

Το Project αποτελείται από τα βασικά αρχεία "program.cs, PositionControllers.cs" καθώς και στο φάκελο model βρίσκονται τα αρχεία του database.

Program.cs: Το συγκεκριμένο αρχείο βασίστηκε στο βασικό WebApi template που δημιουργείται με κάποιο νέο project, και επεκτάθηκε αναλόγως. Συγκεκριμένα προστέθηκε η γραμμή "builder.Services.AddControllers();" και "app.MapControllers();" για την ομαλή λειτουργία των controllers. Επίσης χρησιμοποιήθηκε το EntityFrameworkCore για να προστεθεί η βάση με το "builder.Services.AddDbContext...". Παραπάνω λόγια για τη βάση στη συνέχεια.

PositionController.cs: Στο ίδιο πνεύμα με το program.cs βασίστηκε στο generic template για controllers με αρκετές παραπάνω προσθήκες. Αρχικοποιούμε μεταβλητή _db η οποία αρχικοποιείται αμέσως μετά με τον ανάλογο constructor και τη χρησιμοποιούμε αργότερα για τη δημιουργία αντικειμένων μέσω queries. Το [HttpPost] προσθέτει στον πίνακα Positions της βάσης δεδομένων τις κατάλληλες μεταβλητές name, longitude, latitude ενώ το httpget εκτυπώνει μια λίστα που περιέχει αυτές τις μεταβλητές. Δεν χρησιμοποιούμε κάποιο query γιατί μας καλύπτει πλήρως το _db.SaveChanges(); του EntityFrameworkCore. Ακολουθούν αποτελέσματα από την εκτέλεση του προγράμματος:

The screenshot shows a web API client interface with the following details:

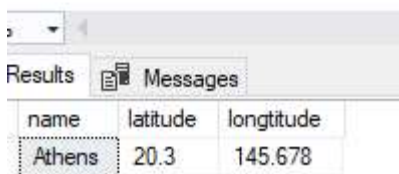
- Method:** POST
- URL:** /Position/add
- Parameters:** No parameters
- Request body:** application/json
- Request body content:**

```
{  "name": "Athens",  "latitude": 20.3,  "longitude": 145.678}
```
- Execute button:** A blue button labeled "Execute".
- Responses:**

Code	Description	Links
200	Success	No links

Πατώντας execute οι εγγραφές αυτές περνάνε στο Position database. Για να ελέγξουμε πως όντως εισήχθαν στον πίνακα δίνουμε ένα απλό select query στον sql server:

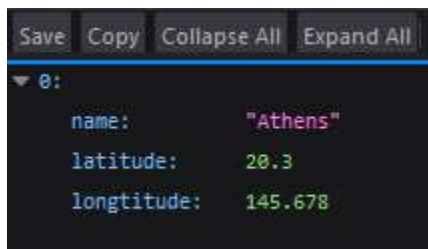
```
select *from Position;
```



name	latitude	longitude
Athens	20.3	145.678

Παραπάνω βλέπουμε πως όντως τα ορίσματα που δώσαμε πέρασαν στον πίνακα της βάσης μας.

Όσο αφορά το [HttpGet] έχουμε:



```
Save Copy Collapse All Expand All
0:
  name: "Athens"
  latitude: 20.3
  longitude: 145.678
```

Βλέπουμε λοιπόν πως εμφανίζει τις μεταβλητές του πίνακα σε json format.

Models:

Η βάση και ο πίνακας αρχικοποιήθηκαν μέσω του sql server και ο κώδικας που υπάρχει μέσα στα αρχεία του Models δημιουργήθηκε αυτόματα όταν εισήγαγα την βάση στο project χρησιμοποιώντας το global dotnet-ef και δεν έχει υπάρξει κάποια αλλαγή. Το μόνο που χρησιμοποίησα στο κώδικα μου είναι η κλάση Position που υπάρχει μέσα στο αρχείο Position.cs

Απόσταση Haversine:

Όπου έβρισκα ότι ισούται το όνομα με αυτό που έδωσε ο χρήστης τότε έφτιαχνα ένα νέο αντικείμενο και κρατούσα τις μεταβλητές longitude και latitude (Υπάρχουν 2 τέτοια αντικείμενα, 1 για κάθε περιοχή). Ο υπόλοιπος κώδικας είναι απλή μετατροπή του αλγορίθμου εύρεσης απόστασης Haversine στη c#. Ακολουθεί ενδεικτική εκτέλεση:

για τις εξής συντεταγμένες:

Code	Details
200	<div><p>Response body</p><pre>[{ "name": "Athens", "latitude": 45.67, "longitude": 32.34 }, { "name": "Chios", "latitude": 678.67, "longitude": 123.34 }]</pre><p>Response headers</p></div>

Η απόσταση haversine είναι η εξής:

Server response	
Code	Details
200	
	Response body
	<pre>13705.34102969023</pre>
	Response headers
	<pre>content-type: application/json; charset=utf-8 date: Thu,08 Feb 2024 17:10:44 GMT server: Kestrel x-firefox-spdy: h2</pre>
Responses	