



Kauno technologijos universitetas

Informatikos fakultetas

P175B120 Paslaugų programavimas debesų kompiuterijoje

Laboratorinis darbas Nr. 2

Nedas Liaudanskis IFF-1/9

Studentas

dėst. Pilkauskas Vytautas

Dėstytojas

Turinys

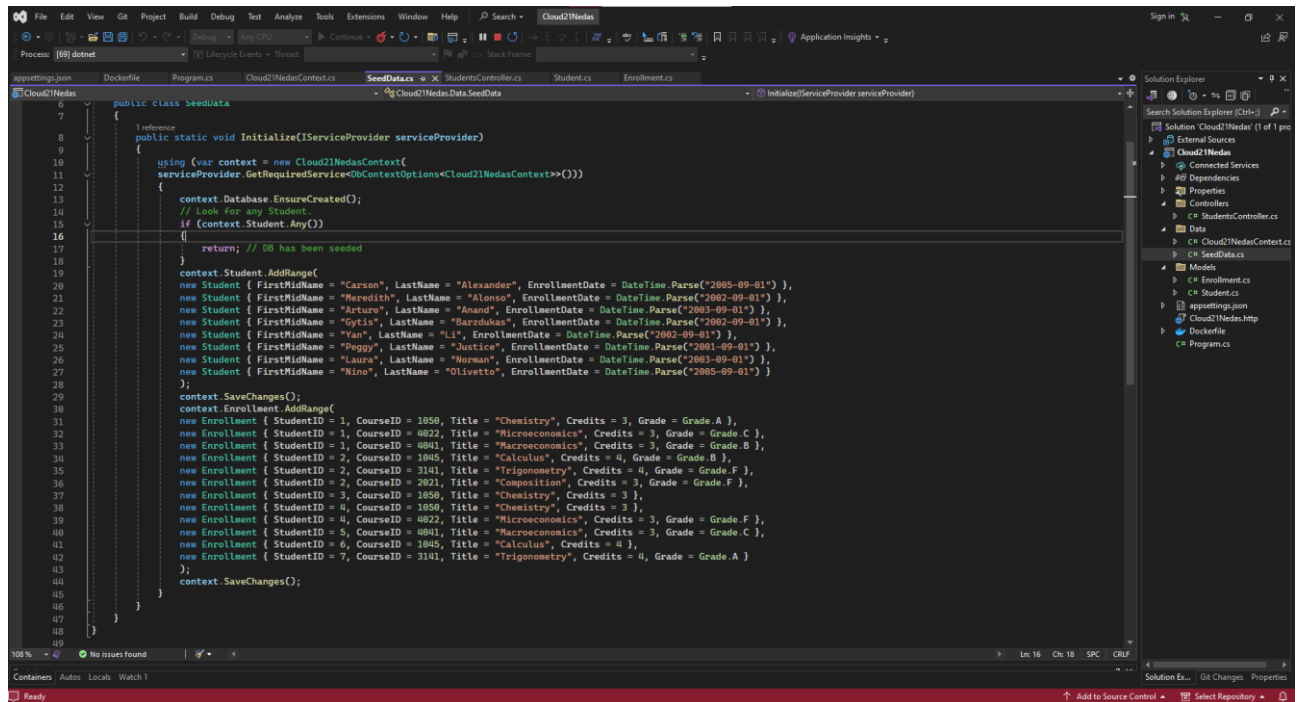
Lab 2.1 Dalis	3
Lab 2.2 Dalis	6
Lab 3.3 dalis	10

Paveikslėlių sąrašas

pav. 1 SeedData.cs	3
pav. 2 Program.cs	4
pav. 3 DockerFile	4
pav. 4 Students controller	5
pav. 5 Docker Desktop	5
pav. 6 Swagger	6
pav. 7 Program.cs	7
pav. 8 Student.cs	7
pav. 9 Students Controller	8
pav. 10 Docker-compose file	8
pav. 11 Docker-override file	9
pav. 12 Docker desktop	9
pav. 13 Swagger	10
pav. 14 Docker-compose override	10
pav. 15 Ocelot.json file	11
pav. 16 Docker-compose file	11
pav. 17 Docker desktop	12
pav. 18 Api response	12

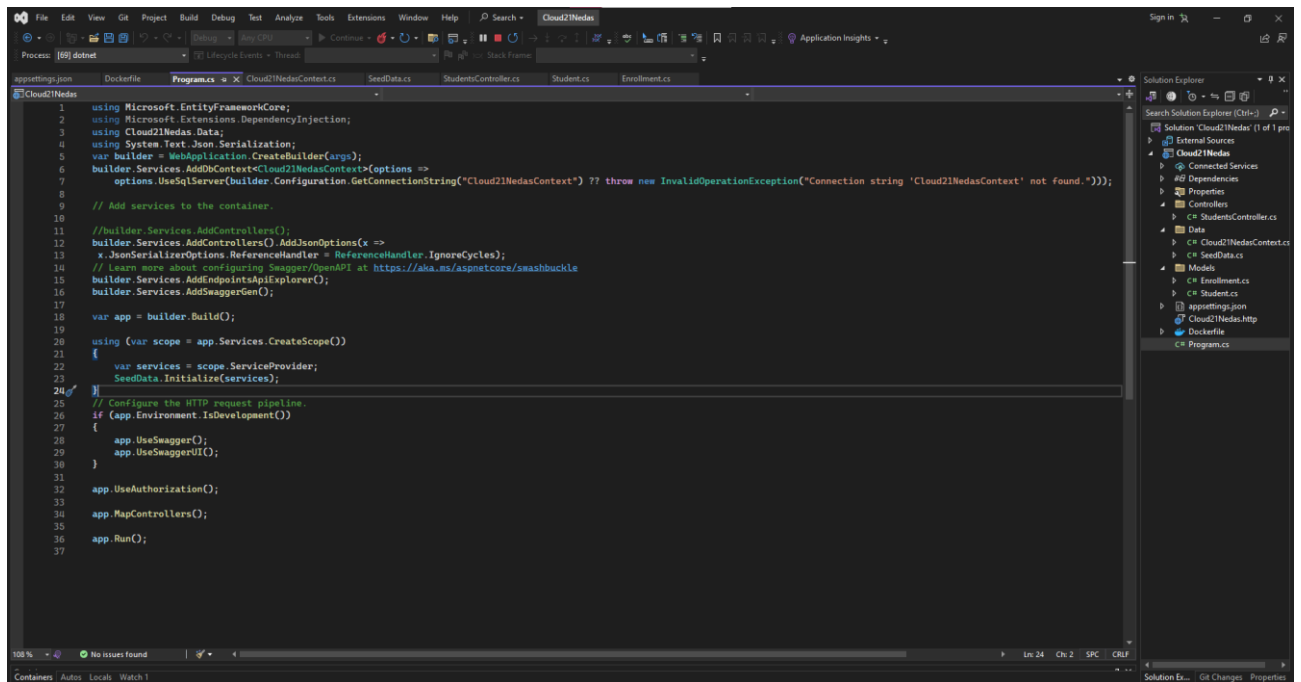
Lab 2.1 Dalis

Sukūriau web API projektą, kuriame yra Docker palaikymas. Tai leidžia sukurti sql serverį, kuriame bus saugomi duomenys apie studentus. Duomenis užpildomi naudojant SeedData.cs klasę. Ištestuoti veikimą, naudojame swagger. Jis padeda sukurti CRUDO operacijų užklausas ir duoti atsakymą. Viskam atlikti buvo naudotas Visual Studio.

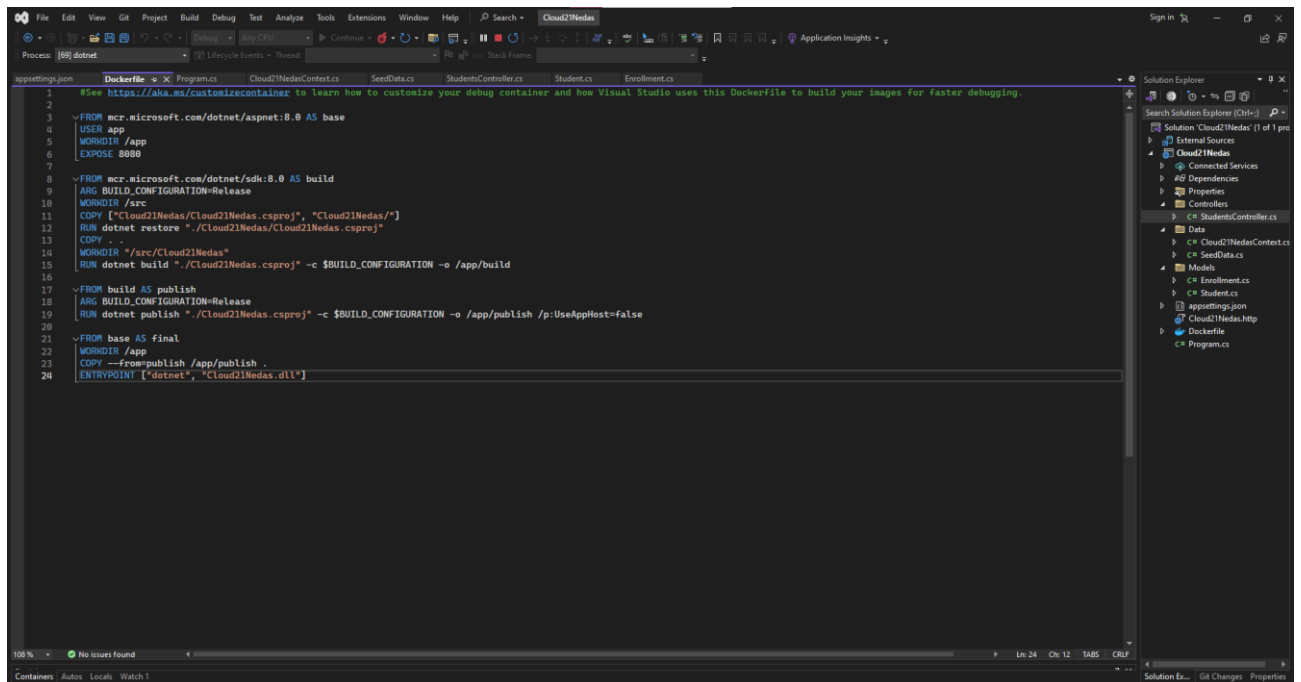


```
6 public class SeedData
7 {
8     [Obsolete]
9     public static void Initialize(IServiceProvider serviceProvider)
10     {
11         using (var context = new Cloud21NedasContext(
12             serviceProvider.GetRequiredService<DbContextOptions<Cloud21NedasContext>>()))
13         {
14             context.Database.EnsureCreated();
15             // Look for any Student
16             if (context.Student.Any())
17             {
18                 return; // DB has been seeded
19             }
20             context.Student.AddRange(
21                 new Student { FirstName = "Carson", LastName = "Alexander", EnrollmentDate = DateTime.Parse("2005-09-01") },
22                 new Student { FirstName = "Meredith", LastName = "Alonso", EnrollmentDate = DateTime.Parse("2002-09-01") },
23                 new Student { FirstName = "Arturo", LastName = "Anand", EnrollmentDate = DateTime.Parse("2003-09-01") },
24                 new Student { FirstName = "Gytis", LastName = "Bazrubius", EnrollmentDate = DateTime.Parse("2002-09-01") },
25                 new Student { FirstName = "Yan", LastName = "Li", EnrollmentDate = DateTime.Parse("2002-09-01") },
26                 new Student { FirstName = "Peggy", LastName = "Justice", EnrollmentDate = DateTime.Parse("2001-09-01") },
27                 new Student { FirstName = "Laura", LastName = "Norman", EnrollmentDate = DateTime.Parse("2003-09-01") },
28                 new Student { FirstName = "Nino", LastName = "Olivetto", EnrollmentDate = DateTime.Parse("2005-09-01") }
29             );
30             context.SaveChanges();
31             context.Enrollment.AddRange(
32                 new Enrollment { StudentID = 1, CourseID = 1050, Title = "Chemistry", Credits = 3, Grade = Grade.A },
33                 new Enrollment { StudentID = 1, CourseID = 4022, Title = "Microeconomics", Credits = 3, Grade = Grade.C },
34                 new Enrollment { StudentID = 1, CourseID = 4041, Title = "Macroeconomics", Credits = 3, Grade = Grade.B },
35                 new Enrollment { StudentID = 2, CourseID = 1045, Title = "Calculus", Credits = 4, Grade = Grade.B },
36                 new Enrollment { StudentID = 2, CourseID = 3141, Title = "Trigonometry", Credits = 4, Grade = Grade.F },
37                 new Enrollment { StudentID = 2, CourseID = 2021, Title = "Composition", Credits = 3, Grade = Grade.F },
38                 new Enrollment { StudentID = 3, CourseID = 1050, Title = "Chemistry", Credits = 3 },
39                 new Enrollment { StudentID = 4, CourseID = 1050, Title = "Chemistry", Credits = 3 },
40                 new Enrollment { StudentID = 4, CourseID = 4022, Title = "Microeconomics", Credits = 3, Grade = Grade.F },
41                 new Enrollment { StudentID = 5, CourseID = 4041, Title = "Macroeconomics", Credits = 3, Grade = Grade.C },
42                 new Enrollment { StudentID = 6, CourseID = 1045, Title = "Calculus", Credits = 4 },
43                 new Enrollment { StudentID = 7, CourseID = 3141, Title = "Trigonometry", Credits = 4, Grade = Grade.A }
44             );
45             context.SaveChanges();
46         }
47     }
48 }
49
```

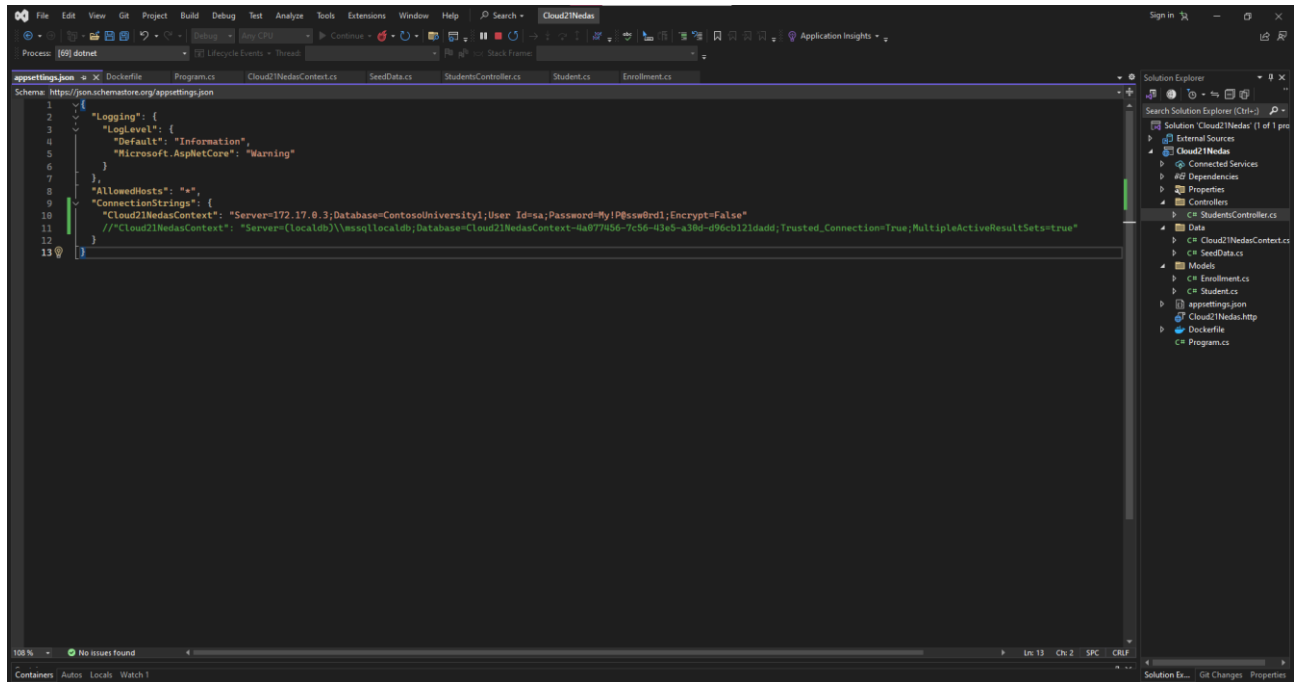
pav. 1 SeedData.cs



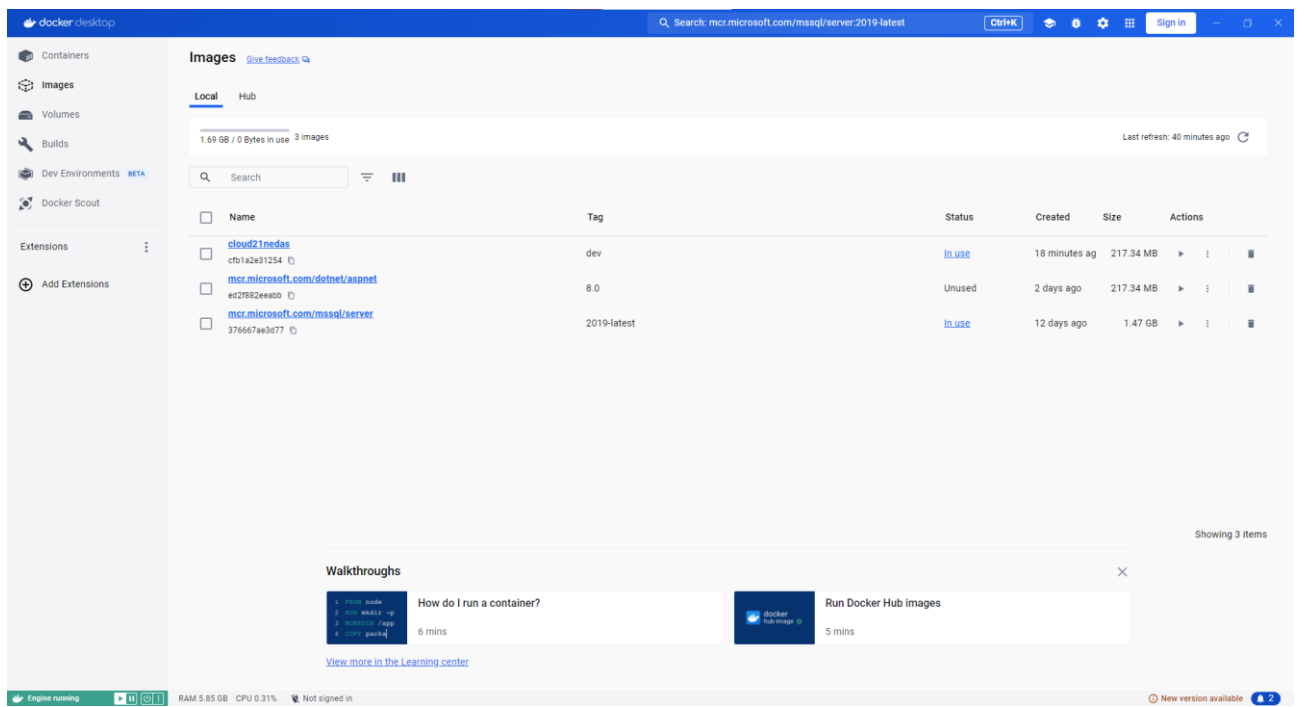
pav. 2 Program.cs



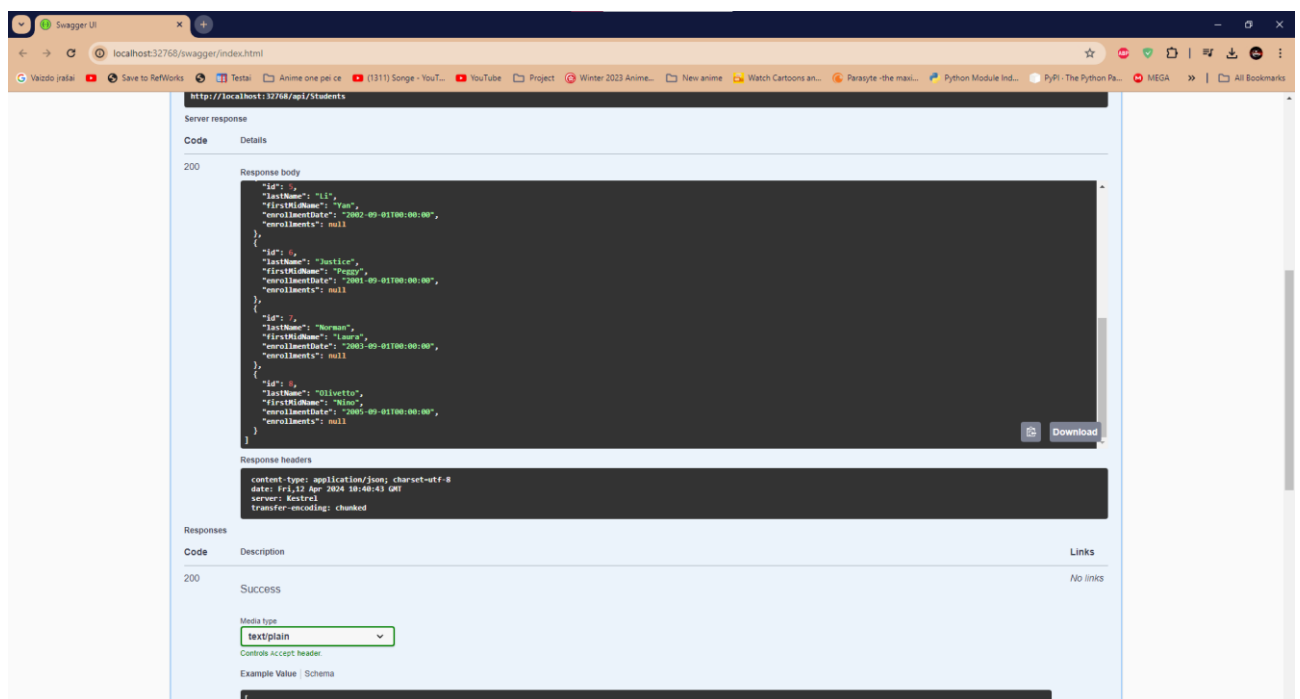
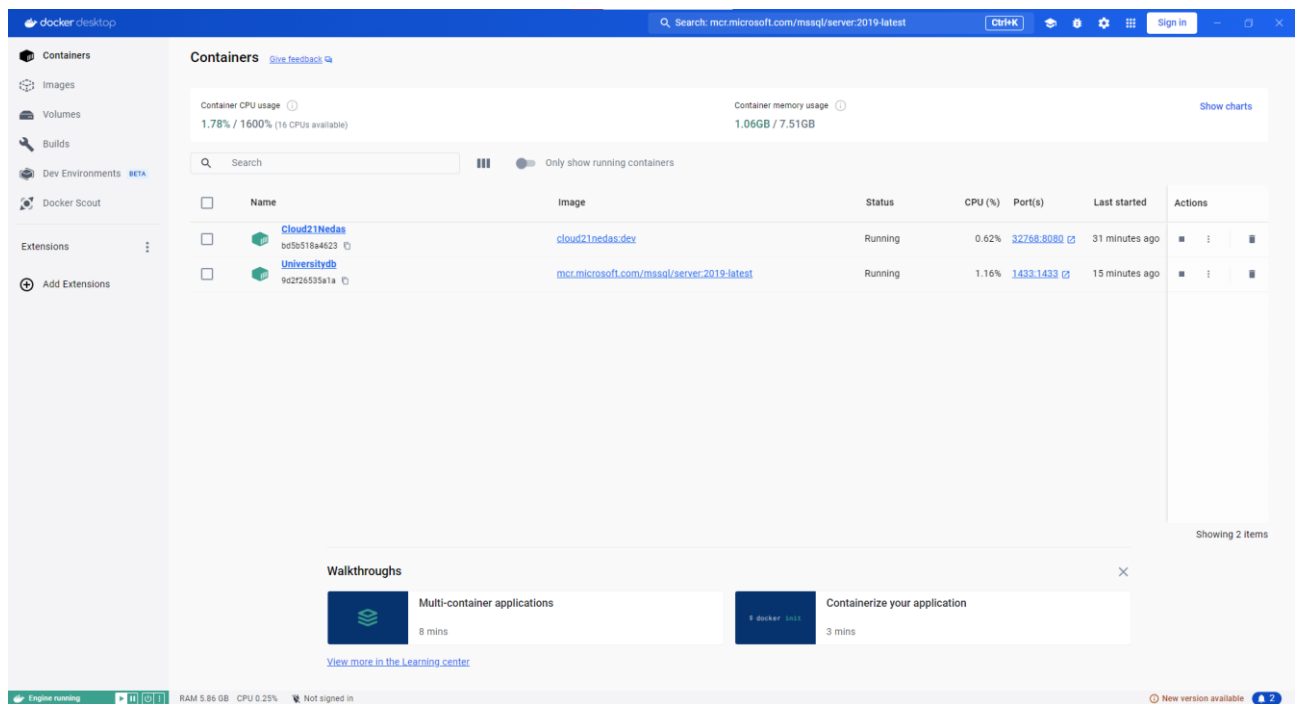
pav. 3 DockerFile



pav. 4 Students controller



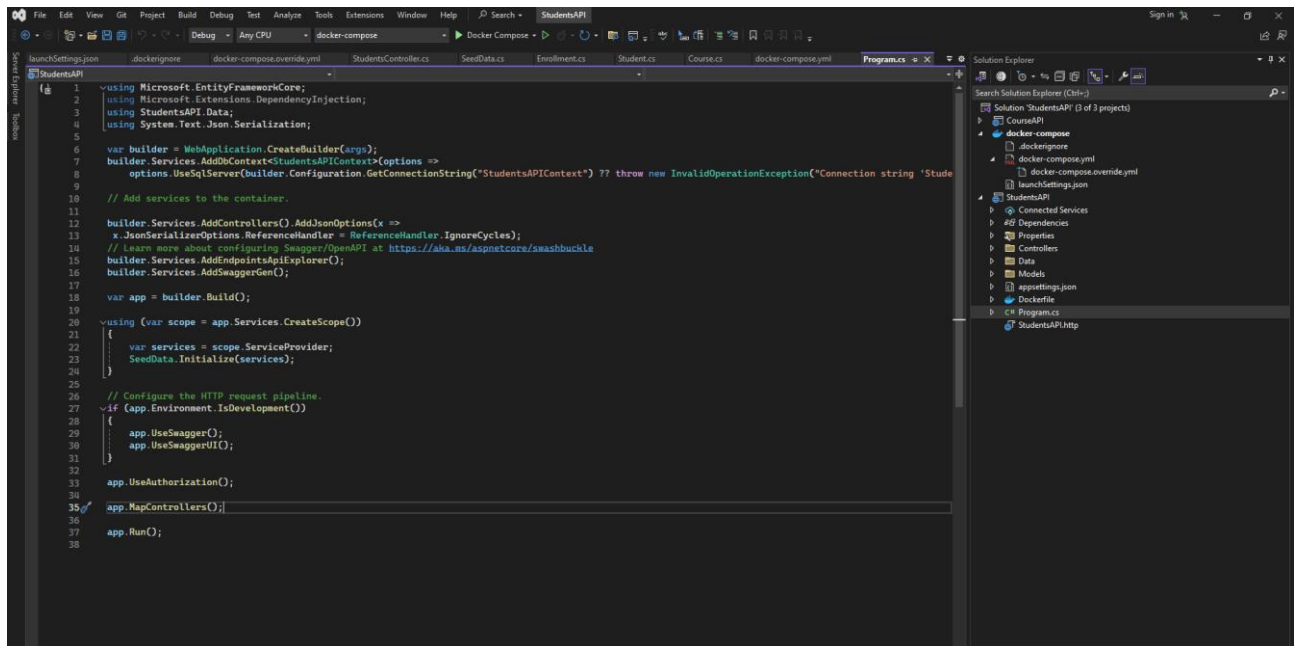
pav. 5 Docker Desktop



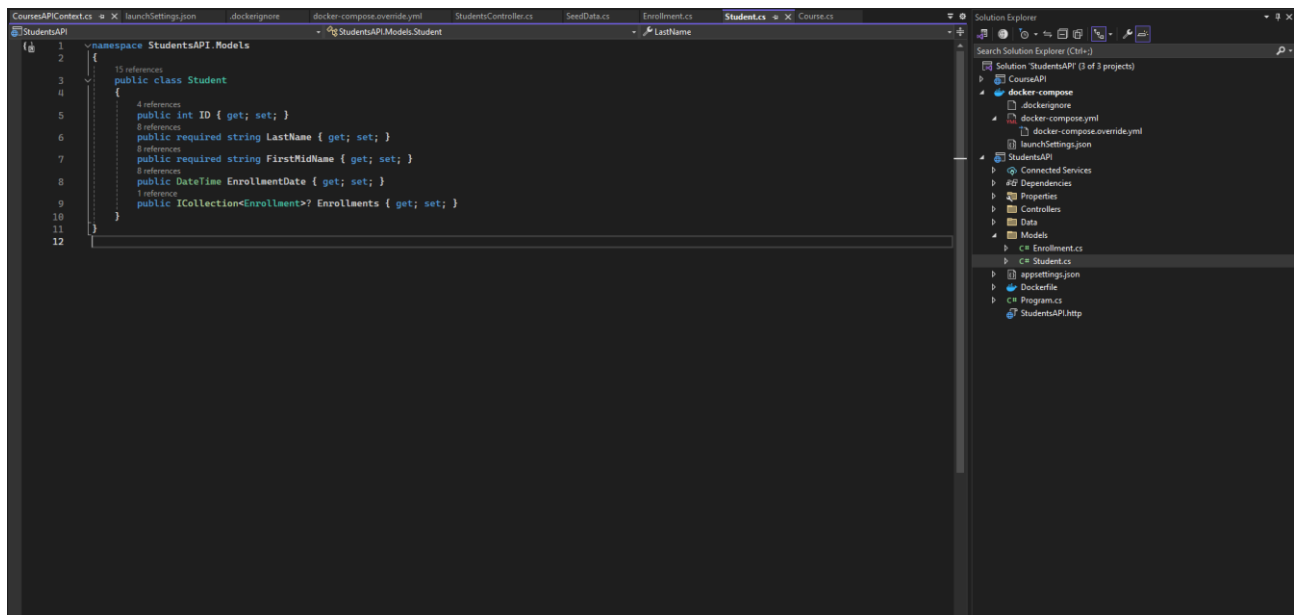
pav. 6 Swagger

Lab 2.2 Dalis

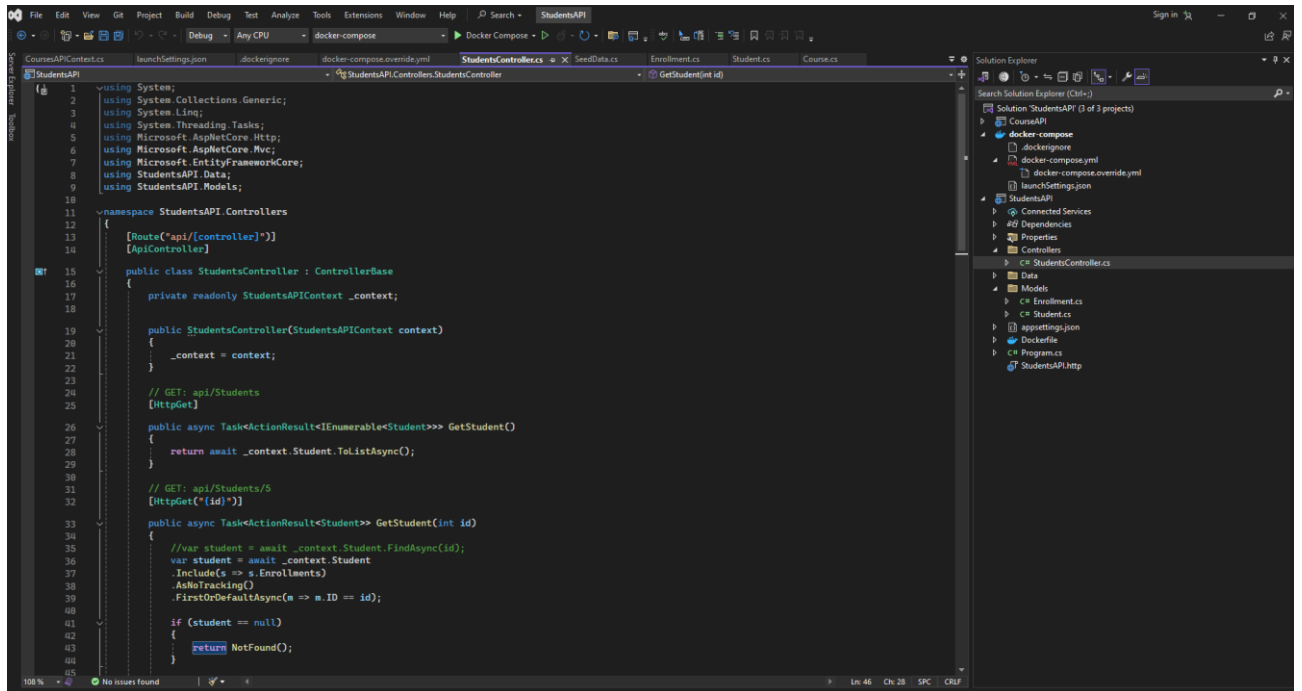
Šioje laboratorinio darbo dalyje, atlikome tą patį kaip ir pirmoje, tik naudojome docker-compose, kad galėtume sujunti abi atskiras API sistemas.



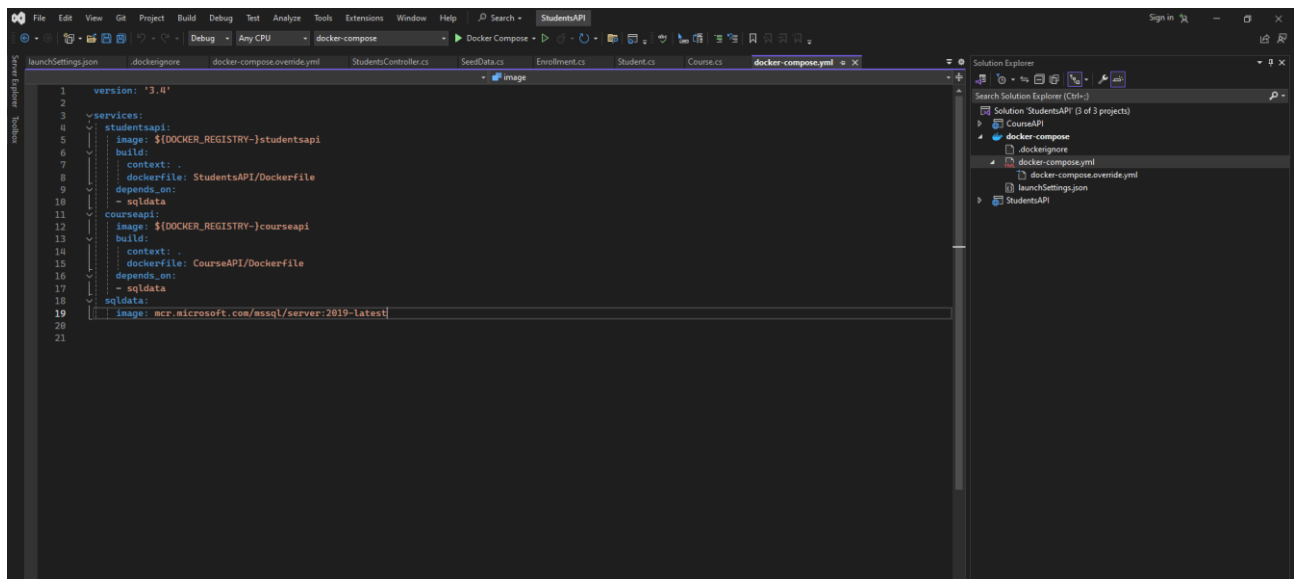
pav. 7 Program.cs



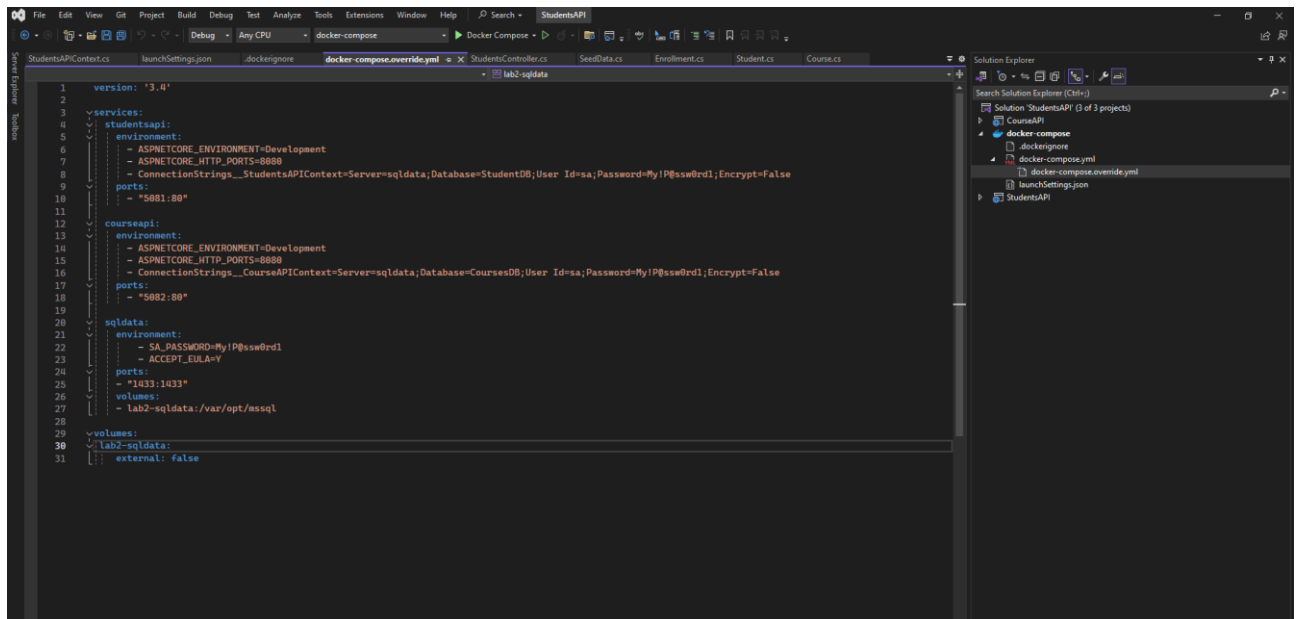
pav. 8 Student.cs



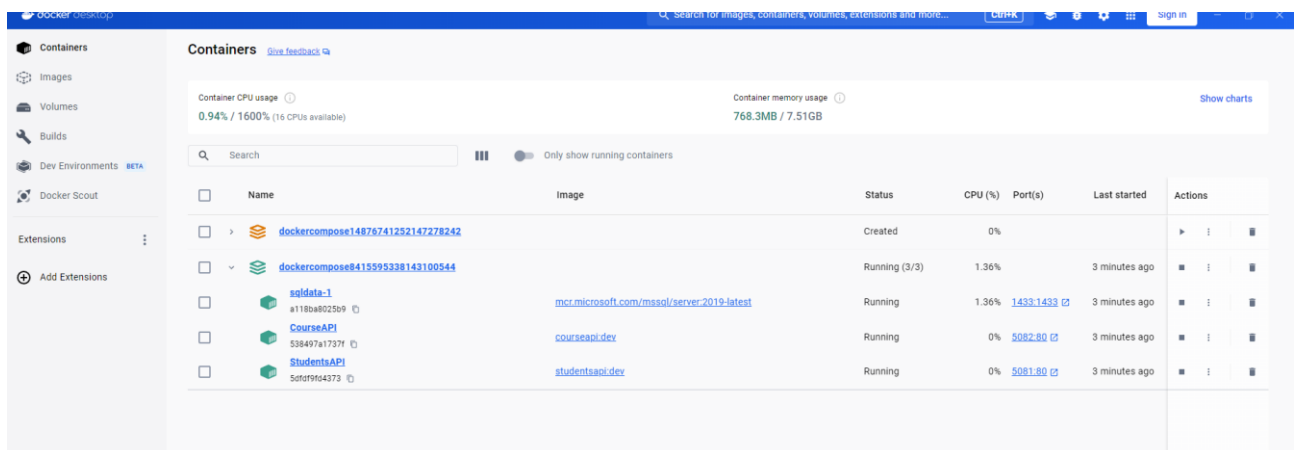
pav. 9 Students Controller



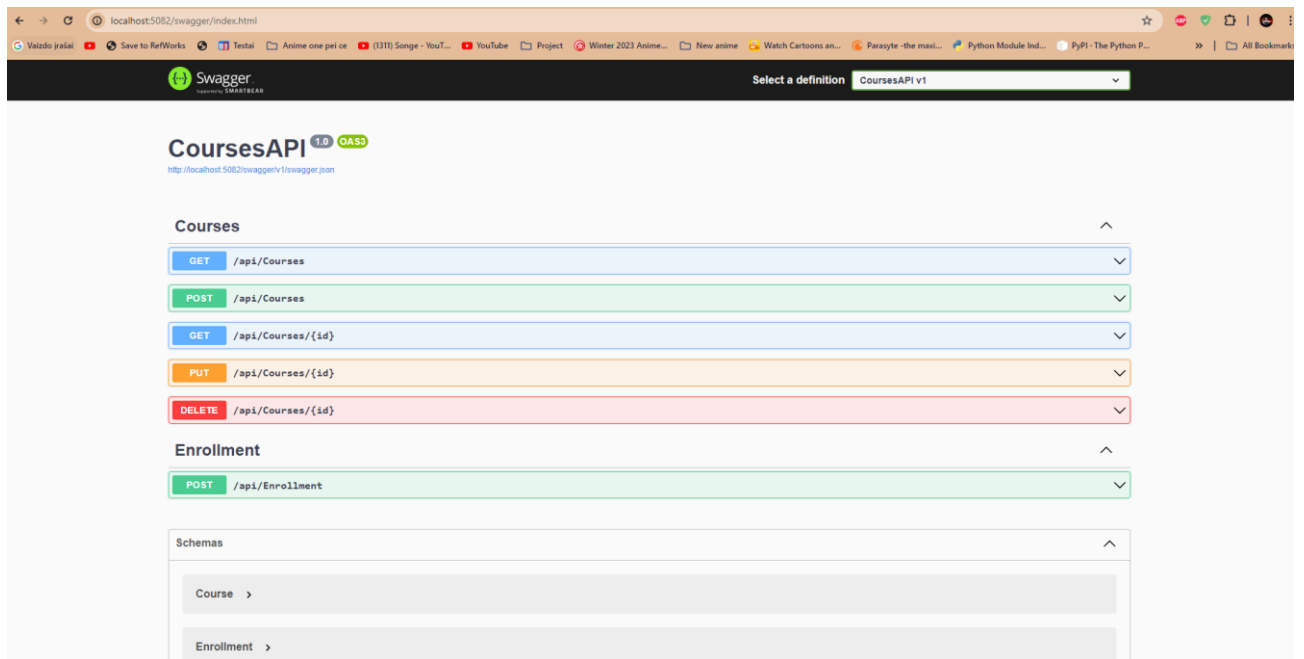
pav. 10 Docker-compose file



pav. 11 Docker-override file



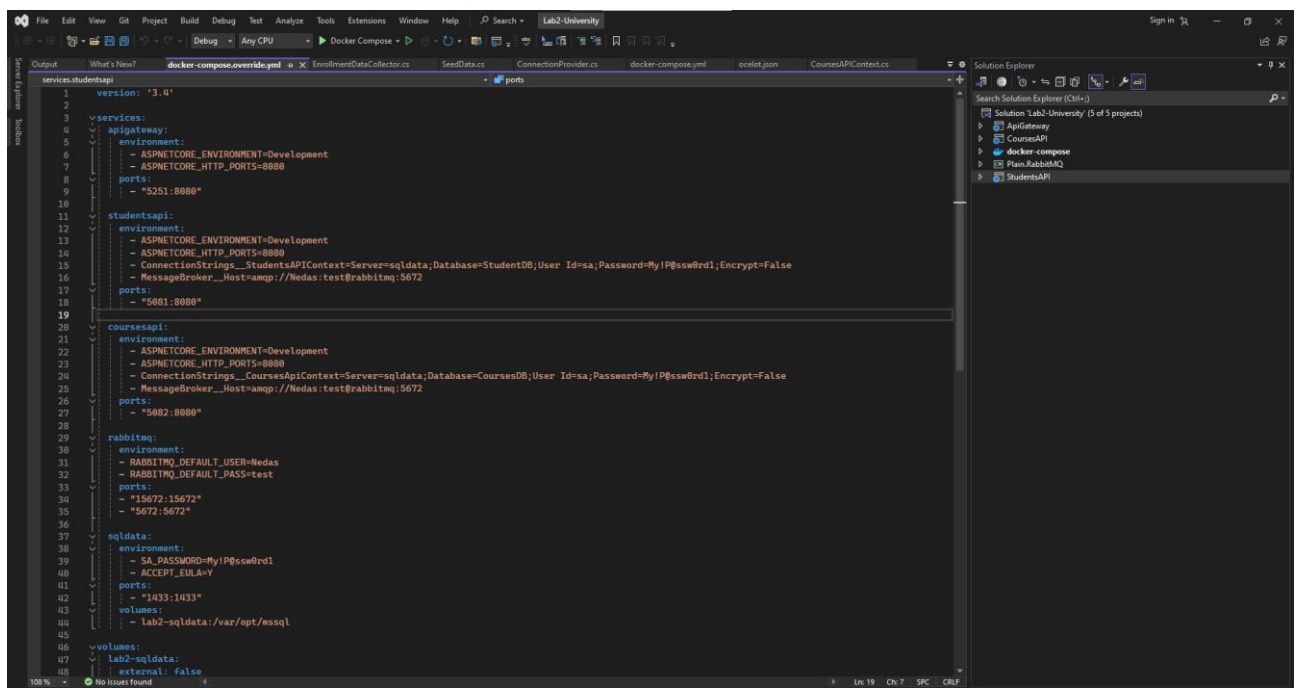
pav. 12 Docker desktop



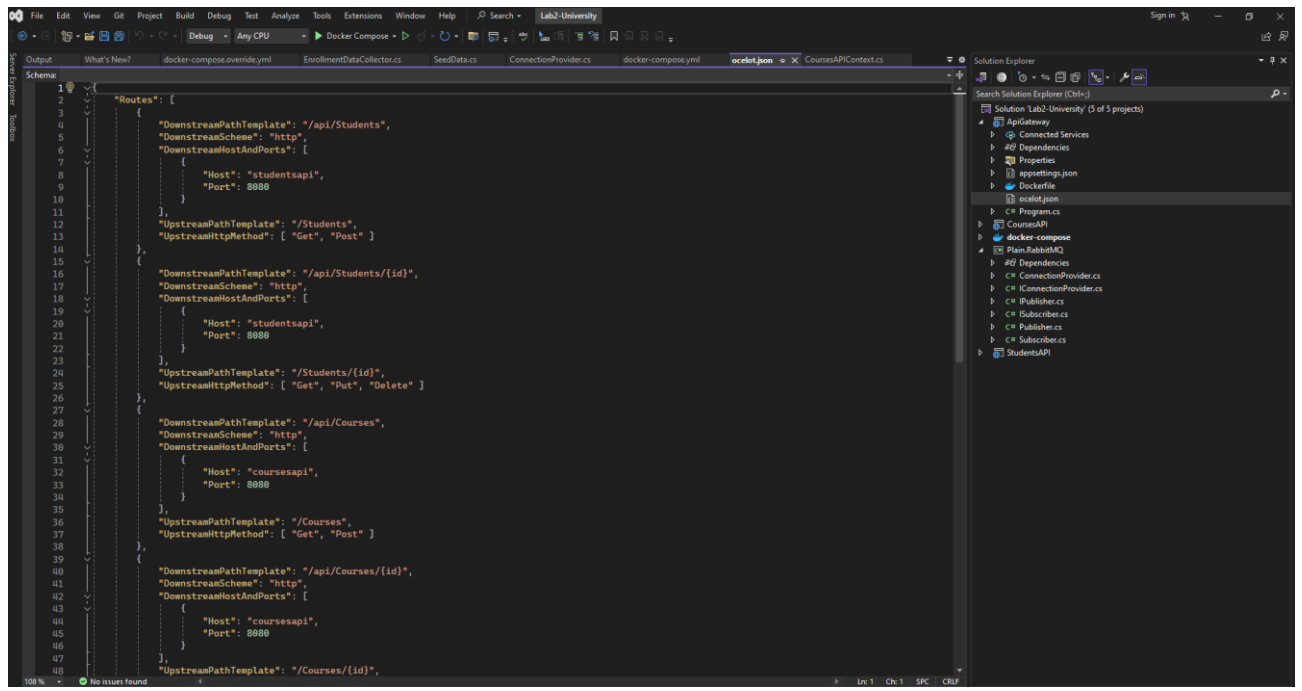
pav. 13 Swagger

Lab 3.3 dalis

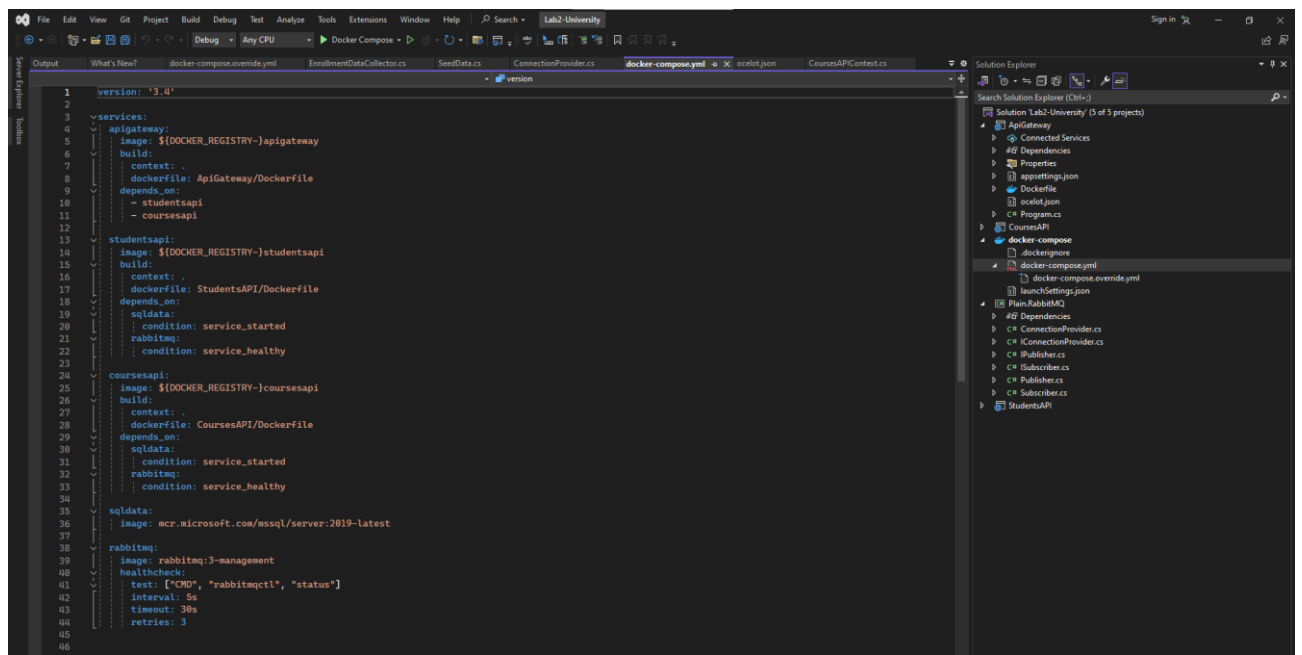
Šioje, laboratorinio darbo dalyje, sudarėme API gateway, kuris leidžia mums valdyti ir sąveikauti su savo mikroservisų architektūros programų sąsajomis.



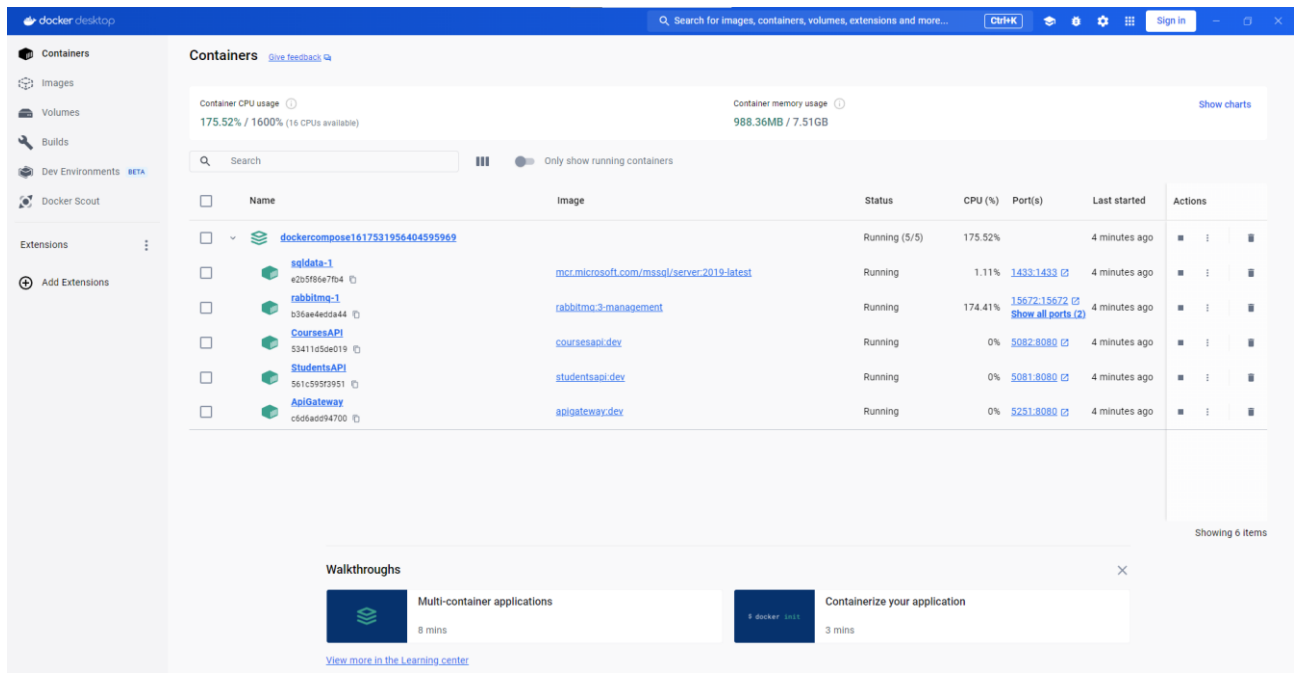
pav. 14 Docker-compose override



pav. 15 Ocelot.json file



pav. 16 Docker-compose file



pav. 17 Docker desktop



pav. 18 Api response