Administracija Azure funkcija

Administracija sistema

Student: Danilo Veljović, 1120 Profesor: Dr. Milorad Tošić

Elektronski fakultet
Univerzitet u Nišu

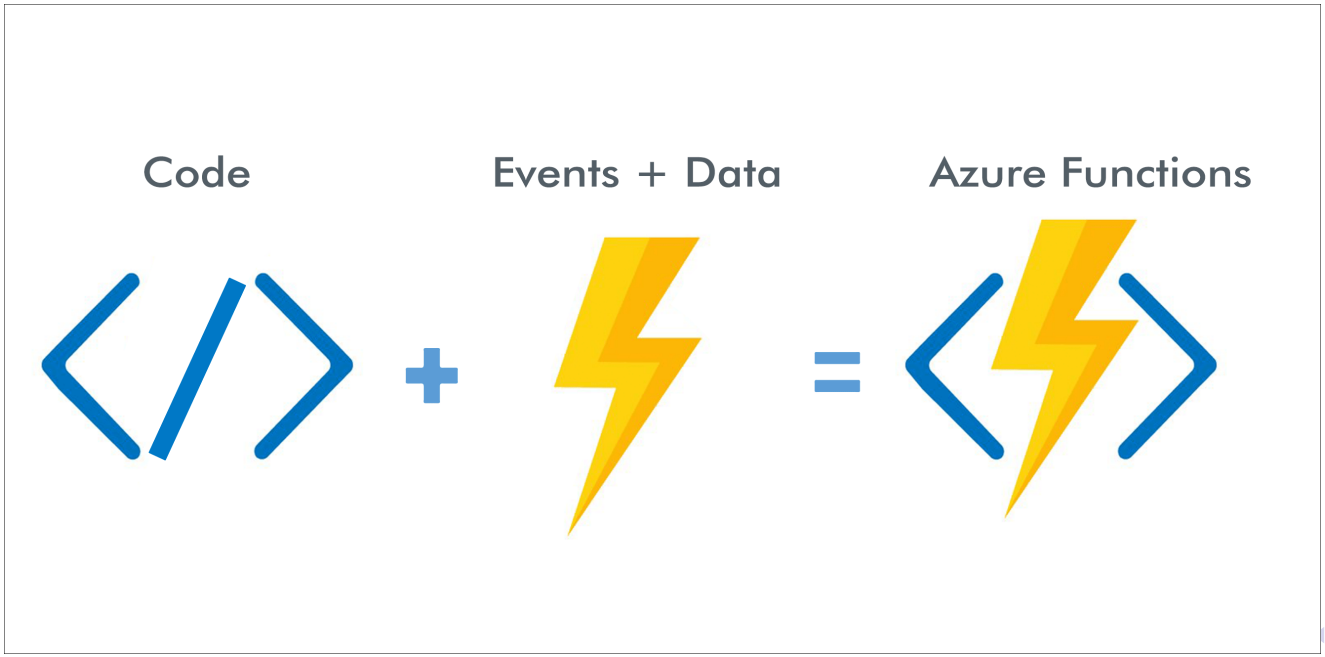
Sadržaj

Sadržaj

Uvod: Šta su Azure funkcije?

Web aplikacije ili sistemi aplikacija se često projektuju tako da reaguju na neke dogadjaje. Bilo da je taj dogadjaj API poziv, neka promena u bazi, obrada IoT tokova podataka ili reagovanje na poruke koje dolaze nekim message brokerima, svaka aplikacija ima kod koji joj omogućava da obradi ove dogadjaje.

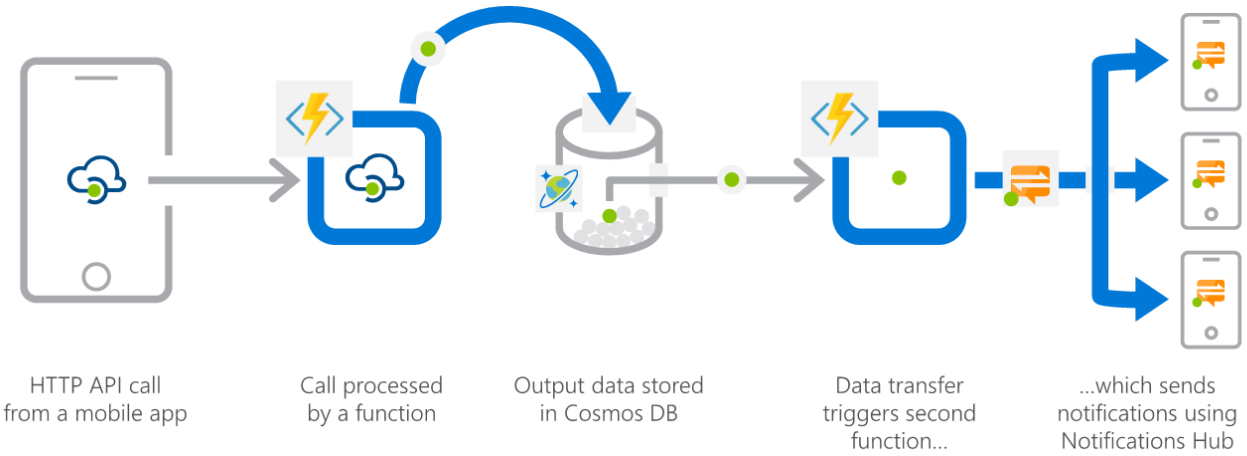
Azure funkcije su jedan od načina na koji se može implementirati ta funkcionalnost.



Uvod: Šta su Azure funkcije?

Azure funkcije su nezavisni procesi koji se pozivaju kao event handleri speficičnih dogadjaja. Osnovne komponente Azure funkcija su okidači koji aktiviraju funkciju (engl. triggers), kod koji se izvršava i skladište (engl. storage) koje može da čuva logove, rezultate izvršenja itd.

Dve glavne prednosti Azure funkcija su modularnost i skalabilnost. Ove prednosti potiču od prirode Azure funkcija kao tipa serverless rešenja.



Uvod: Šta su Azure funkcije?

Modularnost podrazumeva da se odredjena logika iz sistema moze organizovati u funkcije. Te funkcije se mogu organizovati u nezavisne module i pozivati kada god se desi dogadjaj koji je od interesa.

Skalabilnost podrazumeva da se sa povećanjem opterećenja (engl. load) povećava i broj procesa. Odnosno što je broj dogadjaja veći, veći je i broj event handlera, odnosno Azure funkcija koje će obradjivati taj dogadjaj. Broj aktivnih funkcija prati količinu opterećenja, što znači da se sa povećanjem opterećenja povećava broj aktivnih funkcija, dok se sa smanjenjem opterećenja, smanjuje broj aktivnih funkcija. Ovo znači da će pored skalabilnosti, Azure funkcije biti stalno dostupne (engl. available) i responzivne.

Uvod: Serverless arhitektura

Da bi ideja Azure funkcija bila sasvim jasna, potrebno je razumeti koncept serverless arhitekture.

Serverless je vrsta cloud arhitekture gde cloud provajder dodeljuje računarske resurse procesima po potrebi, i održava servere umesto korisnika. Kod ovakvih arhitektura se resursi ne drže u memoriji, već se obrada dešava u malim vremenskim intervalima, intenzivnim korišćenjem resursa i rezultati obrade se smeštaju u nekom skladištu. Kada se aplikacija ne koristi, nema dodeljene resurse. Cena hostovanja aplikacije je srazmerna resursima koje koristi.

Uvod: Gde se mogu koristiti Azure funkcije

Neki od mogućih slučajeva korišćenja Azure funkcija:

- Kreiranje Web API-ja — Treba da se implementira endpoint za web aplikacije koji se okida HTTP okidačem
- Upload fajlova — Treba implementirati akciju koja će se okinuti nakon čuvanja bloba
- Kreiranje serverless workflowa — Treba implementirati više funkcija koje će se kaskadno okidati nakon početne akcije
- Okidanje periodičnih ili zakazanih taskova — Implementirati kod koji će se okinuti u odreeno vreme, jednom ili periodično
- Obrada podataka iz message queue — Implementirati funkciju koja će se okinuti svaki put kada se pročita poruka sa queue

Uvod: Gde se mogu koristiti Azure funkcije

Opcije koje se mogu birati pri kreiranju Azure funkcija:

- Programski jezik - Funkcije mogu da se pišu u raznim programskim jezicima (Java, C#, JavaScript, Python...)
- Automatizacija deploymenta funkcija - Veliki broj alata koji automatizuju CICD proces
- Monitoring funkcija
- Fleksibilan cenovnik usluga

Sadržaj

[T] Azure funkcije: Okidači i ulazne vrednosti

U ovoj sekciji će biti obraeni teorijski delovi Azure aplikacije. Okidači (engl. triggers) i ulazi, databinding, prava pristupa funkciji - autorizacija i monitoring. U narednoj sekciji će biti pokazani detalji implementacije.

Kod Azure funkcija važno je razlikovati termine poput okidača i ulazne vrednosti. Okidač funkcije je ponašanje na koje funkcija treba da reaguje. Funkcija može imati samo jedan okidač ali može imati više ulaza. Primer, okidač funkcije može biti HTTP zahtev, ali njeni ulazi mogu biti još i stanje u bazi, vreme u koje se poziva itd.

[T] Azure funkcije: Okidači i ulazne vrednosti

Neki od mogućih okidača Azure funkcija su dati na slici.

Type	1.x	2.x and higher[1]	Trigger	Input	Output
Blob storage	✓	✓	✓	✓	✓
Azure Cosmos DB	✓	✓	✓	✓	✓
Dapr[3]		✓	✓	✓	✓
Event Grid	✓	✓	✓		✓
Event Hubs	✓	✓	✓		✓
HTTP & webhooks	✓	✓	✓		✓
IoT Hub	✓	✓	✓		✓
Kafka[2]		✓	✓		✓
Mobile Apps	✓			✓	✓
Notification Hubs	✓				✓
Queue storage	✓	✓	✓		✓
RabbitMQ[2]		✓	✓		✓
SendGrid	✓	✓			✓
Service Bus	✓	✓	✓		✓
SignalR		✓		✓	✓
Table storage	✓	✓		✓	✓
Timer	✓	✓	✓		
Twilio	✓	✓			✓

[T] Azure funkcije: Databinding i okidači

Konceptualno kod kodiranja Azure potrebno je poznavati dva glavna koncepta: databinding i triggers. Pojam databindinga je važan za Azure funkcije i predstavlja mogućnost da se funkciji prosleuju podaci i da se generiše i šalje izlaz iz funkcije, bez potrebe da se piše poseban data access kod.

Na slici je dat kod koji ono što mu je prosledjeno na ulaz vraća kao izlaz.

```
public class Function {
    public String echo(@HttpTrigger(name = "req",
      methods = {HttpMethod.POST},  authLevel = AuthorizationLevel.ANONYMOUS)
        String req, ExecutionContext context) {
        return String.format(req);
    }
}
```

[T] Azure funkcije: Databinding i okidači

Na slici sa prethodnong slajda je pokazano kako Azure funkcija enkapsulira i olakšava korišćenje bindinga i triggera. Anotacijom @HttpTrigger se specificira da je okidač funkcije u stvari Http zahtev i specificira se tip zahteva i vrsta autorizacije o kojoj će reči biti nešto kasnije. Navodi se i će argumenti pri okidanju funkcije biti prosledjeni u String formatu.

Umesto klase String ovde je mogla biti bilo koja druga klasa, čak i neka korisnički definisana. Sama Azure funkcija će odraditi data binding proces. U ovom slučaju na Azure funkciju možemo gledati kao na neki REST endpoint.

Azure funkcije rade databinding u oba smera, kada se prihata ulaz funkcije i kada se šalje izlaz funkcije.

[T] Azure funkcije: Autorizacija

Pri kreiranju okidača Azure funkcije, može se specificirati i nivo autorizacije. Tri moguća nivoa autorizacije su:

- Function - traži se neki ključ koji je naveden u podešavanju aplikacije (basic nivo pristupa) pri okidanju funkcije
- Anonymous - ne traži se ključ pri okidanju funkcije
- Admin - potreban je admin ključ (admin privilegije) za poziv funkcije

[T] Azure funkcije: Monitoring

Nakon deploymenta Azure aplikacije, potrebno je pratiti njen rad. U Azure Cloudu postoji Application Insights usluga koja omogućava rad sa logovima Azure funkcije. Moguće je preusmeriti logove koje funkcija generiše na ovaj servis. Kada se kreiraju logovi i upisuju na Application Insights, postoje različiti nivoi logova. Nivoi logova su dati na slici .

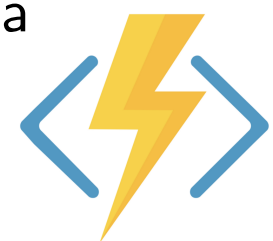
LogLevel	Code	Description
Trace	0	Logs that contain the most detailed messages. These messages may contain sensitive application data. These messages are disabled by default and should never be enabled in a production environment.
Debug	1	Logs that are used for interactive investigation during development. These logs should primarily contain information useful for debugging and have no long-term value.
Information	2	Logs that track the general flow of the application. These logs should have long-term value.
Warning	3	Logs that highlight an abnormal or unexpected event in the application flow, but don't otherwise cause the application execution to stop.
Error	4	Logs that highlight when the current flow of execution is stopped because of a failure. These errors should indicate a failure in the current activity, not an application-wide failure.
Critical	5	Logs that describe an unrecoverable application or system crash, or a catastrophic failure that requires immediate attention.
None	6	Disables logging for the specified category.

Figure: Slika 3 - Nivoi logova

[T] Azure funkcije: Monitoring

Različiti nivoi logova omogućavaju da se prikupljaju različiti logovi u zavisnosti od podešenog okruženja. Moguće je podesiti različita okruženja tako da skupljaju samo logove tipa Debug ili Information i da samo njih prikazuju. Na ovaj način se mnogo lakše upravlja logovima i prikazuju se samo oni neophodni.

Na Azure cloudu postoji rešenja poput Azure Monitor logs koja omogućavaju perzistiranje logova na neko skladište. Takodje, moguće je motriti na trenutno aktivne instance Azure funkcije pomoću Scale controller servisa. Ovakvim načinom monitoringa funkcije moguće je da sam korisnik proceni da li treba smanjiti ili povećati broj aktivnih funkcija i koliko im resursa treba dati. Takodje Scale controller servis pruža mogućnost skladištenja podataka o performansama.



Sadržaj

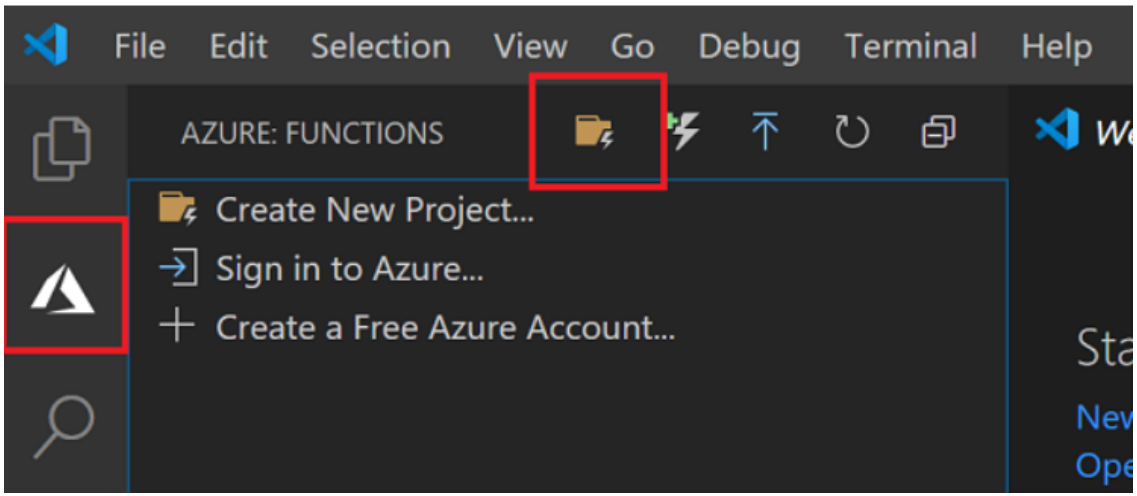
[P] Azure funkcije: Primer

U nastavku će biti opisan proces kreiranja i deploymenta Azure funkcija na Azure cloud. Primer će biti odradjen u programskom jeziku Java. Biće opisan način implementacije Http okidača, kreiranje ključa i deployment aplikacije. Biće i par reči o monitoringu funkcija.

[P] Azure funkcije: Primer

Razvoj i deployment Azure aplikacije se može kompletno odraditi u *Visual Studio Code* okruženju. Potrebno je samo instalirati plugin *Azure functions*.

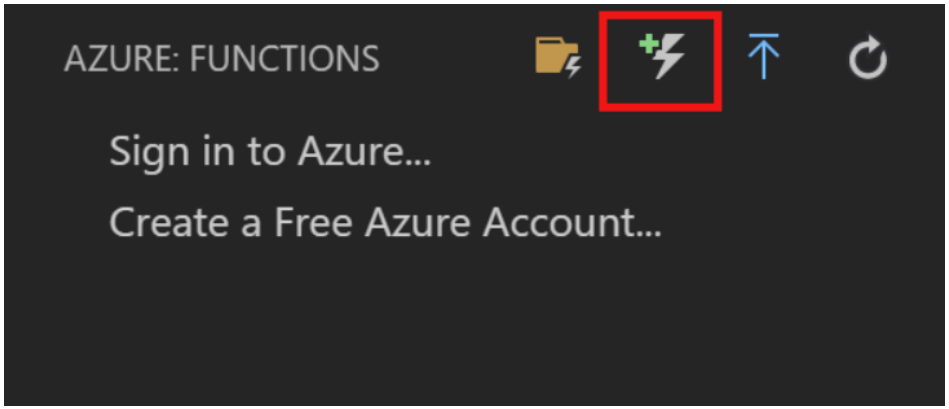
Prvi korak nakon instaliranja plugina je kreiranje novog projekta. U desnom sidebaru treba izabrati Azure functions plugin, i nakon toga kliknuti na Create new project opciju.



[P] Azure funkcije: Primer

Nakon toga treba izabrati direktorijum. Pri kreiranju projekta potrebno je izabrati i verziju Jave, group ID, artifact ID, verziju, ime paketa gde je smešten kod, ime aplikacije i nivo autorizacije. Za ovaj primer, izabrani nivo autorizacije biće Anonymous. Defaultno je build tool za Java aplikaciju Maven, medjutim lako se može izabrati i Gradle.

Ako je potrebno izabrati drugačiji okidač za funkciju onda je potrebno funkciju kreirati klikom na dugme Create Function:



[P] Azure funkcije: Primer

Za funkciju kreiranu preko Create new project opcije, VS Code okruženje kreira Azure funkciju sa HTTP okidačem. Struktura foldera koji se kreira je:

```
FunctionsProject
 | - src
 | | - main
 | | | - java
 | | | | - FunctionApp
 | | | | | - MyFirstFunction.java
 | | | | | - MySecondFunction.java
 | - target
 | | - azure-functions
 | | | - FunctionApp
 | | | | - FunctionApp.jar
 | | | | - host.json
 | | | | - MyFirstFunction
 | | | | | - function.json
 | | | | - MySecondFunction
 | | | | | - function.json
 | | | | - bin
 | | | | - lib
 | - pom.xml
```

[P] Azure funkcije: Primer

U fajlu com.function.Function.java se generiše kod koji će pročitati ulazni parametar i na izlazu generisati poruku Hello, *prosledjeni-parametar*. Ovaj kod je dat na slici .

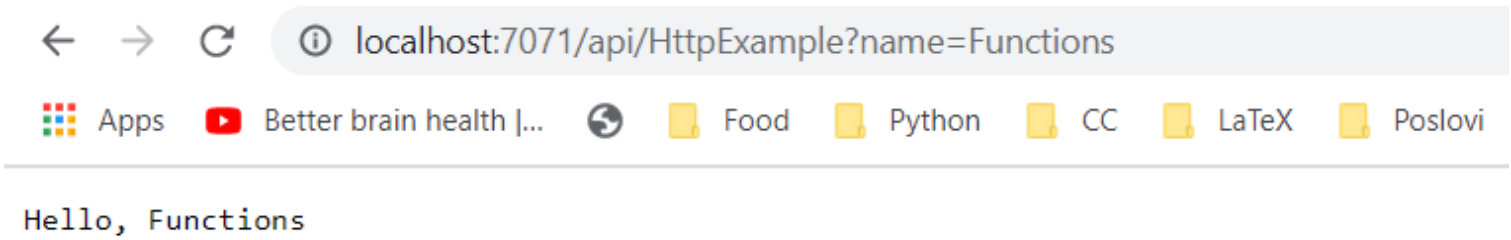
```java
@FunctionName("HttpExample")
public HttpResponseMessage run(
        @HttpTrigger(
            name = "req",
            methods = {HttpMethod.GET, HttpMethod.POST},
            authLevel = AuthorizationLevel.ANONYMOUS)
            HttpRequestMessage<Optional<String>> request,
        final ExecutionContext context) {
    context.getLogger().info("Java HTTP trigger processed a request.");

    // Parse query parameter
    final String query = request.getQueryParameters().get("name");
    final String name = request.getBody().orElse(query);

    if (name == null) {
        return request.createResponseBuilder(HttpStatus.BAD_REQUEST).body("Please pass a name on the query string or in the request
    } else {
        return request.createResponseBuilder(HttpStatus.OK).body("Hello, " + name).build();
    }
}
```

[P] Azure funkcije: Primer

Ovako kreirana aplikacija se izvršava na portu 7071 i izloženi endpoint je http://localhost:7071/api/HttpExample?name=Functions. name=Functions je jedan od ulaznih parametara funkcije. Izlaz funkcije je dat na slici.



[P] Azure funkcije: Primer

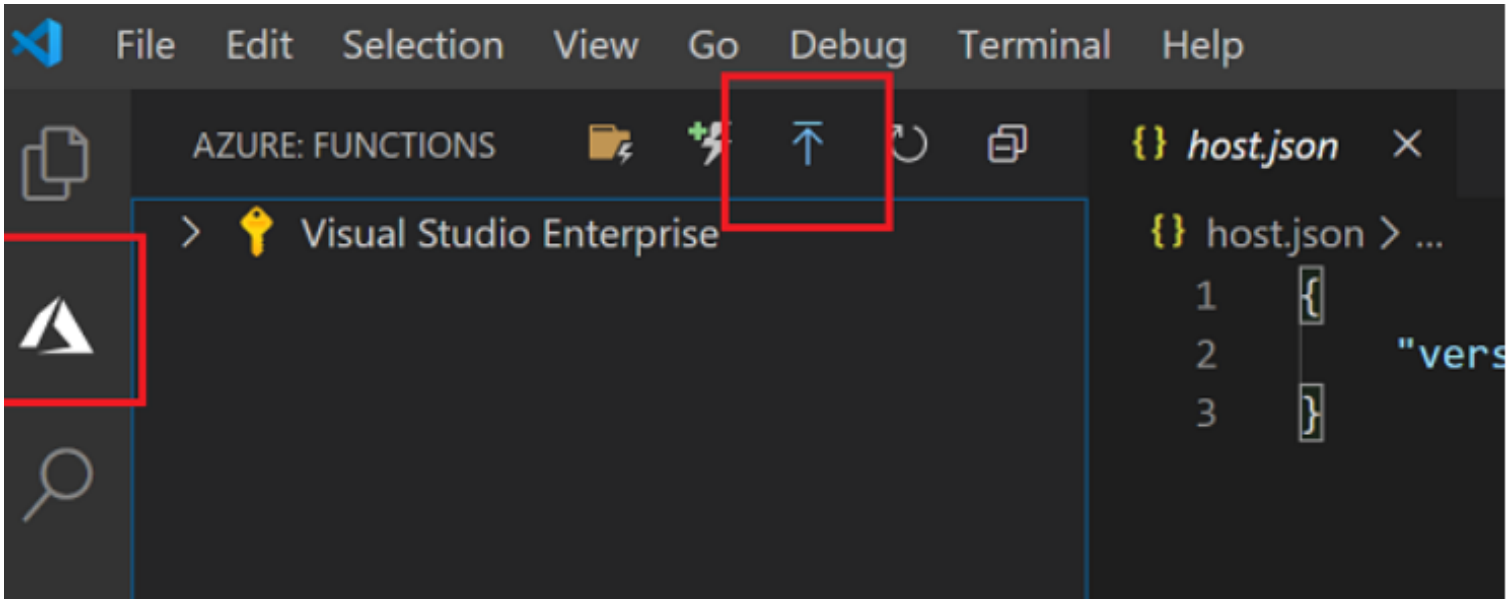
Konzola VS Code okruženja je data na slici. U konzoli postoje informacije o zahtevu koji je primljen.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: Task - host start

```
[1/30/2020 7:26:15 PM] Executing HTTP request: {
[1/30/2020 7:26:15 PM]   "requestId": "6660fd29-2b0d-41fc-9a17-a4f700415a84",
[1/30/2020 7:26:15 PM]   "method": "GET",
[1/30/2020 7:26:15 PM]   "uri": "/api/HttpExample"
[1/30/2020 7:26:15 PM] }
[1/30/2020 7:26:15 PM] Executing 'Functions.HttpExample' (Reason='This function was programmatically called vi
a the host APIs.', Id=65d05c7f-5192-4ff2-a1c6-d8b3a78385d3)
[1/30/2020 7:26:15 PM] JavaScript HTTP trigger function processed a request.
[1/30/2020 7:26:15 PM] Executed 'Functions.HttpExample' (Succeeded, Id=65d05c7f-5192-4ff2-a1c6-d8b3a78385d3)
[1/30/2020 7:26:15 PM] Executed HTTP request: {
[1/30/2020 7:26:15 PM]   "requestId": "6660fd29-2b0d-41fc-9a17-a4f700415a84",
[1/30/2020 7:26:15 PM]   "method": "GET",
[1/30/2020 7:26:15 PM]   "uri": "/api/HttpExample",
[1/30/2020 7:26:15 PM]   "identities": [
[1/30/2020 7:26:15 PM]     {
[1/30/2020 7:26:15 PM]       "type": "WebJobsAuthLevel",
[1/30/2020 7:26:15 PM]       "level": "Admin"
[1/30/2020 7:26:15 PM]     }
[1/30/2020 7:26:15 PM]   ],
[1/30/2020 7:26:15 PM]   "status": 200,
[1/30/2020 7:26:15 PM]   "duration": 39
[1/30/2020 7:26:15 PM] }
```

[P] Azure funkcije: Primer

Deployment funkcija na Azure cloud se radi klikom na dugme Deploy to function app... Opcija je data na slici.



[P] Azure funkcije: Primer

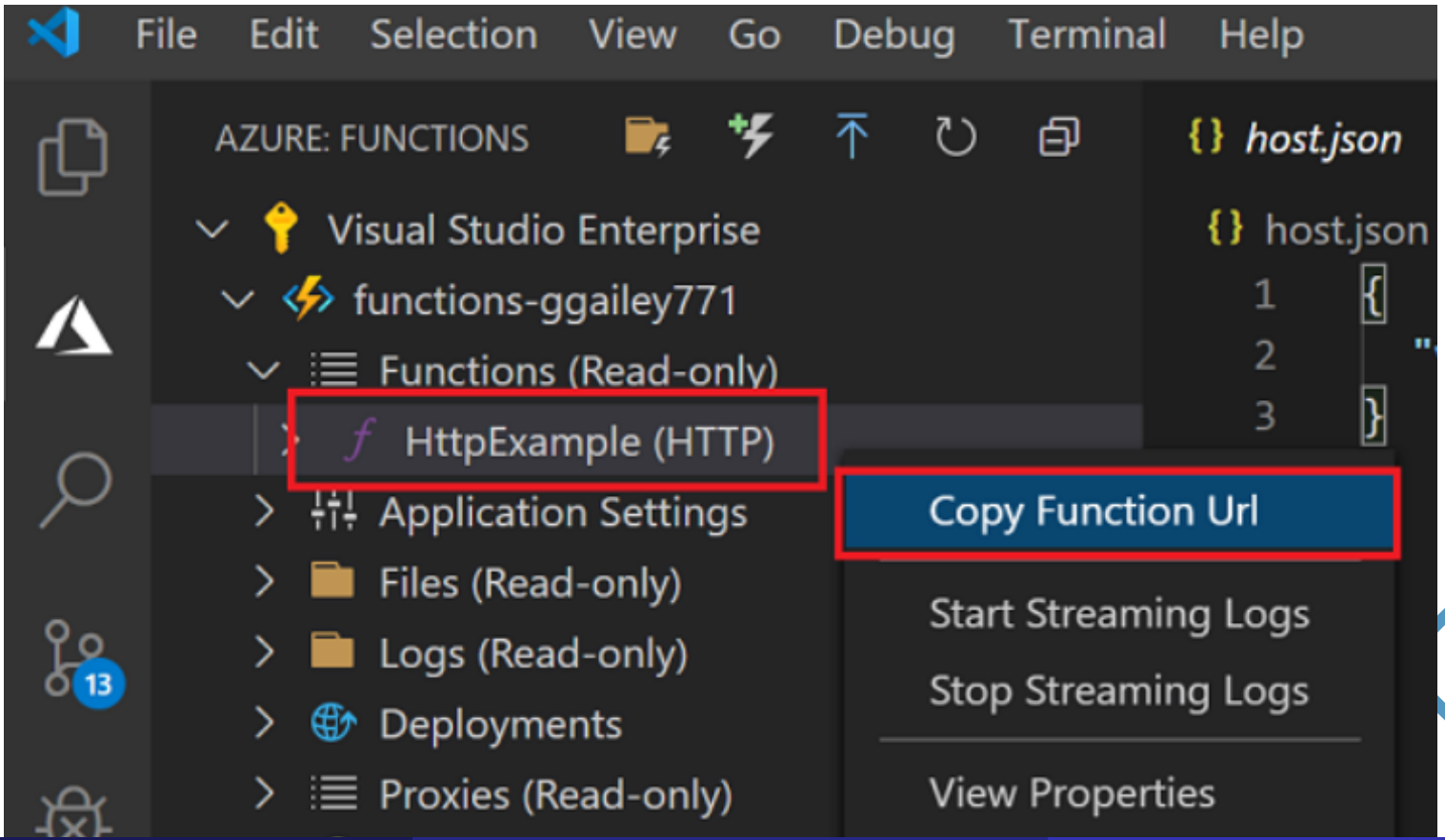
Opcije koje je potrebno izabrati pri deploymentu su:

1. Izabrati direktorijum gde se nalazi funkcija
2. Treba izabrati subscription plan (vrstu plaćenog plana)
3. Izabrati globalno jedinstveno ime za funkciju, to ime će se nalaziti u URL-u funkcije. Mora biti jedinstveno na nivou svih Azure funkcija.
4. Izabrati lokaciju za novi resurs, za bolje performanse treba izabrati region koji je u blizini, npr West Europe

[P] Azure funkcije: Primer

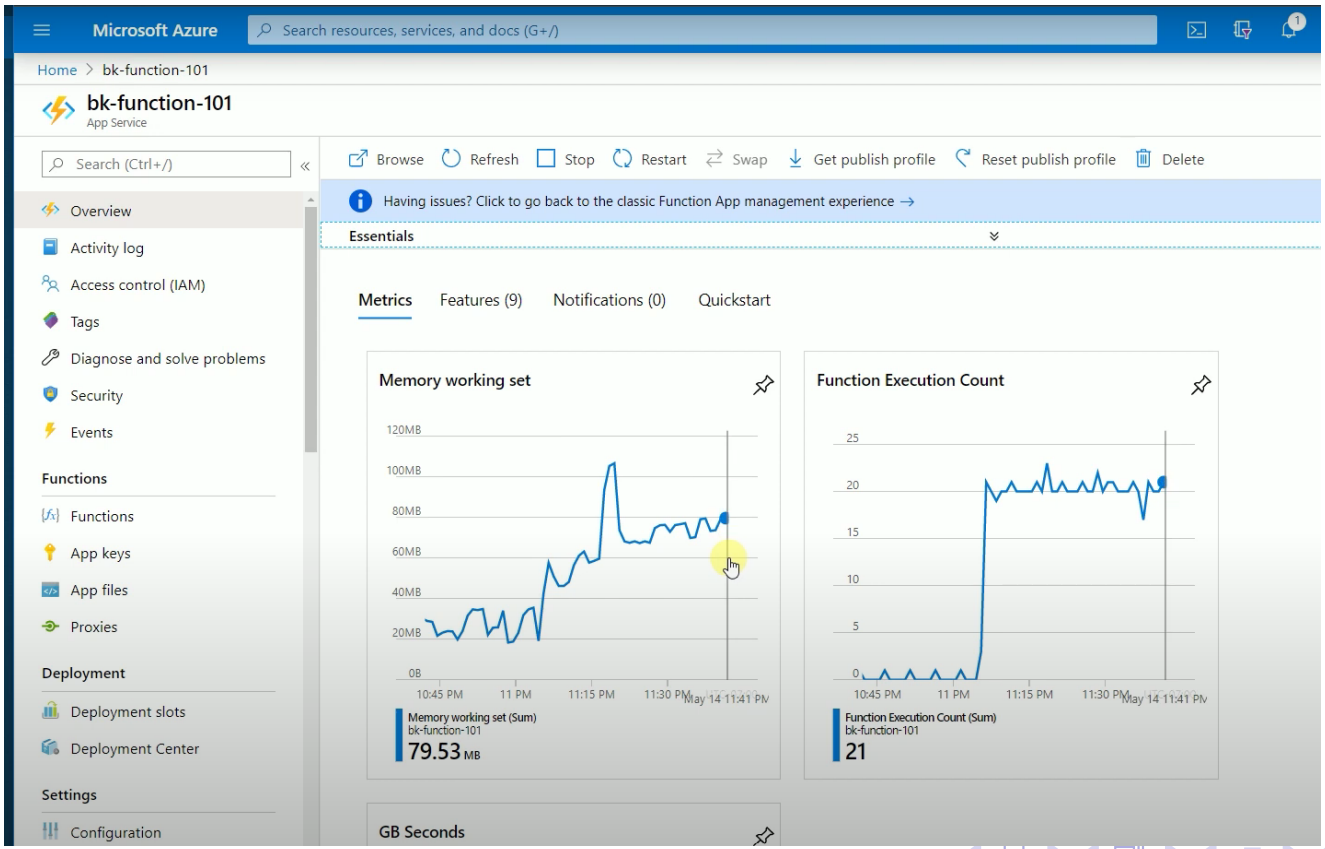
Na Azure cloudu se, uz resource grupu, storage account, consumption plan, kreira i function app (okruženje/kontejner za izvršavanje koda) i instanca Application Insights za monitoring sistema.

Aplikaciji se pristupa linka koji se može pribaviti kao na slici .



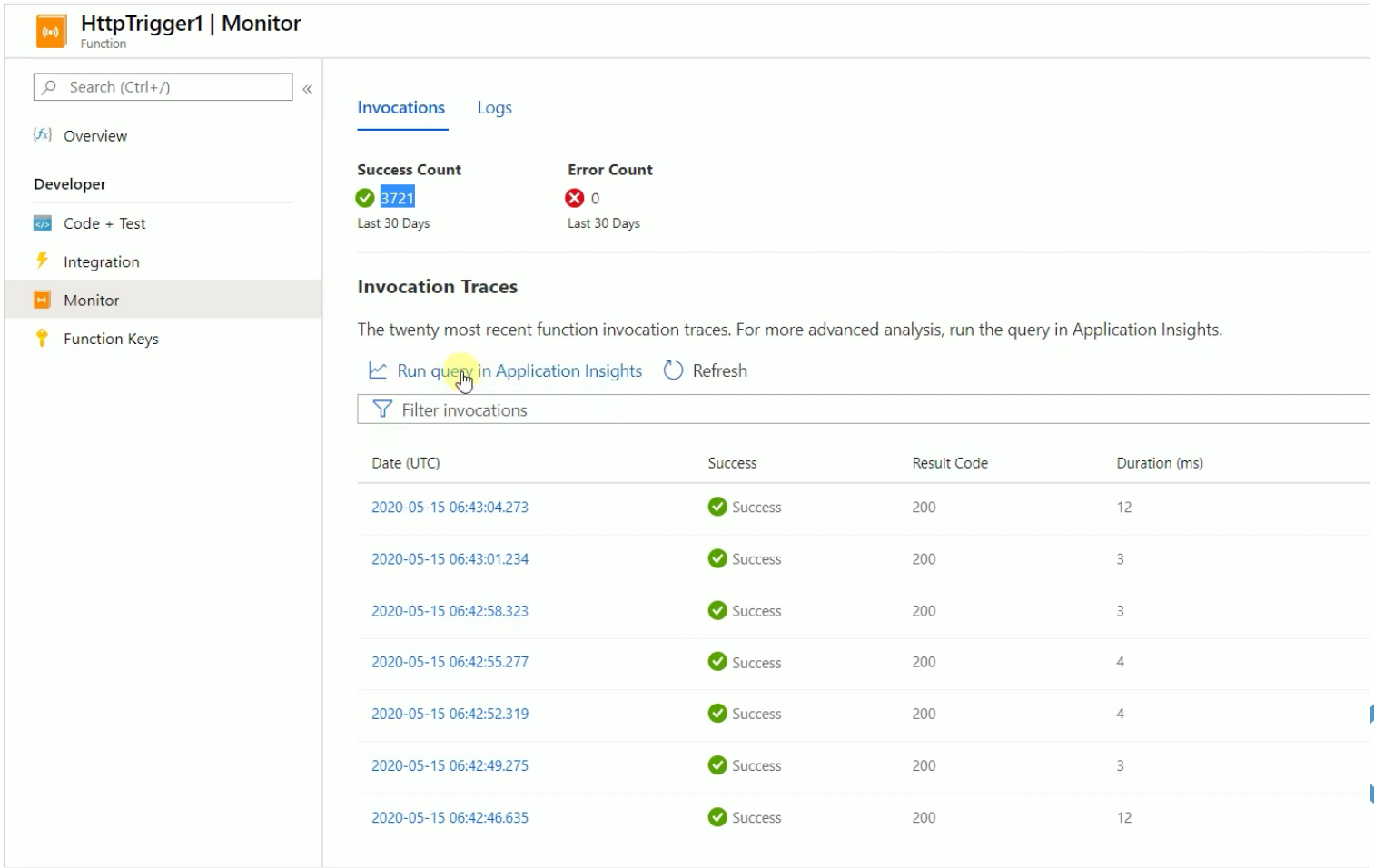
[P] Azure funkcije: Primer

U ovom slučaju se aplikaciji pristupa kao http://FUNCTION-APP-NAME.azurewebsites.net/api/HttpExample?name=Functions, gde je FUNCTION-APP-NAME jedinstveno ime aplikacije koje je zadato pri hostovanju aplikacije. Pri logovanju na Azure cloud, može se videti dashboard tabla hostovane funkcije.



[P] Azure funkcije: Primer

Preko dashboard table se može pristupiti i opciji za monitoring funkcije i njenog okidača. Ovde je prikazan broj uspešno i neuspešno izvršenih zahteva i neke dodatne informacije o njima.

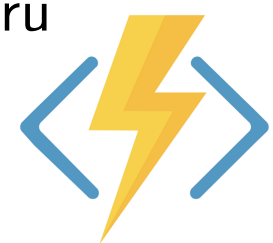


[P] Azure funkcije: Kreiranje ključeva

Za ovaj primer nisu kreirani ključevi, medjutim Azure funkcije omogućavaju njihovo kreiranje. Ako HTTP nivo pristupa na HTTP triggerovanoj funkciji nije setovan na anonymous, za sve ostale nivoe pristupa se uz poziv okidača mora se proslediti iz API pristupni ključ (engl. access key).

Postoje dva opsega pristupa (engl. access scope) za ključeve na nivou funkcija (engl. function-level keys):

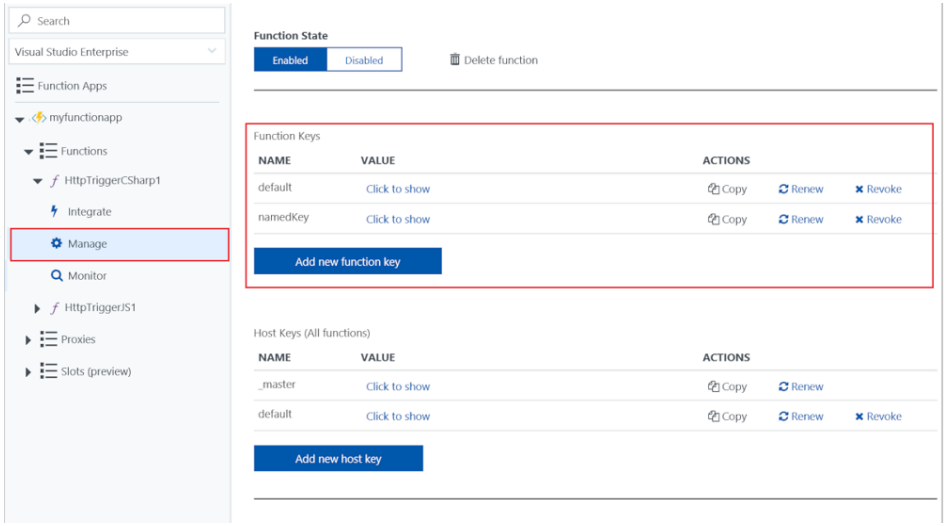
- Opseg pristupa na nivou funkcije - Ovi ključevi se primenjuju samo na funkcije za koje su definisani
- Opseg pristupa na nivou hosta - Ključevi sa host scope-om se mogu koristiti za pristup svim funkcijama u function app kontejneru



[P] Azure funkcije: Kreiranje ključeva

Svaki function app kontejner ima jedan ključ sa administratorskim privilegijama/scopeom (master ključ), kojim se može pristupiti svakoj funkciji u tom kontejneru. Ako neka funkcija ima admin access level, zahtevi koji joj stižu svi moraju da imaju master ključ. Ako funkcija ima Function pristupni nivo, njoj se mogu proslediti ili host scope ključevi ili function scope ključ koji je kreiran za tu funkciju.

Ključevi se čuvaju kao deo function app kontejnera. Upravljanje ključevima se može raditi u kartici Manage u function app kontejneru.



[P] Azure funkcije: Kreiranje ključeva

Slanje ključa pri autorizaciji se radi tako što se u URL HTTP zahteva u promenljivoj code šalje vrednost pristupnog ključa.

http://FUNCTION-APP-NAME.azurewebsites.net/api/HttpExample?name=Functions?code=APIKEY

Ovaj ključ može i umesto u URL da se dostavi u headeru zahteva preko polja x-functions-key. Vrednost ključa može da bude bilo koji ključ definisan na nivou funkcije i ključa.

Funkcije takoe podržavaju anonimne zahteve koji ne zahtevaju ključ. Takoe funkcije mogu da zahtevaju master ključ.

Kraj

Pitanja?