

## **EXAMEN DE CONOCIMIENTOS TÉCNICOS**

### **Instrucciones:**

- Complete las tareas y responda las preguntas en su IDE o entorno de desarrollo preferido.
- Puede utilizar cualquier biblioteca o marco de trabajo de C# .NET 7 que considere apropiado.
- Asegúrese de que su código sea limpio, eficiente y siga las mejores prácticas.
- Proporcione comentarios y documentación adecuada si es necesario.

### **Para las siguientes tablas (que ya existen en la BD):**

```
CREATE TABLE Persona (  
    ID_PERSONA INT PRIMARY KEY,  
    NOMBRE VARCHAR(50),  
    APELLIDO_PATERNO VARCHAR(50),  
    APELLIDO_MATERNO VARCHAR(50),  
    FECHA_NACIMIENTO DATE,  
    DIRECCION VARCHAR(100)  
);
```

```
CREATE TABLE Casa (  
    ID_CASA INT PRIMARY KEY,  
    DIRECCION VARCHAR(100),  
    NUMERO_HABITACIONES INT,  
    PROPIETARIO_ID INT,  
    FOREIGN KEY (PROPIETARIO_ID) REFERENCES Persona(ID_PERSONA)  
);
```

### **Datos de conexión a la BD:**

Servidor: 198.38.83.200

BD: tsseirl\_test\_persona

Usuario: tsseirl\_postulant

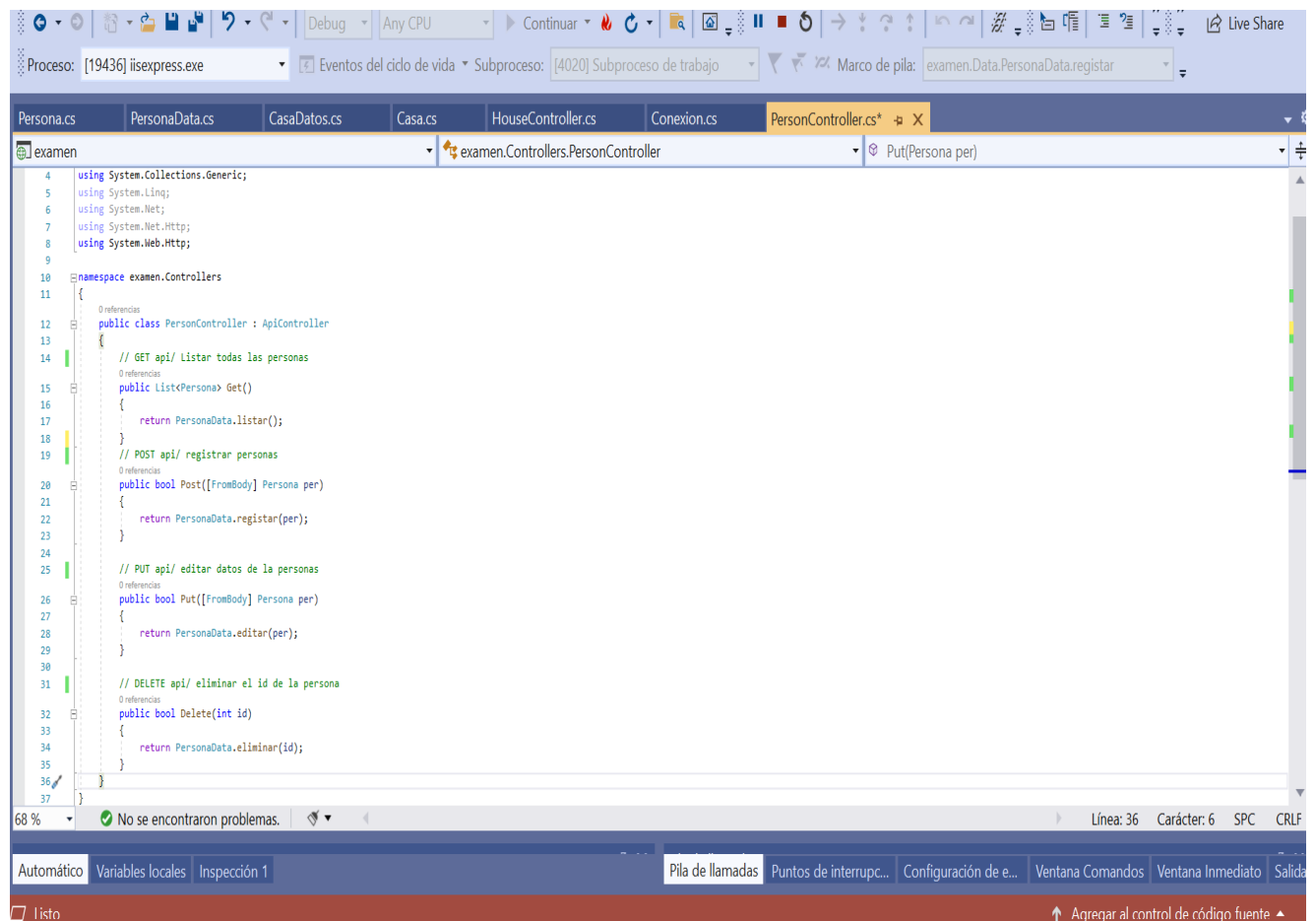
Contraseña: g43tG33P

## IMPLEMENTE LOS SIGUIENTES SERVICIOS WEB API RESTFUL:

### Parte 1: Creación de la API de Persona

Cree un controlador PersonController con las acciones CRUD (Crear, Leer, Actualizar, Eliminar) para la entidad "Persona". Utilice rutas y verbos HTTP adecuados para cada acción.

Creación de las apis para poder realizar un GRUD (Crear, Leer, Actualizar, Eliminar).



The screenshot shows the Visual Studio IDE with the following details:

- Process:** [19436] iisexpress.exe
- Subprocess:** [4020] Subproceso de trabajo
- Stack:** examen.Data.PersonaData.registrar
- Solution Explorer:** Persona.cs, PersonaData.cs, CasaDatos.cs, Casa.cs, HouseController.cs, Conexion.cs, PersonController.cs\*
- Code Editor:** `examen.Controllers.PersonController` with the following code:

```
4 using System.Collections.Generic;
5 using System.Linq;
6 using System.Net;
7 using System.Net.Http;
8 using System.Web.Http;
9
10 namespace examen.Controllers
11 {
12     public class PersonController : ApiController
13     {
14         // GET api/ Listar todas las personas
15         public List<Persona> Get()
16         {
17             return PersonaData.listar();
18         }
19
20         // POST api/ registrar personas
21         public bool Post([FromBody] Persona per)
22         {
23             return PersonaData.registrar(per);
24         }
25
26         // PUT api/ editar datos de la personas
27         public bool Put([FromBody] Persona per)
28         {
29             return PersonaData.editar(per);
30         }
31
32         // DELETE api/ eliminar el id de la persona
33         public bool Delete(int id)
34         {
35             return PersonaData.eliminar(id);
36         }
37     }
38 }
```
- Status Bar:** 68 % zoom, No se encontraron problemas. (Linea: 36, Carácter: 6, SPC, CRLF)
- Toolbox:** Automático, Variables locales, Inspección 1, Pila de llamadas, Puntos de interrupción, Configuración de e..., Ventana Comandos, Ventana Inmediato, Salida.
- Bottom Bar:** Listo, Agregar al control de código fuente.

Implemente la acción para listar todas las personas (GET /api/person) que devuelva una lista de todas las personas en formato JSON.

La API GET nos permite listar todos los datos de una tabla de un servidor de base de datos.

# Online REST & SOAP API Testing Tool

ReqBin is an online API testing tool for REST and SOAP APIs. Test API endpoints by making API requests directly from your browser. Test API responses with built-in JSON and XML validators. Load test your API with hundreds of simulated concurrent connections. Generate code snippets for API automation testing frameworks. Share and discuss your API requests online.

https://localhost:44356/api/person

GET

EXT

Send

Content

Authorization

Headers

Raw (2)

JSON (application/json)

Status: 200 () Time: 362 ms Size: 0.74 kb

Content (36)

Headers (11)

Raw (13)

JSON

```
"apellido_paterno": "Salaz",
"apellido_materno": "Gree",
"fecha_nacimiento": "1999-10-10T00:00:00",
"direccion": "Los Olivos"
}, {
  "idPersona": 4,
  "nombre": "Lius",
  "apellido_paterno": "Contreras",
  "apellido_materno": "Arenas",
  "fecha_nacimiento": "1969-10-10T00:00:00",
  "direccion": "Comas"
}, {
  "idPersona": 5
```

Implemente la acción para crear una nueva persona (POST /api/person). Asegúrese de que la entrada se valide correctamente antes de agregar la persona a la base de datos.

La API POST nos permite Registrar un nuevo registro a nuestra base de datos .

Curl Python JavaScript Node.js PHP Java JSON XML

Test API responses with built-in JSON and XML validators. Load test your API with hundreds of simulated concurrent connections. Generate code snippets for API automation testing frameworks. Share and discuss your API requests online.

Contact Account

https://localhost:44356/api/person

POST

EXT

Send

Content (8)

Authorization

Headers

Raw (13)

JSON (application/json)

```
{
  "nombre": "Lucho Pepe",
  "apellido_paterno": "Reyes",
  "apellido_materno": "Dias",
  "fecha_nacimiento": "1979-10-10T00:00:00",
  "direccion": "Lince"
}
```

Implemente la acción para actualizar una persona existente (PUT /api/person/{id}). Asegúrese de que la entrada se valide correctamente antes de actualizar la persona.

La API PUT nos permite Actualizar los datos de una registro de la tabla

Curl Python JavaScript Node.js PHP Java JSON XML

Contact Account

Test API responses with built-in JSON and XML validators. Load test your API with hundreds of simulated concurrent connections. Generate code snippets for API automation testing frameworks. Share and discuss your API requests online.

https://localhost:44356/api/person

PUT

EXT

Send

Content (8)

Authorization

Headers

Raw (13)

JSON (application/json)

```
{
  "idPersona": 5,
  "nombre": "Lucho Pepe",
  "apellido_paterno": "Examen ",
  "apellido_materno": "Examen",
  "fecha_nacimiento": "1979-10-10T00:00:00",
  "direccion": "Lince"
}
```

Implemente la acción para eliminar una persona (DELETE /api/person/{id}).

La API DELETE nos eliminar cualquier registro por si ID

Curl Python JavaScript Node.js PHP Java JSON XML

Contact Account

Online REST & SOAP API Testing Tool

ReqBin is an online API testing tool for REST and SOAP APIs. Test API endpoints by making API requests directly from your browser. Test API responses with built-in JSON and XML validators. Load test your API with hundreds of simulated concurrent connections. Generate code snippets for API automation testing frameworks. Share and discuss your API requests online.

https://localhost:44356/api/person/5

DELETE

EXT

Send

Content

Authorization

Headers

Raw (2)

JSON (application/json)

Status: 200 () Time: 173 ms Size: 0.00 kb

Content (1)

Headers (12)

Raw (14)

JSON

```
true
```

## Parte 2: Creación de la API de Casas por Persona

Cree un controlador HouseController con una acción para listar las casas de una persona específica. La acción debe aceptar el ID de la persona como parámetro y devolver una lista de casas asociadas a esa persona en formato JSON. Utilice una ruta adecuada.

[Curl](#) [Python](#) [JavaScript](#) [Node.js](#) [PHP](#) [Java](#) [JSON](#) [XML](#) [Contact](#) [Account](#)

### Online REST & SOAP API Testing Tool

ReqBin is an online API testing tool for REST and SOAP APIs. Test API endpoints by making API requests directly from your browser. Test API responses with built-in JSON and XML validators. Load test your API with hundreds of simulated concurrent connections. Generate code snippets for API automation testing frameworks. Share and discuss your API requests online.

GET EXT Send

Content Authorization Headers Raw (2) 🔗 🔄 ⚙️ ↔️ 💾

JSON (application/json) ⌵

Status: **200 ()** Time: **27 ms** Size: **0.16 kb**

Content (9) Headers (11) Raw (13) JSON

```
{
  "id_casa": 2,
  "direccion": "Av. las rocas 443 - Lima ",
  "numero_habitaciones": 5,
  "id_persona": 2,
  "nombre": "Pepe",
  "apellido_paterno": "Mendez",
  "apellido_materno": "Rios"
}
```

**Autor: KENEDYN JHONATAN RODRIGUEZ HURTADO**