



УНИВЕРЗИТЕТ У НОВОМ САДУ  
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У  
НОВОМ САДУ



---

CLOUD COMPUTING U ELEKTROENERGETSKIM  
SYSTEMIMA

***Osnove Microsoft Windows Azure servisa***  
**-SKRIPTA-**

Novi Sad, 2018

## Vežba 4 – Windows Azure Storage service

U vežbi 4 je prikazan osnovni *API storage* servisa i prikazan je način čuvanja podataka u okviru *Table storage* servisa.

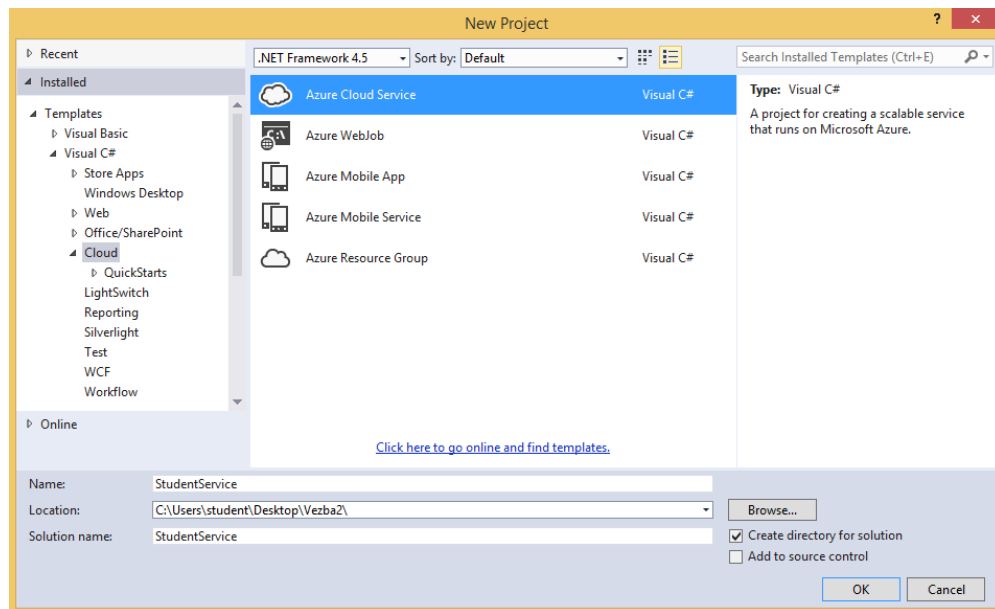
Vežba je izdvojena u tri celine. Prvi deo prikazuje kreiranje *Cloud* projekta i *Worker role* tipa procesa, u drugom delu je implementirana komunikacija sa *table storage* servisom, dok se treći deo odnosi na kreiranje konzolne aplikacije koja koristi usluge servisa.

### 1 Kreiranje *Cloud* projekta koristeći *Visual Studio 2017*

U ovom delu vežbe, napraviće se *Cloud* projekat koristeći šablon koji dolazi uz instalirani *SDK*.

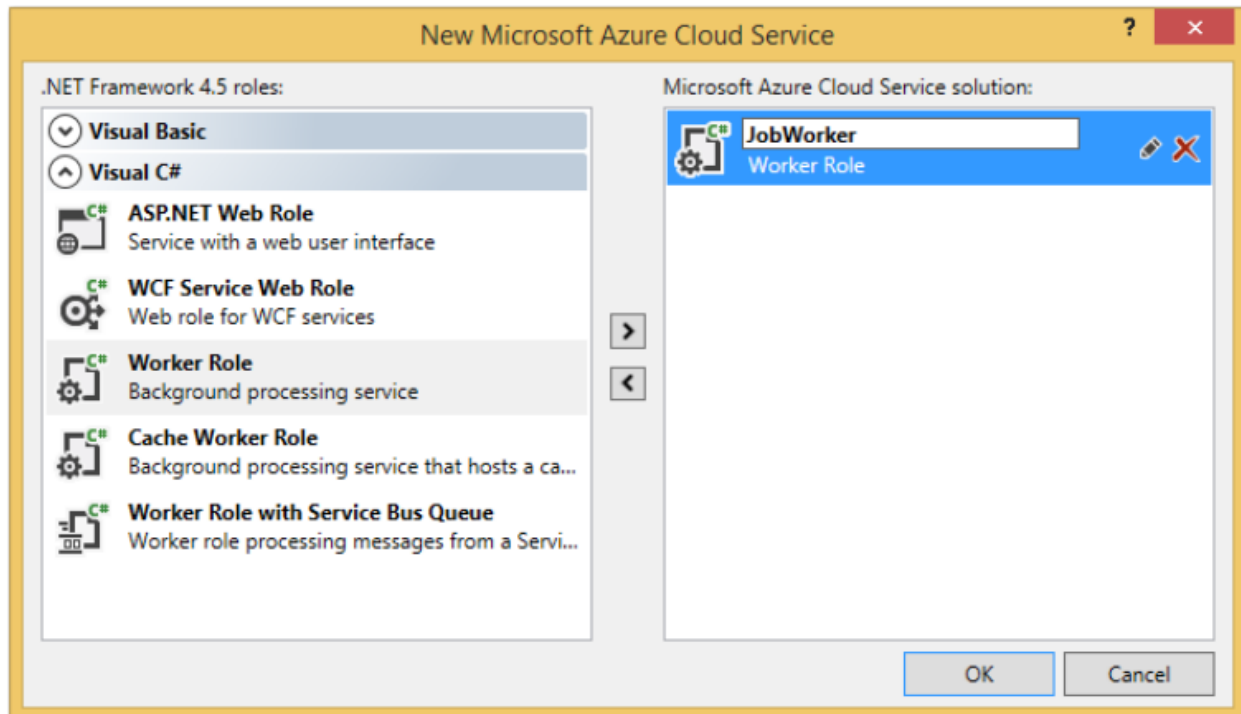
Postupak za kreiranje *cloud* projekta:

- Klikom na *File* meni, odabrati *New/Project*
- U *New Project* dijalogu, proširiti *Visual C#* u listi *Installed Templates* i selektovati *Cloud*, kao što je prikazano na slici 1. Odabrati šablon *Windows Azure Cloud Service*. Uneti ime projekta *StudentsService*. Kliknuti na *OK*.



Slika 1 Kreiranje *cloud* projekta

U *New Windows Azure Project* dijalogu, u okviru *Roles* panela, otvoriti tab *Visual C#*. Selektovati *Worker Role* i dati mu naziv *JobWorker* (Slika 2).



Slika 2 Dodavanje worker role

Kreirati *Class Library* projekat koji će sadržati interfejs za WCF servise. Nazvati projekat “*Contracts*”. Dodati interfejs *IStudent* u biblioteku. Definicija interfejsa je data u listingu 1.

Listing 1 – *IStudent* interfejs

```
[ServiceContract]
public interface IStudent
{
    [OperationContract]
    List<Student> RetrieveAllStudents();

    [OperationContract]
    void AddStudent(string indexNo, string name, string lastName);
}
```

U okviru projekta *JobWorker* kreirati WCF Server kao posebnu klasu (*StudentServer*) u okviru koje će biti metode za otvaranje i zatvaranje konekcije servisa. Koristiti *NetTcpBinding*, a za dobijanje krajnje IP adrese trenutnog procesa, iskoristiti klasu *RoleEnvironment*.

U implementaciji *OnStart()* metode Worker role pozvati kreiranje i otvaranje konekcije ovog servisa.

Kreirati klasu *StudentServerProvider* u novom fajlu i u njoj implementirati interfejs *IStudent*.

## 2 Kreiranje modela podataka za entitete u *Table storage*

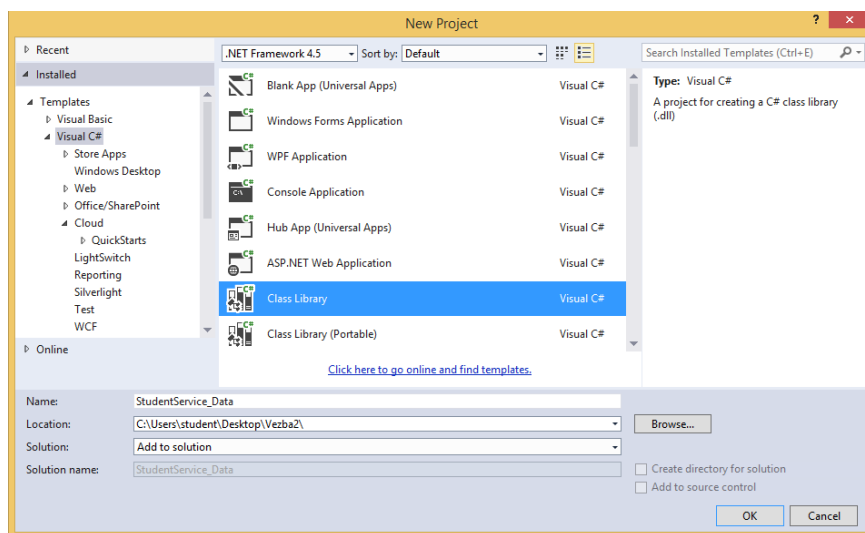
*StudentsService* aplikacija je zamišljena tako da prikazuje listu studenata i da ima mogućnost dodavanja novog studenta. Podaci o studentu bi trebali da budu: broj indeksa, ime i prezime.

U ovom delu vežbe 4, modeluje se entitet koji će *StudentsService* aplikacija koristiti i kreira se kontekstna klasa koja koristi *Microsoft.WindowsAzure.Storage.Table* za pristup table storage servisu. Za model entiteta je potrebno da bude ponovno upotrebljiv kroz više *WindowsAzure* servisa. Iz razloga ponovne upotrebljivosti, potrebno je kreirati biblioteku koja će sadržati model podataka i klase za rukovanje datim modelom.

### 2.1 Dodavanje novog projekta – biblioteke

- Kreirati novu biblioteku za rad sa *table storage* servisom. Odabrati *file* meni, potom *Add/New Project*.

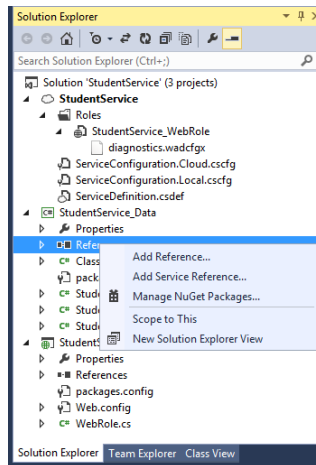
- U *Add New Project* dijalogu, proširiti Visual C# pa potom odabrati *Windows* kategoriju. Odabrati *ClassLibrary* kao šablon za projekat. Postaviti naziv projekta na *StudentsService\_Data*, i birati opciju *Solution: Add to solution* i kliknuti OK. Videti sliku 3.



Slika 3 Kreiranje biblioteke

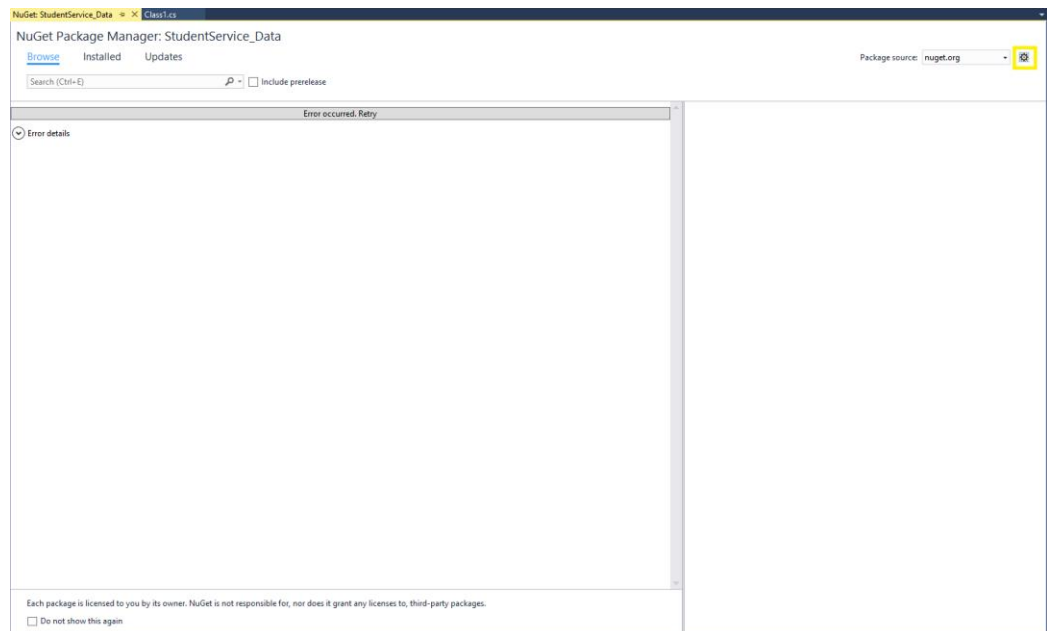
- U okviru projekta *StudentService\_Data* treba dodati *Microsoft.WindowsAzure.Storage* referencu na sledeći način (ako ne postoji):

- Otvoriti “Manage NuGet Packages” prozor, desni klik na References i kliknuti “Manage NuGet Packages” iz kontekst menija.



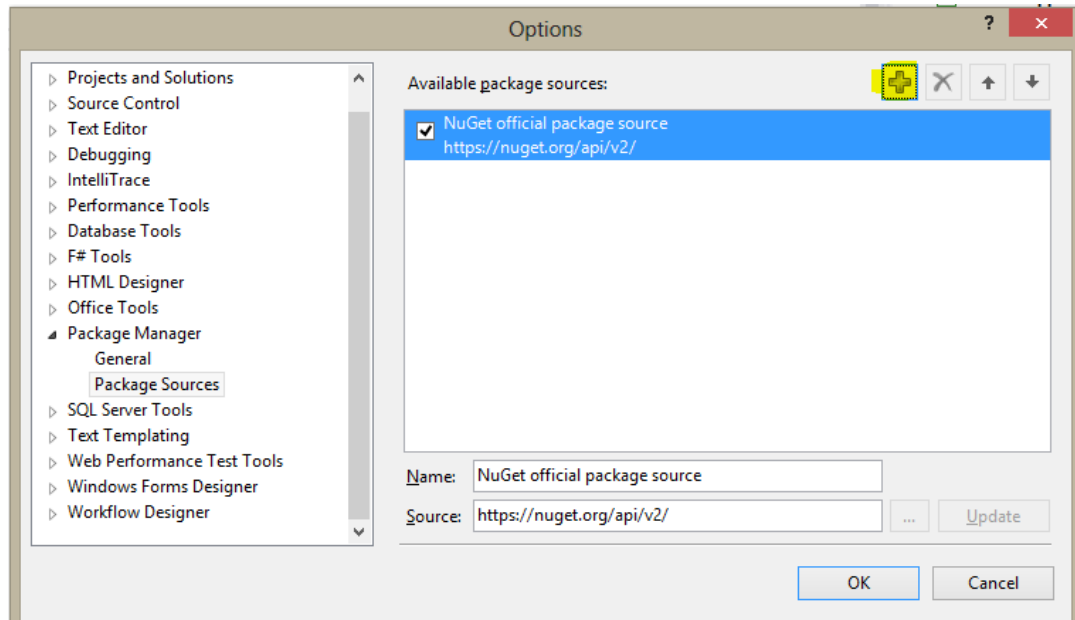
Slika 4 Dodavanje NuGet-a

- Klik na “Settings”



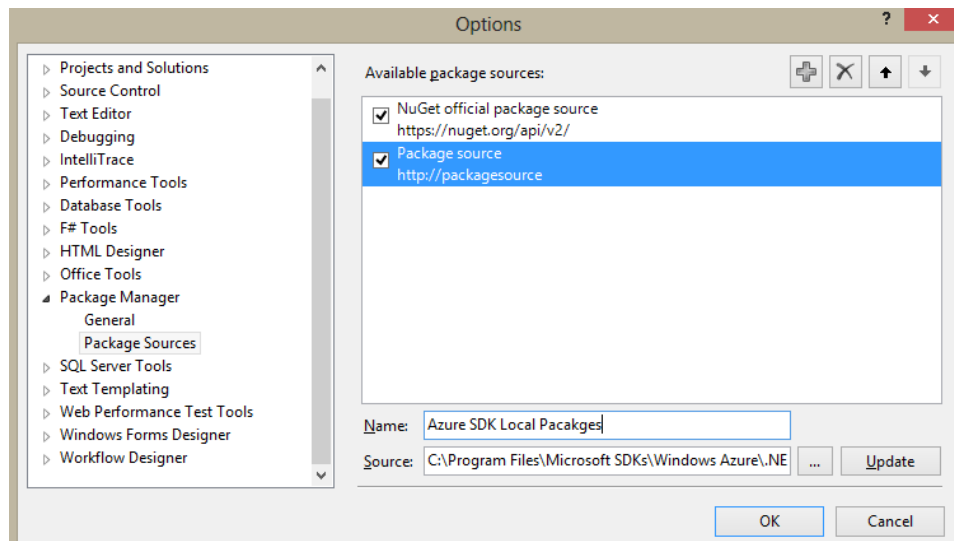
Slika 5 NuGet prozor

- Klik na “+” dugme na vrhu prozora:



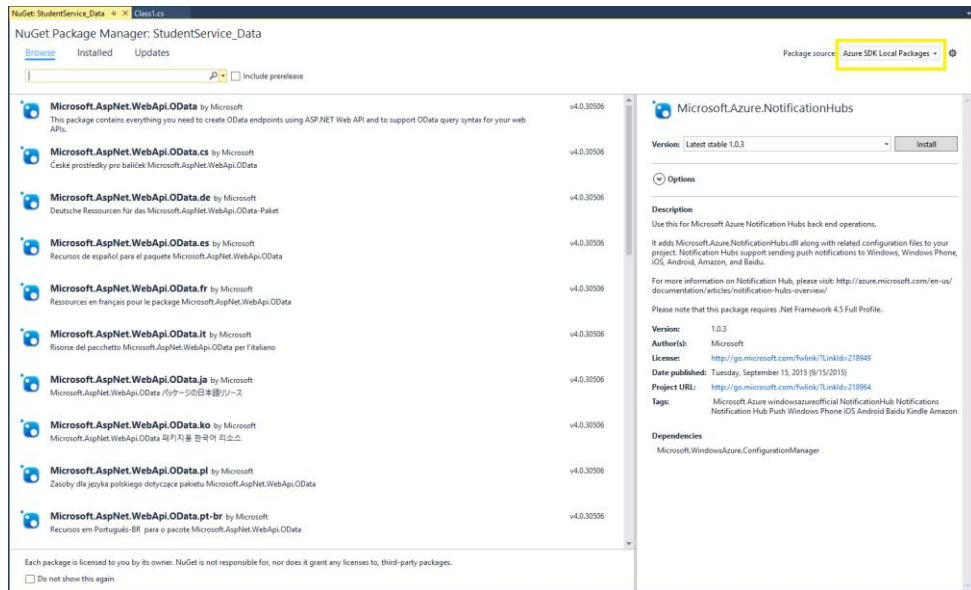
*Slika 6 NuGet opcije*

- Dodati ime i lokaciju ('C:\Program Files\Microsoft SDKs\Azure\.NET SDK\v2.9\packages')



*Slika 7 Dodavanje opcija*

- Primetićete da je lokalni folder dodat i možete ga izabrati desno u gornjem uglu instalirati package Windows Azure storage (Slika 8).



Slika 8 Instaliranje paketa

- U Search ukucati „WindowsAzure.Storage“ i instalirati *WindowsAzure.Storage* paket

## 2.2 Kreiranje klase entiteta

Da bi se entiteti čuvali u tabelama *Table Storage* servisa, prvo se mora izmodelovati entitet studenta.

- Desni klik na *StudentsService\_Data* u *SolutionExplorer*, potom *Add/Class*. U *Add New Item* dijalogu, nazvati klasu *Student* i kliknuti *Add*.

- Prvo klasu *Student* treba modifikovati tako da bude *public* i da nasleđuje *TableEntity*. Da bi klasa *TableEntity* bila vidljiva potrebno je dodati referencu *Microsoft.WindowsAzure.Storage.Table*.

- Napraviti konstruktor klase koji će primati broj indeksa kao jedini parametar. U konstruktoru inicijalizovati *PartitionKey* literalom *Student*, a *RowKey* brojem indeksa.

- Dodati polja za ime i prezime. Sačuvati klasu *Student*. U listingu 1 je dat konačan izgled klase *Student*.

Listing 1 – Student.cs

```
using System;
using Microsoft.WindowsAzure.Storage.Table;

namespace StudentService_Data
{
    public class Student : TableEntity
    {
        public String Name { get; set; }
    }
}
```

```

    public String LastName { get; set; }

    public Student(String indexNo)
    {
        PartitionKey = "Student";
        RowKey = indexNo;
    }

    public Student() { }
}

```

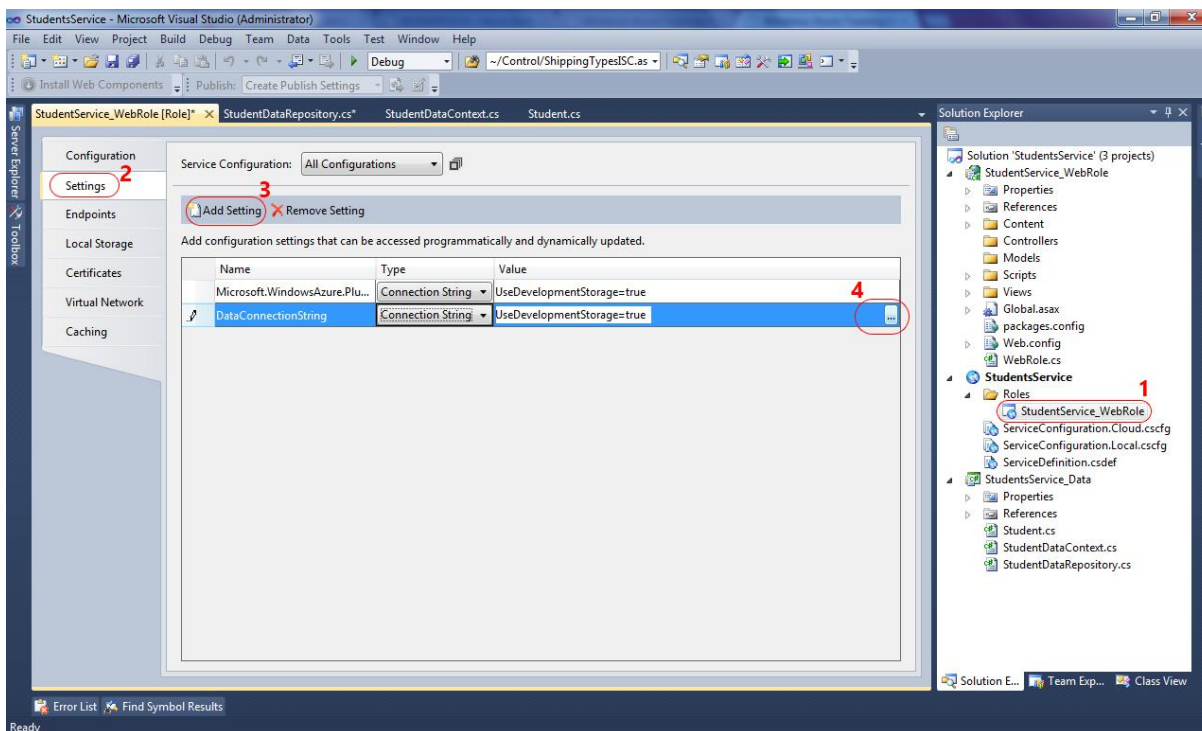
## 2.3 Kreiranje konteksta podataka

Potrebno je dodati kontekstnu klasu koja služi za pristup „*StudentTable*“ tabeli, manipulaciju entitetima tako što izvršava čitanje, upis i modifikaciju entiteta.

- Potrebno je u *StudentsService\_Data* projekat dodati klasu koja se bavi manipulacijom entiteta. Nazvati novu klasu *StudentDataRepository*.

- Modifikovati *Repository* klasu da bude *public*. Potom dodati dva polja s modifikatorima *private* za *CloudTable* i za *CloudStorageAccount*.

- *CloudStorageAccount* klasa se odnosi na podatke vezane za *WindowsAzure* nalog kao što su kredencijali za pristup servisima, postavljanje konfiguracije i metode za kreiranje *storage* servisa. Instalirani *SDK* nudi emulaciju *Storage* servisa, tako da je potrebno konfigurisati servis da koristi lokalni *storage* nalog umesto servisa koji se dobijaju uz pretplatu.



Slika 9 Konfiguracija za konekciju na Storage servis



- Da bi se konfigurisao *ConnectionString* kao na slici 9, potrebno je prvo otvoriti konfiguraciju za *worker role* procese. Konfiguracija se nalazi u okviru *cloud* projekta, u direktorijumu *Roles*. Dvoklikom na *JobWorker* se otvara konfiguracija. Kliknuti na *Settings*, i u okviru ovog prozora kliknuti na *AddSettings*. Pod *Name* kolonom upisati *DataConnectionString*, i selektovati *Type ConnectionString*. U okviru *Windows Azure web* portala, može se pronaći *connection string* namenjen pristupu servisima i prekopirati. U ovoj vežbi će se koristiti emulirani *storage* servis. Kliknuti na dugme sa tri tačke i otkaćiti opciju *Windows Azure storage emulator*. Kliknuti na OK i u polju *Value* bi trebalo da se pojavi *UseDevelopmentStorage=true*.

- Kada se *ConnectionString* koji će koristiti emulirani *storage* konfiguriše, potrebno je refencirati se na emulirani servis. Dodati podrazumevani konstruktor u klasi *StudentDataRepository*. U okviru podrazumevanog konstruktora, inicijalizovati *StudentDataContext* koristeći *CloudStorageAccount* i kreirati sve neophodne tabele. *CloudStorageAccount* treba da koristi emulirano okruženje što je određeno konfiguracijom.

- U *Repository* klasi, neophodno je još dodati metode koje čine smisao same *Repository* klase. U ovom slučaju, to su metode za listanje svih studenata i za dodavanje studenta. Kasnije će biti potrebe za proširenjem funkcionalnosti kao što je dodavanje metode za postavljanje novog linka ka *thumbnail* slici i slično. Implementirana *Repository* klasa data je u listingu 4.

- Dodati referencu na *Microsoft.WindowsAzure.Configuration* (ako već nije dodata) kako bi se mogao uključiti *Microsoft.WindowsAzure namespace* da bi klasa *CloudConfigurationManager* bila vidljiva. *CloudConfigurationManager* klasa omogućava čitanje vrednosti *XML* konfiguracionih datoteka.

#### Listing 4 – StudentRepository class

```
using Microsoft.WindowsAzure;
using Microsoft.WindowsAzure.Storage;
using Microsoft.WindowsAzure.Storage.Table;
using System;
using System.Linq;

namespace StudentService_Data
{
    public class StudentDataRepository
    {
        private CloudStorageAccount _storageAccount;
        private CloudTable _table;

        public StudentDataRepository()
        {
            _storageAccount =
CloudStorageAccount.Parse(CloudConfigurationManager.GetSetting("DataConnectionString"));
            CloudTableClient tableClient = new CloudTableClient(new
Uri(_storageAccount.TableEndpoint.AbsoluteUri), _storageAccount.Credentials);
            _table = tableClient.GetTableReference("StudentTable");
            _table.CreateIfNotExists();
        }
    }
}
```

```

public IQueryable<Student> RetrieveAllStudents()
{
    var results = from g in _table.CreateQuery<Student>()
                   where g.PartitionKey == "Student"
                   select g;

    return results;
}

public void AddStudent(Student newStudent)
{
    // Samostalni rad: izmestiti tableName u konfiguraciju servisa.
    TableOperation insertOperation = TableOperation.Insert(newStudent);
    _table.Execute(insertOperation);
}
}

```

- U konstruktoru *StudentDataRepository* klase, vrši se inicijalizacija objekata *CloudStorageAccount* i *CloudTable*. *CloudStorageAccount* se inicijalizuje koristeći konfigurabilni *ConnectionString*. Na osnovu objekta *CloudStorageAccount*, inicijalizuje se *CloudTable*, koja će koristiti prosleđeni *storage* nalog.

- *RetrieveAllStudents* metoda preuzima sve studente iz tabele “*Student*”. Koristi se *LINQ* upit za preuzimanje podataka.

- *AddStudent* metoda dodaje novog studenta u tabelu. Koristi se *TableOperation* i metoda *Insert*.

U ovom delu vežbi je napravljena biblioteka koja obezbeđuje rad sa *Table storage* servisom. Potrebno je kreirati konzolnu aplikaciju pomoću koje će se omogućiti unos novog studenta kao i prikaz imena i prezimena svih studenata.