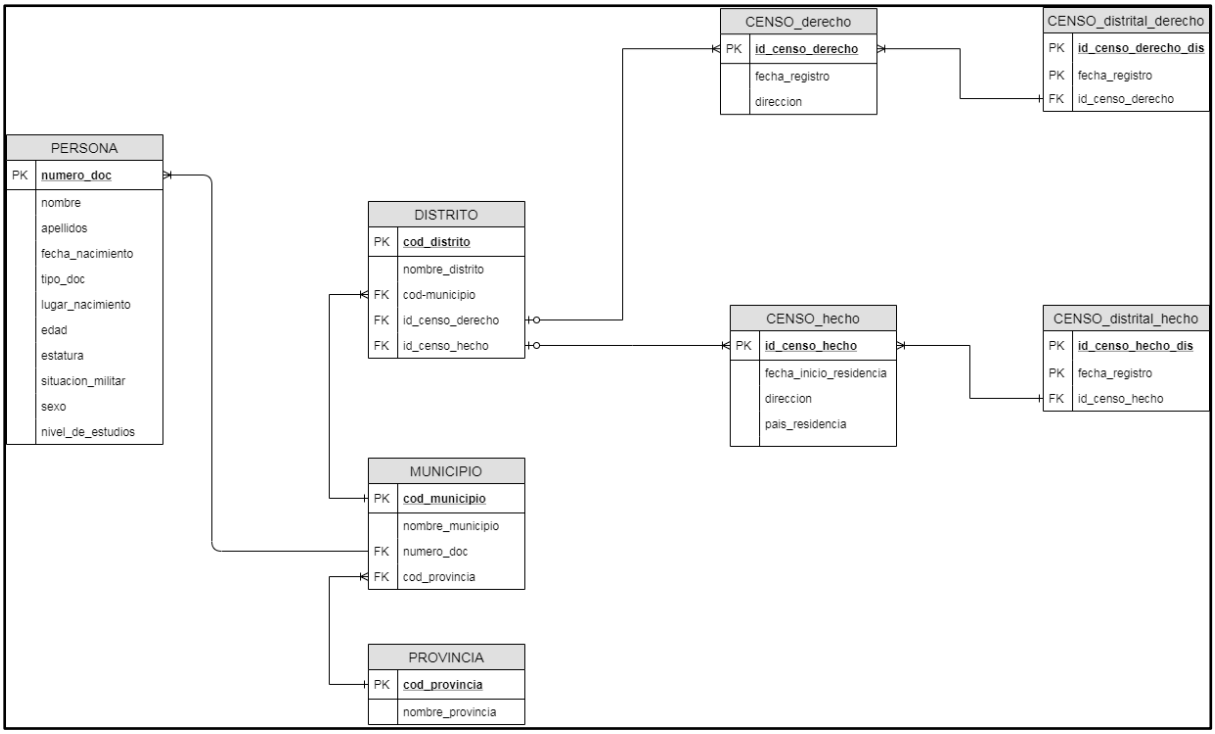
Google Chrome v84



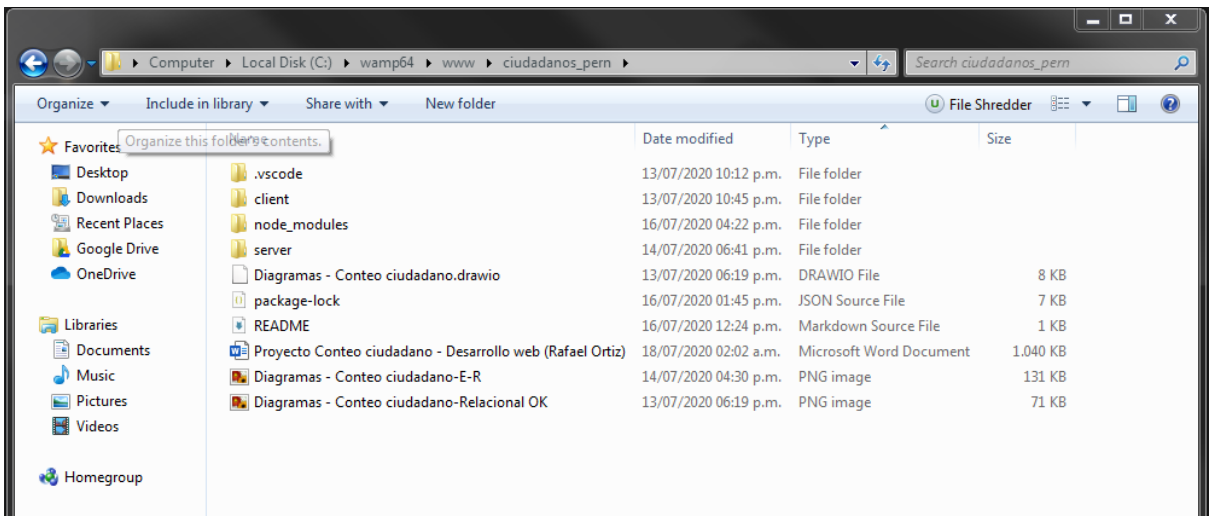
Modelo de BD



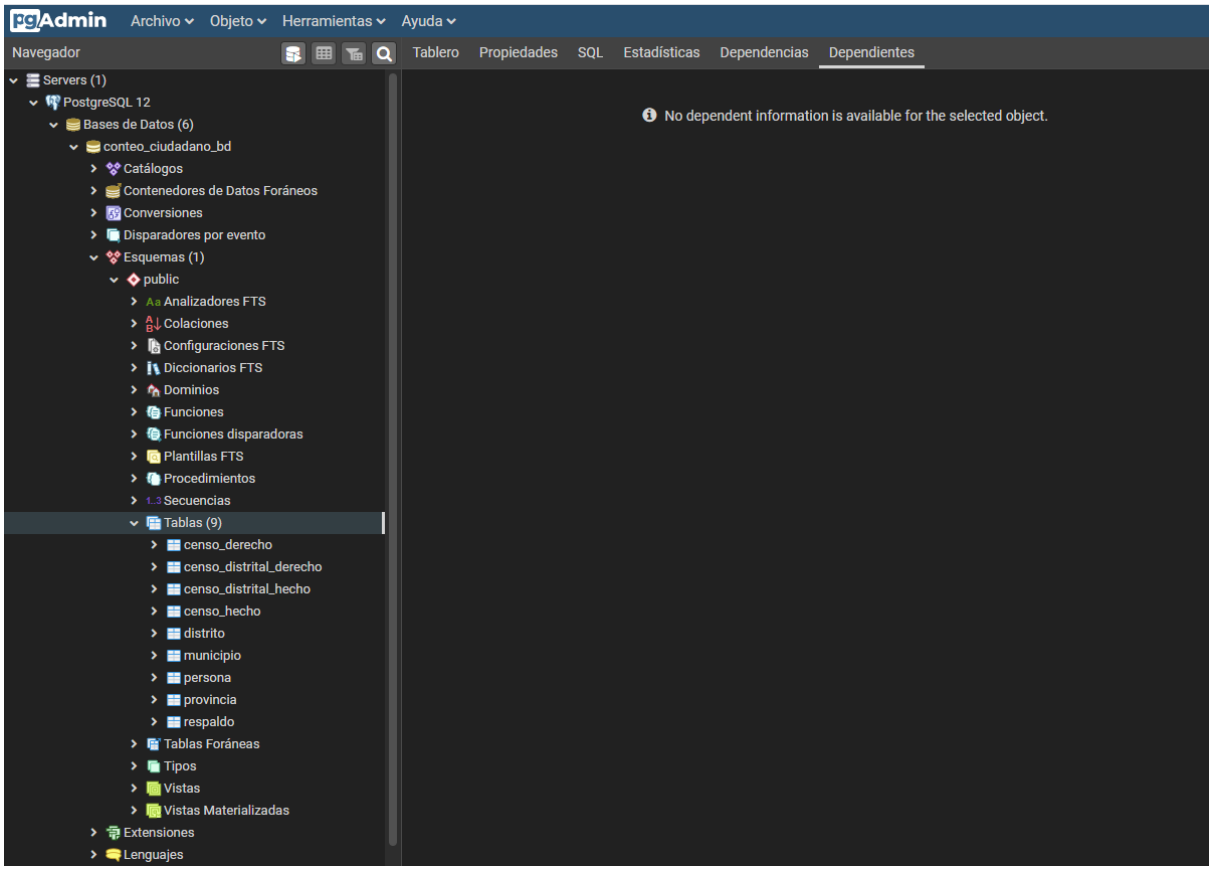
Modelo de clases Base o Modelo Conceptual



Información sobre la estructura de carpeta y archivos del proyecto



ADMINISTRACIÓN DE LA BASE DE DATOS CON PGADMIN4



Vista (carpeta): A continuación la estructura de la carpeta con el modelo de componentes manejado por React:

- Client
  - Public
    -
    - Favicon.ico
    - Index.html
    - Logo192.png
    - Logo512.png
    - Manifest.json
    - Robots.txt
    - Src
      - Components
        - Censo_derecho
          - EditCensoDerecho.js
          - InputCensoDerecho.js
          - ListCensoDerecho.js
        - Censo_distrital_derecho
          - EditCensoDistritalDerecho.js
          - InputCensoDistritalDerecho.js
          - ListCensoDistritalDerecho.js
        - Censo_distrital_hecho
          - EditCensoDistritalHecho.js
          - InputCensoDistritalHecho.js
          - ListCensoDistritlHecho.js
        - Censo_hecho
          - EditCensoHecho.js
          - InputCensoHecho.js
          - ListCensoHecho.js
        - Home
          - Home.js
        - Municipio
          - EditMunicipio.js

        - InputMunicipio.js
        - ListMunicipio.js
    - Navbar
        - Navbar.js
    - Persona
        - EditPersona.js
        - InputPersona.js
        - ListPersona.js
    - Provincia
        - EditProvincia.js
        - InputProvincia.js
        - ListProvincia.js
- App.css
- App.js
- Index.css
- Index.js

- Server
    - Db.js
    - Index.js
    - Package.json
    - Package-lock.json

SERVER

Dentro de esta carpeta, el archivo index.js comprende la API REST que controla el backend.

Instrucciones donde agrega, Busca, edita, elimina y lista

Estas instrucciones son llevadas a cabo en el archivo index.js, donde tenemos todas los comandos de los diferentes *crud's.*

```
const express = require("express");
const app = express();
const cors = require("cors");
const pool = require("./db");

//middleware
app.use(cors());
app.use(express.json()); //req.body
app.use(express.urlencoded({extended: false}));




//****** ROUTES *******//


//***PROVINCIA***

//create a provincia

app.post("/provincia", async (req, res) => {
  try {
    const { nombre_provincia } = req.body;
    const newProvincia = await pool.query(
      "INSERT INTO provincia (nombre_provincia) VALUES($1) RETURNING *",
      [nombre_provincia]
    );

    res.json(newProvincia.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});

//get all provincia
```

```
app.get("/provincia", async (req, res) => {
  try {
    const allProvincia = await pool.query("SELECT * FROM provincia");
    res.json(allProvincia.rows);
  } catch (err) {
    console.error(err.message);
  }
});

//get a provincia

app.get("/provincia/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const provincia = await pool.query("SELECT * FROM provincia WHERE cod_prov
incia = $1", [
      id
    ]);

    res.json(provincia.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});

//update a provincia

app.put("/provincia/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { nombre_provincia } = req.body;
    const updateProvincia = await pool.query(
      "UPDATE provincia SET nombre_provincia = $1 WHERE cod_provincia = $2",
      [nombre_provincia, id]
    );

    res.json("Provincia actualizada!");
  } catch (err) {
    console.error(err.message);
  }
});

//delete a provincia

app.delete("/provincia/:id", async (req, res) => {
  try {
    const { id } = req.params;
```

```
    const deleteProvincia = await pool.query("DELETE FROM provincia WHERE cod_
provincia = $1", [
      id
    ]);
    res.json("Provincia eliminada!");
  } catch (err) {
    console.log(err.message);
  }
});




//***PERSONA***


//create a persona

app.post("/persona", async (req, res) => {
  try {
    const { numero_doc, nombre, apellidos, fecha_nacimiento, tipo_doc, edad, e
statura, situacion_militar, sexo, nivel_de_estudios, fk_persona_cod_municipio
} = req.body;
    const newPersona = await pool.query(
      "INSERT INTO persona (numero_doc, nombre, apellidos, fecha_nacimiento, t
ipo_doc, edad, estatura, situacion_militar, sexo, nivel_de_estudios, fk_person
a_cod_municipio) VALUES($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11) RETURNIN
G *",
      [numero_doc, nombre, apellidos, fecha_nacimiento, tipo_doc, edad, estatu
ra, situacion_militar, sexo, nivel_de_estudios, fk_persona_cod_municipio]
    );

    res.json(newPersona.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all persona

app.get("/persona", async (req, res) => {
  try {
    const allPersona = await pool.query("SELECT * FROM persona");
    res.json(allPersona.rows);
  } catch (err) {
    console.error(err.message);
  }
```

```
});


//get a persona

app.get("/persona/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const persona = await pool.query("SELECT * FROM persona WHERE numero_doc =
 $1", [
      id
    ]);

    res.json(persona.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a persona

app.put("/persona/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { nombre, apellidos, fecha_nacimiento, tipo_doc, edad, estatura, sit
uacion_militar, sexo, nivel_de_estudios } = req.body;
    const updatePersona = await pool.query(
      'UPDATE persona SET nombre=$1, apellidos=$2, fecha_nacimiento=$3, tipo_d
oc=$4, edad=$5, estatura=$6, situacion_militar=$7, sexo=$8, nivel_de_estudios=
$9 WHERE numero_doc = $10',
      [nombre, apellidos, fecha_nacimiento, tipo_doc, edad, estatura, situacio
n_militar, sexo, nivel_de_estudios, id]
    );

    res.json("Persona actualizada!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a persona

app.delete("/persona/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deletePersona = await pool.query("DELETE FROM persona WHERE numero_d
oc = $1", [id]);
```

```
    res.json("Persona eliminada!");
  } catch (err) {
    console.log(err.message);
  }
});









//***MUNICIPIO***


//create a municipio

app.post("/municipio", async (req, res) => {
  try {
    const { cod_municipio, nombre_municipio, fk_cod_provincia } = req.body;
    const newMunicipio = await pool.query(
      "INSERT INTO municipio (cod_municipio, nombre_municipio, fk_cod_provinci
a) VALUES($1, $2, $3) RETURNING *",
      [cod_municipio, nombre_municipio, fk_cod_provincia]
    );

    res.json(newMunicipio.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all municipio

app.get("/municipio", async (req, res) => {
  try {
    const allMunicipio = await pool.query("SELECT * FROM municipio");
    res.json(allMunicipio.rows);
  } catch (err) {
    console.error(err.message);
  }
});


//get a municipio

app.get("/municipio/:id", async (req, res) => {
  try {
    const { id } = req.params;
```

```
    const municipio = await pool.query("SELECT * FROM municipio WHERE cod_muni
cipio = $1", [
      id
    ]);

    res.json(municipio.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a municipio

app.put("/municipio/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { nombre_municipio, fk_cod_provincia } = req.body; //
    const updateMunicipio = await pool.query(
      'UPDATE municipio SET nombre_municipio=$1, fk_cod_provincia=$2 WHERE cod
_municipio = $3',
      [nombre_municipio,fk_cod_provincia, id]
    );

    res.json("Municipio actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a municipio

app.delete("/municipio/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteMunicipio = await pool.query("DELETE FROM municipio WHERE cod_
municipio = $1", [id]);
    res.json("Municipio eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});




//***CENSO DERECHO***
```

```
//create a censo_derecho

app.post("/censo_derecho", async (req, res) => {
  try {
    const { id_censo_derecho, fecha_registro, direccion } = req.body;
    const newCensoDerecho = await pool.query(
      "INSERT INTO censo_derecho (id_censo_derecho, fecha_registro, direccion)
 VALUES($1, $2, $3) RETURNING *",
      [id_censo_derecho, fecha_registro, direccion]
    );

    res.json(newCensoDerecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all censo_derecho

app.get("/censo_derecho", async (req, res) => {
  try {
    const allCensoDerecho = await pool.query("SELECT * FROM censo_derecho");
    res.json(allCensoDerecho.rows);
  } catch (err) {
    console.error(err.message);
  }
});


//get a censo_derecho

app.get("/censo_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const censo_derecho = await pool.query("SELECT * FROM censo_derecho WHERE
id_censo_derecho = $1", [
      id
    ]);

    res.json(censo_derecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});
```

```
//update a censo_derecho

app.put("/censo_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { fecha_registro, direccion } = req.body; //
    const updateCensoDerecho = await pool.query(
      'UPDATE censo_derecho SET fecha_registro=$1, direccion=$2 WHERE id_censo
_derecho = $3',
      [fecha_registro,direccion, id]
    );

    res.json("Censo Derecho actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a censo_derecho

app.delete("/censo_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteCensoDerecho = await pool.query("DELETE FROM censo_derecho WHE
RE id_censo_derecho = $1", [id]);
    res.json("Censo Derecho eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});




//***CENSO HECHO***


//create a censo_hecho

app.post("/censo_hecho", async (req, res) => {
  try {
    const { id_censo_hecho, fecha_inicio_residencia, direccion, pais_residenci
a } = req.body;
    const newCensoHecho = await pool.query(
      "INSERT INTO censo_hecho (id_censo_hecho, fecha_inicio_residencia, direc
cion, pais_residencia) VALUES($1, $2, $3, $4) RETURNING *",
      [id_censo_hecho, fecha_inicio_residencia, direccion, pais_residencia]
```

```
    );

    res.json(newCensoHecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all censo_hecho

app.get("/censo_hecho", async (req, res) => {
  try {
    const allCensoHecho = await pool.query("SELECT * FROM censo_hecho");
    res.json(allCensoHecho.rows);
  } catch (err) {
    console.error(err.message);
  }
});


//get a censo_hecho

app.get("/censo_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const censo_hecho = await pool.query("SELECT * FROM censo_hecho WHERE id_c
enso_hecho = $1", [
      id
    ]);

    res.json(censo_hecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a censo_hecho

app.put("/censo_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { fecha_inicio_residencia, direccion, pais_residencia } = req.body;
//
    const updateCensoHecho = await pool.query(
      'UPDATE censo_hecho SET fecha_inicio_residencia=$1, direccion=$2, pais_r
esidencia=$3 WHERE id_censo_hecho = $4',
      [fecha_inicio_residencia, direccion, pais_residencia, id]
```

```
    );

    res.json("Censo Hecho actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a censo_hecho

app.delete("/censo_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteCensoHecho = await pool.query("DELETE FROM censo_hecho WHERE i
d_censo_hecho = $1", [id]);
    res.json("Censo Hecho eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});




//***DISTRITO***


//create a distrito

app.post("/distrito", async (req, res) => {
  try {
    const { cod_distrito, nombre_distrito, fk_cod_municipio, fk_cod_provincia
} = req.body;
    const newDistrito = await pool.query(
      "INSERT INTO distrito (cod_distrito, nombre_distrito, fk_cod_municipio,
fk_cod_provincia) VALUES($1, $2, $3, $4) RETURNING *",
      [cod_distrito, nombre_distrito, fk_cod_municipio, fk_cod_provincia]
    );

    res.json(newDistrito.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all distrito
```

```
app.get("/distrito", async (req, res) => {
  try {
    const allDistrito = await pool.query("SELECT * FROM distrito");
    res.json(allDistrito.rows);
  } catch (err) {
    console.error(err.message);
  }
});


//get a distrito

app.get("/distrito/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const distrito = await pool.query("SELECT * FROM distrito WHERE cod_distri
to = $1", [
      id
    ]);

    res.json(distrito.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a distrito

app.put("/distrito/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { nombre_distrito, fk_cod_municipio, fk_cod_provincia } = req.body;
//
    const updateDistrito = await pool.query(
      'UPDATE distrito SET nombre_distrito=$1, fk_cod_municipio=$2, fk_cod_pro
vincia=$3 WHERE cod_distrito = $4',
      [nombre_distrito,fk_cod_municipio,fk_cod_provincia, id]
    );

    res.json("Distrito actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a distrito
```

```
app.delete("/distrito/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteDistrito = await pool.query("DELETE FROM distrito WHERE cod_di
strito = $1", [id]);
    res.json("Distrito eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});




//***CENSO DISTRITAL DERECHO***


//create a censo_distrital_derecho

app.post("/censo_distrital_derecho", async (req, res) => {
  try {
    const { id_censo_derecho_dis, id_fecha_registro, fk_id_censo_derecho } = r
eq.body;
    const newCensoDistritalDerecho = await pool.query(
      "INSERT INTO censo_distrital_derecho (id_censo_derecho_dis, id_fecha_reg
istro, fk_id_censo_derecho) VALUES($1, $2, $3) RETURNING *",
      [id_censo_derecho_dis, id_fecha_registro, fk_id_censo_derecho]
    );

    res.json(newCensoDistritalDerecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all censo_distrital_derecho

app.get("/censo_distrital_derecho", async (req, res) => {
  try {
    const allCensoDistritalDerecho = await pool.query("SELECT * FROM censo_dis
trital_derecho");
    res.json(allCensoDistritalDerecho.rows);
  } catch (err) {
    console.error(err.message);
  }
});
```

```
//get a censo_distrital_derecho

app.get("/censo_distrital_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const censo_distrital_derecho = await pool.query("SELECT * FROM censo_dist
rital_derecho WHERE id_censo_derecho_dis = $1", [
      id
    ]);

    res.json(censo_distrital_derecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a censo_distrital_derecho

app.put("/censo_distrital_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { id_fecha_registro, fk_id_censo_derecho } = req.body; //
    const updateCensoDistritalDerechoo = await pool.query(
      'UPDATE censo_distrital_derecho SET id_fecha_registro=$1, fk_id_censo_de
recho=$2 WHERE id_censo_derecho_dis = $3',
      [fecha_registro, fk_id_censo_derecho, id]
    );

    res.json("Censo Distrital Derecho actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a censo_distrital_derecho

app.delete("/censo_distrital_derecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteCensoDistritalDerecho = await pool.query("DELETE FROM censo_di
strital_derecho WHERE id_censo_derecho_dis = $1", [id]);
    res.json("Censo Distrital Derecho eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});
```

```
//***CENSO DISTRITAL HECHO***


//create a censo_distrital_hecho

app.post("/censo_distrital_hecho", async (req, res) => {
  try {
    const { id_censo_hecho_dis, id_fecha_registro_hecho, fk_id_censo_hecho } =
 req.body;
    const newCensoDistritalHecho = await pool.query(
      "INSERT INTO censo_distrital_hecho (id_censo_hecho_dis, id_fecha_registr
o_hecho, fk_id_censo_hecho) VALUES($1, $2, $3) RETURNING *",
      [id_censo_hecho_dis, id_fecha_registro_hecho, fk_id_censo_hecho]
    );

    res.json(newCensoDistritalHecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//get all censo_distrital_hecho

app.get("/censo_distrital_hecho", async (req, res) => {
  try {
    const allCensoDistritalHecho = await pool.query("SELECT * FROM censo_distr
ital_hecho");
    res.json(allCensoDistritalHecho.rows);
  } catch (err) {
    console.error(err.message);
  }
});


//get a censo_distrital_hecho

app.get("/censo_distrital_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const censo_distrital_hecho = await pool.query("SELECT * FROM censo_distri
tal_hecho WHERE id_censo_hecho_dis = $1", [
      id
    ]);
```

```
    res.json(censo_distrital_hecho.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});


//update a censo_distrital_hecho

app.put("/censo_distrital_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { id_censo_hecho_dis, id_fecha_registro_hecho, fk_id_censo_hecho } =
 req.body; //
    const updateCensoDistritalHechoo = await pool.query(
      'UPDATE censo_distrital_hecho SET id_fecha_registro_hecho=$1, fk_id_cens
o_hecho=$2 WHERE id_censo_hecho_dis = $3',
      [id_fecha_registro_hecho, fk_id_censo_hecho, id]
    );

    res.json("Censo Distrital Hecho actualizado!");
  } catch (err) {
    console.error(err.message);
  }
});


//delete a censo_distrital_hecho

app.delete("/censo_distrital_hecho/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteCensoDistritalHecho = await pool.query("DELETE FROM censo_dist
rital_hecho WHERE id_censo_hecho_dis = $1", [id]);
    res.json("Censo Distrital Hecho eliminado!");
  } catch (err) {
    console.log(err.message);
  }
});



// Server running

app.listen(5000, () => {
  console.log("Servidor corriendo en puerto 5000");
});
```

Librerías o bibliotecas adicionales necesarias

Para el desarrollo de este aplicativo fue necesario usar un Framework de FrontEnd el cual fue Bootstrap en su versión 4.5, para realizar algunos diseños a las tablas, vistas y alertas entre otro.

Con este comando instalamos Bootstrap:

```
Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern (master)
$ npm install --save bootstrap
```

En el archivo App.js encontramos el siguiente fragmento de código que se encarga de importar Bootstrap:

```
import 'bootstrap/dist/css/bootstrap.min.css';
```

Script con fragmentos de código que contengan las instrucciones más importantes del desarrollo, y su respectiva explicación detallada

Entre fragmentos, tenemos el siguiente, en el que definimos la conexión a nuestra base de datos local PostgreSQL.

Nombre base de datos: **conteo_ciudadano_bd**

Usuario base de datos: **postgres**

Contraseña base de datos: **(en blanco)**

Enlace base de datos:

```
const Pool = require("pg").Pool;

const pool = new Pool({
    user: "postgres",
    password: "Montero.84",
    host: "localhost",
    port: 5432,
    database: "conteo_ciudadano_bd"
});

module.exports = pool;
```

Integración de las Vistas con el Controlador y el Modelo

Este trabajo es realizado en la renderización de los componentes de React.

Estos archivos son los que visualizan la información de cada una de las tablas que procesa la base de datos.

Components

/provincia/ListProvincia.js

```
import React, { Fragment, useEffect, useState } from "react";

import EditProvincia from "./EditProvincia";
import InputProvincia from "./InputProvincia";

const ListProvincia = () => {
  const [provincia, setProvincia] = useState([]);

  //delete provincia function

  const deleteProvincia = async id => {
    try {
      const deleteProvincia = await fetch(`http://localhost:5000/provincia/${i
d}`, {
        method: "DELETE"
      });

      setProvincia(provincia.filter(provincia => provincia.cod_provincia !== i
d));
    } catch (err) {
      console.error(err.message);
    }
  };

  const getProvincia = async () => {
    try {
      const response = await fetch("http://localhost:5000/provincia");
      const jsonData = await response.json();

      setProvincia(jsonData);
    } catch (err) {
      console.error(err.message);
    }
  };
```

```
  useEffect(() => {
    getProvincia();
  }, []);

  console.log(provincia);

  return (
    <Fragment>
      {" "}
      <InputProvincia />
      {/* <h1 className="text-center mt-5">PROVINCIA</h1> */}
      <table className="table mt-5 text-center">
        <thead>
          <tr>
            <th>Nombre Provincia</th>
            <th>Editar</th>
            <th>Eliminar</th>
          </tr>
        </thead>
        <tbody>
          {/*<tr>
            <td>John</td>
            <td>Doe</td>
            <td>john@example.com</td>
          </tr> */}
          {provincia.map(provincia => (
            <tr key={provincia.cod_provincia}>
              <td>{provincia.nombre_provincia}</td>
              <td>
                <EditProvincia provincia={provincia} />
              </td>
              <td>
                <button
                  className="btn btn-danger"
                  onClick={() => deleteProvincia(provincia.cod_provincia)}
                >
                  Eliminar
                </button>
              </td>
            </tr>
          ))}
        </tbody>
      </table>
    </Fragment>
  );
};
export default ListProvincia;
```

/provincia/InputProvincia.js

```
import React, { Fragment, useState } from "react";

const InputProvincia = () => {
  const [nombre_provincia, setNombreProvincia] = useState("");

  const onSubmitForm = async e => {
    e.preventDefault();
    try {
      const body = { nombre_provincia };
      const response = await fetch("http://localhost:5000/provincia", {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify(body)
      });

      window.location = "/provincia";
    } catch (err) {
      console.error(err.message);
    }
  };

  return (
    <Fragment>
      <h1 className="text-center mt-5">PROVINCIA</h1>
      <form className="d-flex mt-3" onSubmit={onSubmitForm}>
        <input
          type="text"
          className="form-control ml-5 mr-2"
          value={nombre_provincia}
          placeholder="Nombre PROVINCIA a agregar..."
          onChange={e => setNombreProvincia(e.target.value)}
        />
        <button className="btn btn-success ml-3 mr-5">Agregar</button>
      </form>
    </Fragment>
  );
};

export default InputProvincia;
```

/provincial/EditProvincia.js

```
import React, { Fragment, useState } from "react";

const EditProvincia = ({ provincia }) => {
  const [nombre_provincia, setNombreProvincia] = useState(provincia.nombre_pro
vincia);

  //edit nombre_provincia function

  const updateNombreProvincia = async e => {
    e.preventDefault();
    try {
      const body = { nombre_provincia };
      const response = await fetch(
        `http://localhost:5000/provincia/${provincia.cod_provincia}`,
        {
          method: "PUT",
          headers: { "Content-Type": "application/json" },
          body: JSON.stringify(body)
        }
      );

      window.location = "/provincia";
    } catch (err) {
      console.error(err.message);
    }
  };

  return (
    <Fragment>
      <button
        type="button"
        className="btn btn-warning"
        data-toggle="modal"
        data-target={`#id${provincia.cod_provincia}`}
      >
        Edit
      </button>

      {/*
        id = id10
      */}
      <div
        className="modal"
        id={`id${provincia.cod_provincia}`}
        onClick={() => setNombreProvincia(provincia.nombre_provincia)}
      >
```

```
        <div className="modal-dialog">
          <div className="modal-content">
            <div className="modal-header">
              <h4 className="modal-title">Editar Provincia</h4>
              <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setNombreProvincia(provincia.nombre_provincia)}
              >
                &times;
              </button>
            </div>

            <div className="modal-body">
              <input
                type="text"
                className="form-control"
                value={nombre_provincia}
                onChange={e => setNombreProvincia(e.target.value)}
              />
            </div>

            <div className="modal-footer">
              <button
                type="button"
                className="btn btn-warning"
                data-dismiss="modal"
                onClick={e => updateNombreProvincia(e)}
              >
                Edit
              </button>
              <button
                type="button"
                className="btn btn-danger"
                data-dismiss="modal"
                onClick={() => setNombreProvincia(provincia.nombre_provincia)}
              >
                Close
              </button>
            </div>
          </div>
        </div>
      </Fragment>
    );
};
export default EditProvincia;
```

/persona/ListPersona.js

```
import React, { Fragment, useEffect, useState } from "react";

import EditPersona from "./EditPersona";
import InputPersona from "./InputPersona";

const ListPersona = () => {
  const [persona, setPersona] = useState([]);

  //delete persona function

  const deletePersona = async id => {
    try {
      const deletePersona = await fetch(`http://localhost:5000/persona/${id}`,
 {
        method: "DELETE"
      });

      setPersona(persona.filter(persona => persona.numero_doc !== id));
    } catch (err) {
      console.error(err.message);
    }
  };

  const getPersona = async () => {
    try {
      const response = await fetch("http://localhost:5000/persona");
      const jsonData = await response.json();

      setPersona(jsonData);
    } catch (err) {
      console.error(err.message);
    }
  };

  useEffect(() => {
    getPersona();
  }, []);

  console.log(persona);

  return (
    <Fragment>
      {" "}
      <InputPersona />
      <table className="table mt-5 text-center">
        <thead>
```

```
          <tr>
            <th>Numero doc</th>
            <th>Nombre</th>
            <th>Apellidos</th>
            <th>Fecha nacim</th>
            <th>Tipo doc</th>
            <th>Edad</th>
            <th>Estatura</th>
            <th>Situacion milit</th>
            <th>Sexo</th>
            <th>Nivel estudios</th>
          </tr>
        </thead>
        <tbody>
          {/*<tr>
            <td>John</td>
            <td>Doe</td>
            <td>john@example.com</td>
          </tr> */}
          {persona.map(persona => (
            <tr key={persona.numero_doc}>
              <td>{persona.numero_doc}</td>
              <td>{persona.nombre}</td>
              <td>{persona.apellidos}</td>
              <td>{persona.fecha_nacimiento}</td>
              <td>{persona.tipo_doc}</td>
              <td>{persona.edad}</td>
              <td>{persona.estatura}</td>
              <td>{persona.situacion_militar}</td>
              <td>{persona.sexo}</td>
              <td>{persona.nivel_de_estudios}</td>
              <td>
                <EditPersona persona={persona} />
              </td>
              <td>
                <button
                  className="btn btn-danger"
                  onClick={() => deletePersona(persona.numero_doc)}
                >
                  Eliminar
                </button>
              </td>
            </tr>
          ))}
        </tbody>
      </table>
    </Fragment>
  );
```

```
};

export default ListPersona;
```

/personaEditPersona.js

```
import React, { Fragment, useState } from "react";

const EditPersona = ({ persona }) => {
  const [nombre, setNombrePersona] = useState(persona.nombre);
  const [apellidos, setApellidosPersona] = useState(persona.apellidos);
  const [fecha_nacimiento, setFechaNacimientoPersona] = useState(persona.fecha
_nacimiento);
  const [tipo_doc, setTipoDocPersona] = useState(persona.tipo_doc);
  const [edad, setEdadPersona] = useState(persona.edad);
  const [estatura, setEstaturaPersona] = useState(persona.estatura);
  const [situacion_militar, setSituacionMilitarPersona] = useState(persona.sit
uacion_militar);
  const [sexo, setSexoPersona] = useState(persona.sexo);
  const [nivel_de_estudios, setNivelEstudiosPersona] = useState(persona.nivel_
de_estudios);
  const [fk_persona_cod_municipio, setFkPersonaCodMunicipio] = useState(person
a.fk_persona_cod_municipio);

  //edit nombre_persona function

  const updateNombrePersona = async e => {
    e.preventDefault();
    try {
      const body = { nombre, apellidos, fecha_nacimiento, tipo_doc, edad, esta
tura, situacion_militar, sexo, nivel_de_estudios, fk_persona_cod_municipio };
      const response = await fetch(
        `http://localhost:5000/persona/${persona.numero_doc}`,
        `http://localhost:5000/persona/${persona.nombre}`,
        `http://localhost:5000/persona/${persona.apellidos}`,
        `http://localhost:5000/persona/${persona.fecha_nacimiento}`,
        `http://localhost:5000/persona/${persona.tipo_doc}`,
        `http://localhost:5000/persona/${persona.edad}`,
        `http://localhost:5000/persona/${persona.estatura}`,
        `http://localhost:5000/persona/${persona.sexo}`,
        `http://localhost:5000/persona/${persona.nivel_de_estudios}`,
        `http://localhost:5000/persona/${persona.fk_persona_cod_municipio}`,
        {
```

```
        method: "PUT",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify(body)
      }
    );

    window.location = "/persona";
  } catch (err) {
    console.error(err.message);
  }
};

return (
  <Fragment>
    <button
      type="button"
      className="btn btn-warning"
      data-toggle="modal"
      data-target={`#id${persona.numero_doc}`}
    >
      Edit
    </button>
    <div
      className="modal"
      id={`id${persona.numero_doc}`}
      onClick={() => setNombrePersona(persona.nombre)}
    >
      <div className="modal-dialog">
        <div className="modal-content">
          <div className="modal-header">
            <h4 className="modal-title">Editar Persona</h4>
            <button
              type="button"
              className="close"
              data-dismiss="modal"
              onClick={() => setNombrePersona(persona.nombre)}
            />
              <button
              type="button"
              className="close"
              data-dismiss="modal"
              onClick={() => setApellidosPersona(persona.apellidos)}
            />
              <button
              type="button"
              className="close"
              data-dismiss="modal"
```

```
                onClick={() => setFechaNacimientoPersona(persona.fecha_nacimie
nto)}
              />
                <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setTipoDocPersona(persona.tipo_doc)}
              />
                <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setEdadPersona(persona.edad)}
              />
                <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setEstaturaPersona(persona.estatura)}
              />
              <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setSituacionMilitarPersona(persona.situacion_mi
litar)}
              />
              <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setSexoPersona(persona.sexo)}
              />
              <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setNivelEstudiosPersona(persona.nivel_de_estudi
os)}
              />
              <button
                type="button"
                className="close"
                data-dismiss="modal"
                onClick={() => setFkPersonaCodMunicipio(persona.fk_persona_cod
_municipio)}
              />
```

```
            &times;
          </div>

          <div className="modal-body">
            Nombre:
            <input
              type="text"
              className="form-control"
              value={nombre}
              onChange={e => setNombrePersona(e.target.value)}
            />
            Apellidos:
            <input
              type="text"
              className="form-control"
              value={apellidos}
              onChange={e => setApellidosPersona(e.target.value)}
            />
            Fecha Nacimiento:
            <input
              type="text"
              className="form-control"
              value={fecha_nacimiento}
              onChange={e => setFechaNacimientoPersona(e.target.value)}
            />
            Tipo documento:
            <input
              type="text"
              className="form-control"
              value={tipo_doc}
              onChange={e => setTipoDocPersona(e.target.value)}
            />
            Edad:
            <input
              type="text"
              className="form-control"
              value={edad}
              onChange={e => setEdadPersona(e.target.value)}
            />
            Estatura:
            <input
              type="text"
              className="form-control"
              value={estatura}
              onChange={e => setEstaturaPersona(e.target.value)}
            />
            Situacion militar:
            <input
```

```
            type="text"
            className="form-control"
            value={situacion_militar}
            onChange={e => setSituacionMilitarPersona(e.target.value)}
          />
          Sexo:
          <input
            type="text"
            className="form-control"
            value={sexo}
            onChange={e => setSexoPersona(e.target.value)}
          />
          Nivel de estudios:
          <input
            type="text"
            className="form-control"
            value={nivel_de_estudios}
            onChange={e => setNivelEstudiosPersona(e.target.value)}
          />
          Municipio:
          <input
            type="text"
            className="form-control"
            value={fk_persona_cod_municipio}
            onChange={e => setFkPersonaCodMunicipio(e.target.value)}
          />
        </div>

        <div className="modal-footer">
          <button
            type="button"
            className="btn btn-warning"
            data-dismiss="modal"
            onClick={e => updateNombrePersona(e)}
          >
            Edit
          </button>
          <button
            type="button"
            className="btn btn-danger"
            data-dismiss="modal"
            onClick={() => setNombrePersona(persona.numero_doc)}
          >
            Close
          </button>
        </div>
      </div>
    </div>
```

```
      </div>
    </Fragment>
  );
};

export default EditPersona;
```

/persona/InputPersona.js

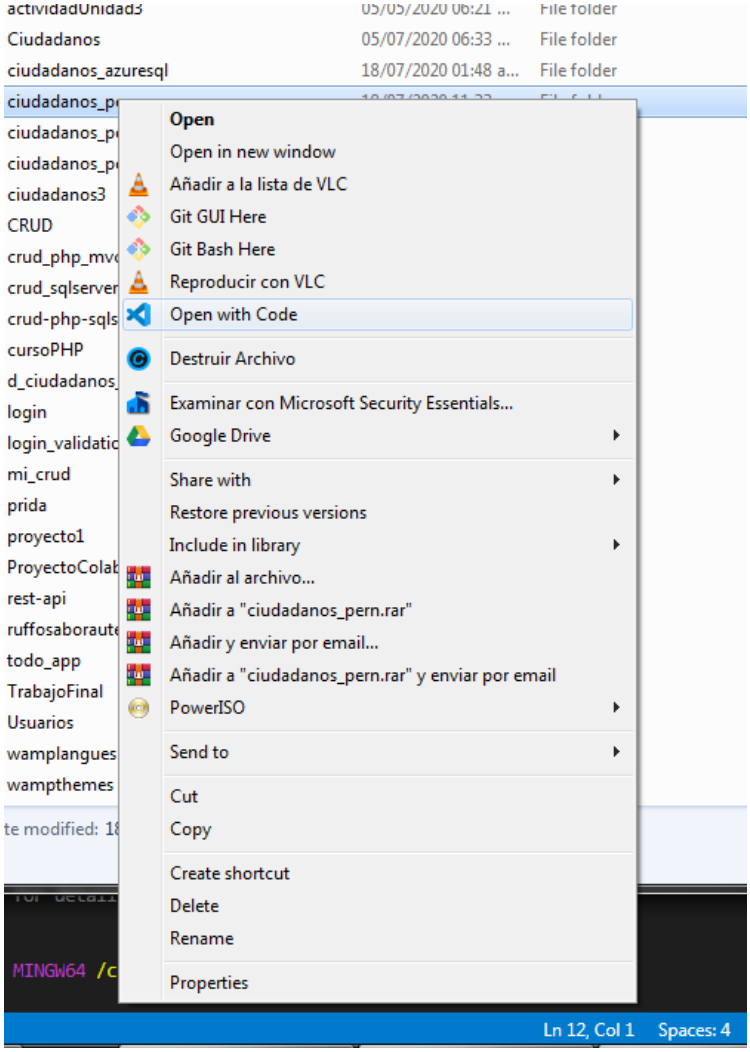
```
import React, { Fragment, useState } from "react";

const InputPersona = () => {
  const [nombre_provincia, setNombreProvincia] = useState("");
  const [nombre, setNombrePersona] = useState("");
  const [apellidos, setApellidosPersona] = useState("");
  const [fecha_nacimiento, setFechaNacimientoPersona] = useState("");
  const [tipo_doc, setTipoDocPersona] = useState("");
  const [edad, setEdadPersona] = useState("");
  const [estatura, setEstaturaPersona] = useState("");
  const [situacion_militar, setSituacionMilitarPersona] = useState("");
  const [sexo, setSexoPersona] = useState("");
  const [nivel_de_estudios, setNivelEstudiosPersona] = useState("");
  const [fk_persona_cod_municipio, setFkPersonaCodMunicipio] = useState("");

  const onSubmitForm = async e => {
    e.preventDefault();
    try {
      const body = { nombre, apellidos, fecha_nacimiento, tipo_doc, edad, esta
tura, situacion_militar, sexo, nivel_de_estudios, fk_persona_cod_municipio };
      const response = await fetch("http://localhost:5000/persona", {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify(body)
      });

      window.location = "/persona";
    } catch (err) {
      console.error(err.message);
    }
  };

  return (
    <Fragment>
      <h1 className="text-center mt-5">PERSONA</h1>
      <form className="dmt-3 form-group" onSubmit={onSubmitForm}>
        <input
```

```
          type="text"
          className="form-control m-2"
          value={nombre}
          placeholder="Nombre..."
          onChange={e => setNombrePersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={apellidos}
          placeholder="Apellidos..."
          onChange={e => setApellidosPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={fecha_nacimiento}
          placeholder="Fecha de nacimiento..."
          onChange={e => setFechaNacimientoPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={tipo_doc}
          placeholder="Tipo de documento..."
          onChange={e => setTipoDocPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={edad}
          placeholder="Edad..."
          onChange={e => setEdadPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={estatura}
          placeholder="Estatura..."
          onChange={e => setEstaturaPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={situacion_militar}
          placeholder="Situación militar..."
          onChange={e => setSituacionMilitarPersona(e.target.value)}
        />
```

```
        <input
          type="text"
          className="form-control m-2"
          value={sexo}
          placeholder="Sexo..."
          onChange={e => setSexoPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={nivel_de_estudios}
          placeholder="Nivel de estudios..."
          onChange={e => setNivelEstudiosPersona(e.target.value)}
        />
        <input
          type="text"
          className="form-control m-2"
          value={fk_persona_cod_municipio}
          placeholder="Municipio..."
          onChange={e => setFkPersonaCodMunicipio(e.target.value)}
        />
        <button className="btn btn-success ml-3 mr-5">Agregar</button>
      </form>
    </Fragment>
  );
};

export default InputPersona;
```

Procedimiento para correr la app

1. Descomprimir el archivo ciudadanos_pern en una carpeta con el mismo nombre.
2. Abrir la carpeta con VS Code o tu editor de código preferido.



3. Utilizando el bash de la consola y corremos el siguiente comando y esperamos que instale los módulos node.

```
Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern (master)
$ npm init -y
```

4. Cambiamos a la carpeta server y corremos el siguiente comando:

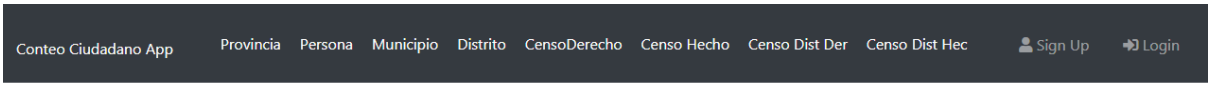
```
Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern (master)
$ cd server

Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern/server (master)
$ nodemon start
[nodemon] 2.0.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node start index.js`
Servidor corriendo en puerto 5000
```

Ya tenemos el servidor corriendo.

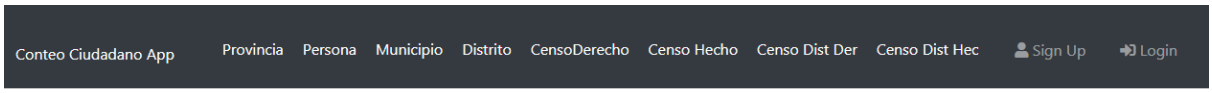
Ahora ejecutamos la app, para lo cual nos regresamos al fichero base y nos cambiamos al fichero client:

```
Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern (master)
$ cd client

Rafael@RAFAEL-PC MINGW64 /c/wamp64/www/ciudadanos_pern/client (master)
$ npm start

> client@0.1.0 start C:\wamp64\www\ciudadanos_pern\client
> react-scripts start

Starting the development server...
```

Inicio



Bienvenidos



PROVINCIA

Nombre PROVINCIA a agregar...

Agregar

Nombre Provincia	Editar	Eliminar
Cundinamarca	Edit	Eliminar
Choco	Edit	Eliminar
Huila	Edit	Eliminar
Atlantico	Edit	Eliminar
Amazonas	Edit	Eliminar
Sucre	Edit	Eliminar
Cesar	Edit	Eliminar
Bolívar	Edit	Eliminar
Nariño	Edit	Eliminar

Conteo Ciudadano App | Provincia | Persona | Municipio | Distrito | CensoDerecho | Censo Hecho | Censo Dist Der | Censo Dist Hec | Sign Up | Login

PERSONA

Nombre...

Apellidos...

Fecha de nacimiento...

Tipo de documento...

Edad...

Estatura...

Situación militar...

Sexo...

Nivel de estudios...

Municipio...

Agregar

Numero doc	Nombre	Apellidos	Fecha nacim	Tipo doc	Edad	Estatura	Situacion milit	Sexo	Nivel estudios		
33159774	Chun	Li	1993-03-05T04:00:00.000Z	Cedula de Extanjeria	27	1.78	SIN DEFINIR	MUJ	Primaria	Edit	Eliminar
9017017	Hanzo	Hattori	1975-09-09T05:00:00.000Z	Cedula de Extanjeria	45	1.90	SIN DEFINIR	HOM	Secundaria	Edit	Eliminar
8100100	Hermenegildo	Salas	2000-02-01T05:00:00.000Z	Cedula	20	1.88	DEFINIDA	HOM	Universitario	Edit	Eliminar
73207500	Rafael	Ortiz	1984-04-06T05:00:00.000Z	Cedula	25	1.88	DEFINIDA	HOMBRE	Universitario	Edit	Eliminar
9047015	Galford	Kazashi	1980-01-05T05:00:00.000Z	Cedula	27	1.91	SIN DEFINIR	HOM	Superior	Edit	Eliminar

CONCLUSIONES

- Se aprendió a conectar la web app en lenguaje JavaScript con librería React para trabajar con una base de datos PostgreSQL.
- Se fusionó también lo aprendido en la asignatura Desarrollo de software web con esta, de manera que la web app realizará CRUD’s en la base de datos.
- Se aprendió a utilizar funciones y triggers para darle más dinamismo a la base de datos.
- Se aprendió el proceso de realizar backup y restauraciones de base de datos en el motor en mención.