УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ

CLOUD ZASNOVANI SMARTGRID SISTEMI

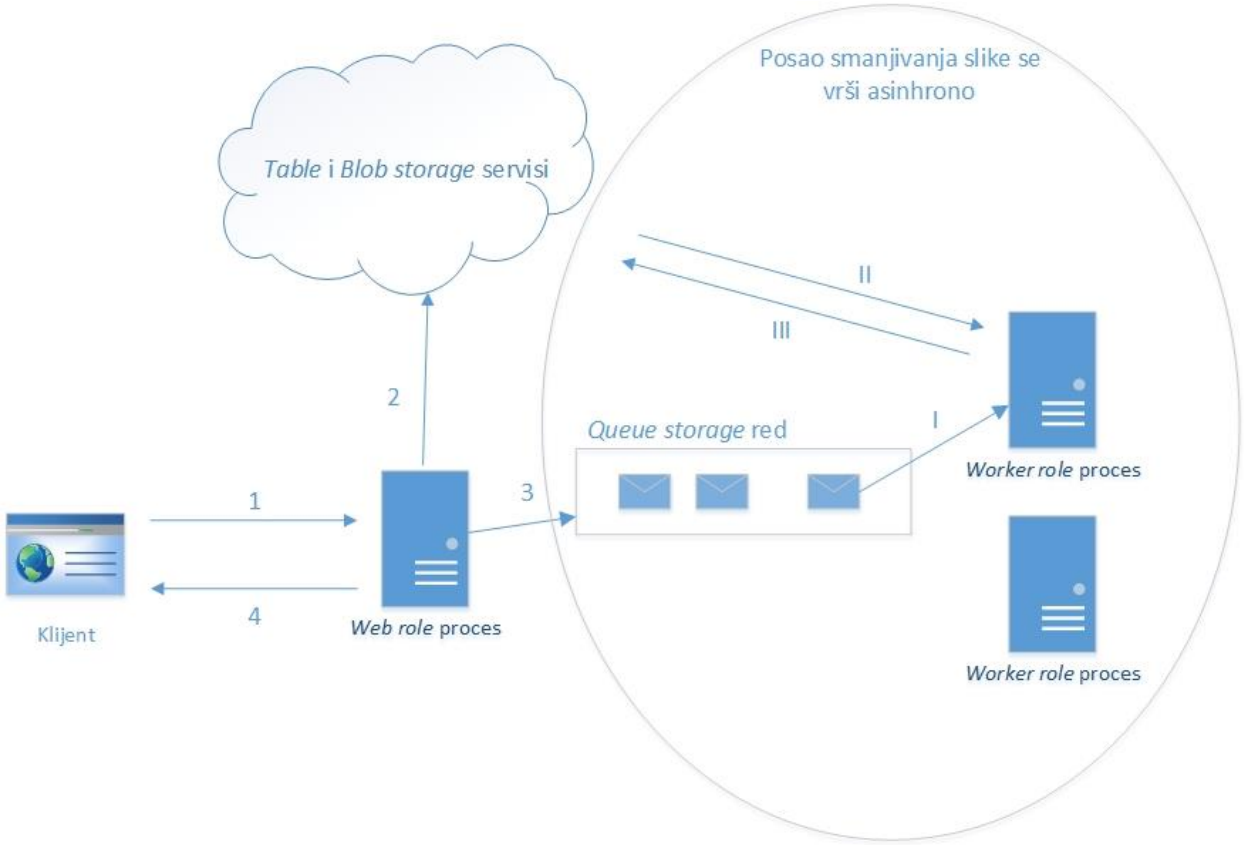
Osnove *Microsoft WindowsAzure* servisa

-SKRIPTA-

Novi Sad, 2018

Vežba 7 – Asinhrona komunikacija između *web role* i *worker role* procesa

U vežbi 5 se rešava zadatak koji koristi asinhronu komunikaciju između različitih procesa. Rešenje se implementira kao dodatak projektu koji je korišćen u vežbi 3. Zadatak je izdeljen u tri celine – A, B i C. Preporuka je da se zadaci rešavaju datim redosledom. Na slici 1 je prikazana arhitektura krajnjeg rešenja zadatka.



Slika 1 – Zadatak vežbe pet.

U zaokruženom delu slike 1, prikazan je zadatak koji se implementira u vežbi 5. U levom delu slike je prikazano rešenje zadatka sa vežbi 3, dodavanje i listanje studenata. Potrebno je dodati *Windows Azure queue storage* red, čije će poruke nositi informacije o broju indeksa dodatih studenata. *Worker role* procesi preuzimaju poruke i obrađuju ih. Na osnovu broja indeksa se za datog studenta preuzme slika, izvrši smanjivanje slike i njeno postavljanje na *BLOB storage* servis.

Zadatak A:

U *cloud* projekat *StudentsService* dodati *worker role* projekat po nazivu *ImageConverter_WorkerRole*. *Worker role* proces treba da na svakih pet sekundi preuzima poruku iz reda. Ukoliko poruke nema u redu, ispisati u *compute* emulatoru "Trenutno ne postoji poruka u redu." Ukoliko se poruka pronađe, ispisati njen sadrzaj na *compute* emulatoru, potom je izbrisati.

Nakon implementacije zadatka, testirati rešenje tako što se poruke dodaju u red putem *Visual Studio Server Explorer* komponente.

Smernice za rešavanje zadatka A:

Worker role projekat se može dodati desnim klikom nad *Roles* direktorijumu u okviru *StudentsService cloud* projekta.

Napisati *QueueHelper* klasu čija je uloga da vrati referencu na *CloudQueue* objekat. *QueueHelper* klasu implementirati u projektu *StudentService_Data* i napisati statičku metodu `CloudQueue GetQueueReference(String queueName)`. Metoda kreira red ukoliko vec ne postoji. Povratna vrednost metode je objekat koji predstavlja red po imenu koji se prosledi kao parametar „queueName“.

Za ispis u *compute* emulator koristiti klasu *Trace* koja se nalazi u *System.Diagnostics namespace*.

Dodatni zadatak A1:

Zakomentarisati brisanje poruke iz reda nakon njenog očitavanja. Dodati jednu poruku u red i protumačiti ponašanje rešenja.

Zadatak B:

Smatrati da se u poruci nalazi broj indeksa studenta. Na primljenu poruku, sprovesti kreiranje umanjene slike (engl. *thumbnail*) na osnovu postojeće slike studenta. Dodati metodu *ResizeImage* u *WorkerRole* klasu čiji je opis dat u listingu 1.

Listing 1: Metoda ResizeImage

```
/// <summary>
/// Metoda pronalazi studenta po prosledjenom broju indeksa. Ukoliko student ne postoji,
ispisuje poruku o tome u compute emulatoru.
/// Ukoliko student postoji, preuzima sliku, konvertuje je u manju sliku i vrsi upload
manje slike. Url do manje slike se cuva u okviru atributa ThumbnailUrl student entiteta.
/// </summary>
/// <param name="indexNo">broj indeksa studenta</param>
public void ResizeImage(String indexNo)
```

Za potrebe konverzije slike može se koristiti klasa data u listingu 2.

Listing 2: Klasa za kreiranje slike manjih dimenzija.

```
class ImageConvertes
{
	public static Image ConvertImage(Image img)
	{
		return (Image)(new Bitmap(img, new Size(20, 20)));
	}
}
```

Uvek se kreira *Bitmap* tip slike ukoliko se koristi klasa iz listinga 2. S toga je potrebno pri postavljanju Blob sadržaja, postaviti polje *content type* na vrednost “image/bmp”.

Smernice za rešavanje zadatka B:

Blob storage metode očekuju *stream* objekte u slučaju postavke ili preuzimanja sadržaja. Kako se u ovoj vežbi sa *blob* sadržajem radi isključivo u radnoj memoriji, pogodno je koristiti *MemoryStream* klasu. U listingu 3 je prikazan način upotrebe *MemoryStream* klase.

Listing 3 – *Memory stream* za rad sa *blob storage* servisom

```
using (MemoryStream memoryStream = new MemoryStream())
{
	// pisanje iz i upis u memoryStream iskoristiti u using bloku

	// VAŽNA napomena: nakon čuvanja sadržaja u memory stream bloku, neophodno je vratiti pokazivač memorije na početno stanje ukoliko se posle treba iščitavati sadržaj memory stream objekta
	memoryStream.Position = 0;
}
```

Dodatni zadatak B1:

Detekcija zarazne poruke i njeno uklanjanje. Ukoliko prosleđena poruka ne sadrži broj indeksa, dozvoliti njeno procesuiranje maksimalno tri puta – ovakva poruka se smatra zaraznom jer nikada ne može biti korektno procesuirana.

Zadatak C:

Nakon dodavanja novog studenta, generisati poruku čija će vrednost biti broj indeksa studenta. Testirati rešenje tako što će se nakon procesuirane poruke od strane *worker role* procesa, uraditi osvežavanje stranice u *browser* klijentu.

Dodatni zadatak C1:

Omogućiti da se poruka generiše 30 sekundi od dodavanja studenta. Ukoliko se u međuvremenu student obriše, poruka će biti “zarazna”.