participants-app

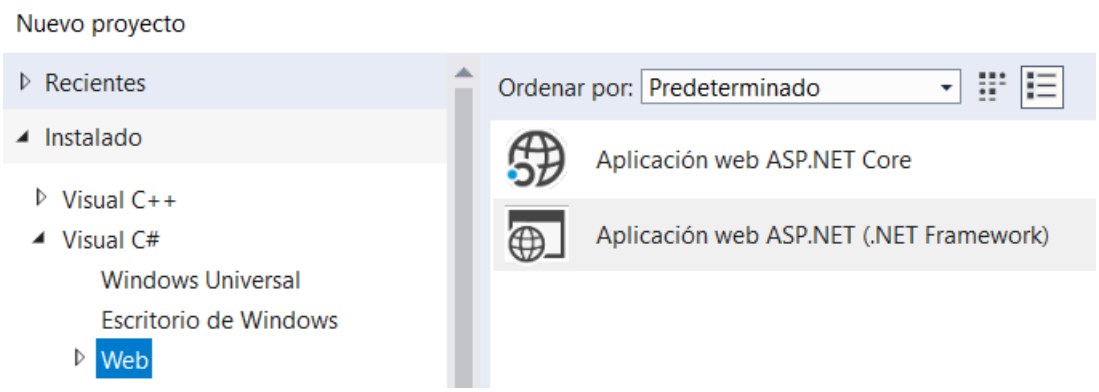
step-by-step

1. **Use SSMS to create the ParticipantDB database and the Participant table.**

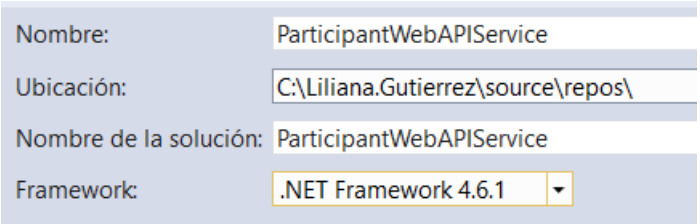
participant.sql - (lo...iliana Gutierrez (52))

```
CREATE DATABASE ParticipantDB
use ParticipantDB
GO

CREATE TABLE Participant
(
    initials nvarchar(5) PRIMARY KEY,
    name nvarchar(30),
    address nvarchar(80),
    preferredLanguage nvarchar(20)
)
GO

INSERT INTO Participant VALUES
    ('LGF', 'Liliana Gutiérrez', 'Lejos', 'Java'),
    ('AAH', 'Alejandro Arellano', 'Por allí', 'Java'),
    ('JLV', 'José Luis Vera', 'Cerca', 'C#')
;
```

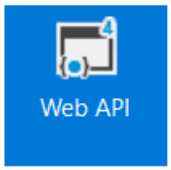
2. **Use MSVS to create a Web API to get info from the DB.**
    a. **Create the ASP.NET Application**



Name it ParticipantWebAPIService



Select Web API

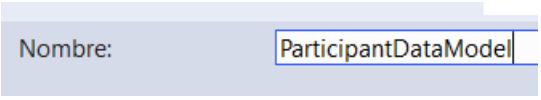


a. Create the model

Add a new ADO.NET Entity Data Model:

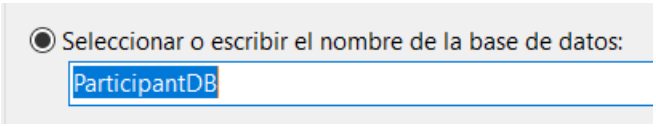
- (right-click) the models folder, Add, New Item…
- Select Data and ADO.NER Entity Data Model

- Name it ParticipantDataModel



- Select EF Designer from database

- Connect to the local server (.) and select de ParticipantDB you just create.

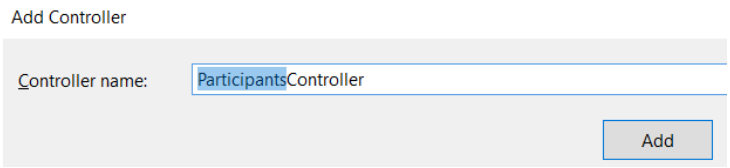


- Select the Participant table to include it in the model.7

b. Add a controller

(rigth-click) the Controllers folder and select Add, and then select Web API 2 Controller Empty

Name it ParticipantsController



The code of the ParticipantsController should look like this

```
namespace ParticipantWebAPIService.Controllers
{
    public class ParticipantsController : ApiController
    {
        public IEnumerable<Participant> Get()
        {
            ParticipantDBEntities entities = new ParticipantDBEntities();
            return entities.Participant.ToList();

        }
        public Participant Get(string initials)
        {
            ParticipantDBEntities entities = new ParticipantDBEntities();
            return entities.Participant.FirstOrDefault(p => p.initials == initials);
        }

    }
}
```

Change the default routes in the file WebApiConfig.cs from `id` to `initials`.

```
config.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{initials}",
    defaults: new { initials = RouteParameter.Optional }
);
```

Add the following to the Web.config file

```
<system.webServer>
  <httpProtocol>
  <customHeaders>
   <add name="Access-Control-Allow-Origin" value="*" />
   <add name="Access-Control-Allow-Headers" value="Content-Type" />
   <add name="Access-Control-Allow-Methods" value="GET, POST, PUT, DELETE" />
 </customHeaders>
 </httpProtocol>
```

☑ Use the browser to verify your application works as expected.

This XML file does not appear to have any style information associated with it. The d
shown below.

```
▼<ArrayOfParticipant xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://schemas.datacontract.org/2004/07/ParticipantWebAPIService">
  ▼<Participant>
     <address>Por allí</address>
     <initials>AAH</initials>
     <name>Alejandro Arellano</name>
     <preferredLanguage>Java</preferredLanguage>
   </Participant>
  ▼<Participant>
     <address>Cerca</address>
     <initials>JLV</initials>
     <name>José Luis Vera</name>
     <preferredLanguage>C#</preferredLanguage>
   </Participant>
  ▼<Participant>
     <address>Lejos</address>
     <initials>LGF</initials>
     <name>Liliana Gutiérrez</name>
     <preferredLanguage>Java</preferredLanguage>
   </Participant>
 </ArrayOfParticipant>
```

This XML file does not appear to have any style information associated with it. The d
shown below.

```
▼<Participant xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://schemas.datacontract.org/2004/07/ParticipantWebAPIService">
   <address>Lejos</address>
   <initials>LGF</initials>
   <name>Liliana Gutiérrez</name>
   <preferredLanguage>Java</preferredLanguage>
 </Participant>
```

3. Use the HTTP module to call the ASP.NET Web API service.

a. Import the HttpModule in the app.module.ts file

```
import { HttpModule } from '@angular/http';

imports: [
  BrowserModule,
  AppRoutingModule,
  FormsModule,
  HttpModule
],
```

Using the participant-data.service.ts.

b. Do the following imports.

```
import { Http, Response } from '@angular/http';
import { Observable } from 'rxjs/Observable';
import 'rxjs/add/operator/map'
```

c. Use the constructor to inject[1] the service.

Remember that this short-notation means:

- The creation of the private variable _http
- The creation the constructor parameter with the same name.
- The initialization of this._http with the parameter

```
constructor(
  private _http: Http
) { }
```

d. Use the _http to issue Web-service calls.

Modify the `getParticipants()` method, so it now returns an observable.

[1] Inject the service so far means, to have an instance of that service.

```
@Injectable()
export class ParticipantDataService {

  urlWebAPI= "http://localhost:56618/api/participants/";
  headers: Headers = new Headers({
    'Content-Type': 'application/json'
   });
  options = new RequestOptions({ headers: this.headers });

  constructor(
    private _http: Http
  ) { }

  getParticipants(): Observable<Participant[]> {
    let observableParticipants = this._http.get(this.urlWebAPI)
        .map((response: Response) => <Participant[]>response.json());
    return observableParticipants;
  }
```

Comment by now the putParticipant method.

e. Subscribe to the service in the name-list.component.ts.

```
ngOnInit() {
  this.participantDataService.getParticipants()
    .subscribe((participantsData)=>this.participants = participantsData);
}
```

f. Comment by now the use of the putParticipant() method in the participant-form.component.ts.

```
 newHandler(participant1: Participant){
   //this.participantDataService.putParticipant(participant1);
 }
```

☑ Run the Web API and make sure your angular app is served. Then Insert a participant record directly in the DB and verify you get the data in the browser.

4. Modify the Web API to post info to the DB.

a. Add the following to the ParticipantsController

```
public void Post([FromBody]Participant participant)
{
    ParticipantDBEntities entities = new ParticipantDBEntities();
    entities.Participant.Add(participant);
    entities.SaveChanges();
}

public IHttpActionResult Options()
{
    HttpContext.Current.Response.AppendHeader("Allow", "GET, OPTIONS");
    return Ok();
}
```

5. In the angular application, modify the putParticipant method in the participant-data.service.ts file.