|  |  |  |
| --- | --- | --- |
|  | UNIVERZITET U NOVOM SADU  **FAKULTET TEHNIČKIH NAUKA U NOVOM SADU** |  |

Goran Pavićević PR100/2015

**Asset Management**

Dokumentacija projektnog zadatka

- Master akademske studije-

-Novi Sad, 2020-

Sadržaj

**Uvod3**

Opis rešavanog problema2

Arhitektura3

Simulator3

Scada3

CalculationEngine3

NetworkModelService3

UI Adapter3

Publish-Subscribe mehanizam3

TransactionService3

Korisnički interfejs3

Standardi i modeliranje EES**4**

Obezbediti istorijske podatke5

Simulacija EES sa kritičnom misijom**4**

Simulator5

CLOUD**4**

Cloud istorijska baza podataka5

Napredni računarski sistemi**4**

Verovatnoća5

Sigurnost i bezbednost**4**

Model - Elementi sistema5

**Uvod**

Cilj svake uspešne kompanije jeste da smanjenjem troškova poveća svoj profit. To se može uraditi na više načina, ali jedan od najefikasnijih i najbržih jeste korišćenjem softverskog rešenja. Upravo razvoj jedne takve aplikacije, čijim korišćenjem se efikasno mogu smanjiti kapitalni troškovi i jeste cilj našeg projekta.

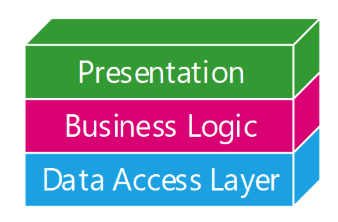
Aplikacija bi trebalo da obuhvatitok kretanja opreme(prekidači i transformatori) kroz elektroenergetski sistem, a samim tim aplikacija se može iskoristiti od više timova u okviru jedne firme. Osim toga aplikacija treba da podrži komandovanje nad opremom, praćenje i čuvanje tih akcija, kao i generisanje raznih izveštaja. Čuvanje tih akcija nam omogućava jednu od najbitnijih funkcionalnosti aplikacije, a to je predikcija datuma kad je potrebno poslati opremu na popravku. Na ovaj način se unapred može odrediti i isplanirati remont, što značajno smanjuje mogućnost dolaska do havarije i gubitka napajanja kod velikog dela korisnika.

*Opis rešavanog problema*

Napraviti aplikaciju za podršku poslovima održavanja u elektrodistributivnoj mreži, namenjenu unosu podataka o tehničkim karakteristikama i listama održavanja elektroenergetske opreme, u bazu podataka, kao i za pregled i obradu podataka iz baze i za pravljenje raznovrsnih izveštaja.

## *Arhitektura*

Arhitektura projekta je podeljena u tri sloja (Slika 1.1)



***Slika 1.1*** *– Slojevi aplikacije*

* Prezentacioni sloj predstavlja korisnički interfejs kroz koji korisnik ima uvid u dešavanja u sistemu i vrši interakciju sa istim.
* Biznis logiku obuhvataju servisi: **PubSub, CE, Scada, NMS, UI Adapter**. Logika je sakrivena od korisnika i može da bude distribuirana jer je komunikacija između svih servisa preko mreže.
* Sloj podataka je Azure SQL Server kome servisi pristupaju koristeći razvijenu biblioteku za komunikaciju sa istorijskom bazom podataka.

*Slika 1.2* je slikovita reprezentacija arhitekture celokupnog sistema.

### *Simulator*

Simulator predstavlja third-party aplikaciju koja služi za simuliranje vrednosti elemenata u polju. Vrednosti mogu da se zadaju ručno kliktanjem po korisničkom interfejsu ili nekom aplikacijom koja komunicira sa njim i šalje komande. Za komunikaciju se koristi TCP, a format poruke mora da poštuje MODBUS protokol.

### *Scada*

Scada je servis koji vrši periodičnu akviziciju stanja vrednosti na simulatoru koristeći MODBUS protokol. Kada primi trenutno stanje, šalje se razlika trenutnog i prethodnog stanja na CalculationEngine servis na interpretaciju.

### *CalculationEngine (CE)*

Servis koji prima podatke sa Scada-e i interpretira njihovo značenje. Dužan je da proveri da li se neki od uređaja nalazi u alarmnom stanju i da zapisuje podatke u Azure SQL bazu podataka (na slici HIST). Za to vreme se pristigli podaci proslede na UI Adapter. Ako se utvrdi da je neki uređaj u alarmnom stanju, dodatne informacije se šalju na UI Adapter da bi se korisnički interfejs informisao o tom stanju. Na zahtev UI Adapter-a generiše izveštaje i šalje ih nazad.

### *NetworkModelService (NMS)*

Servis koji čuva konektivnost mreže. Podatke o konektivnosti može da primi od Importer aplikacije i sa korisničkog interfejsa. Pristigle podatke validira i primenjuje ako su validni, u suprotnom informiše pošiljaoca o tome šta je bila greška u validaciji. Nakon primenjenog modela, ostali servisi mogu po potrebi da šalju zahteve i traže informacije o statičkom modelu.

### *UI Adapter*

Specifičnost ovog servisa je u tome da služi kao most između korisničkog interfejsa i ostatka sistema. Korisnički interfejs šalje jedinstvene zahteve i dužnost ovog servisa je da koordiniše zahteve na jedan ili više servisa da bi u potpunosti odgovorio na zahtev korisničkog interfejsa. Tako korisnički interfejs i ostatak sistema nemaju deljene modele podataka.

### *Publish-Subscribe mehanizam (PubSub)*

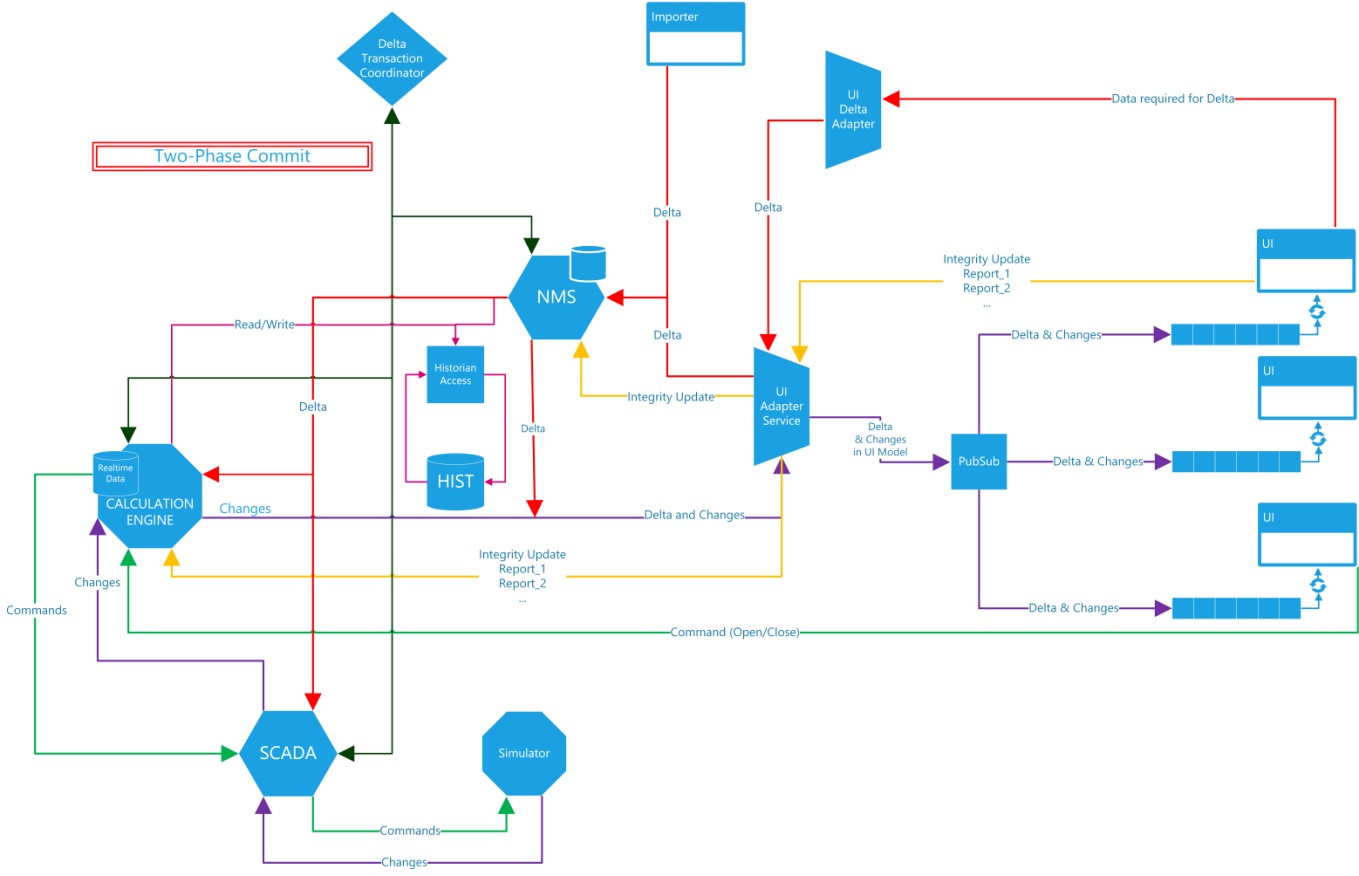
Da bi moglo da se podrži više od jednog korisničkog interfejsa i da to bude nezavisno od celog sistema koristi se publish-subscribe mehanizam koji će pristigle podatke da demultipleksira na svaki od korisničkih interfejsa tako da svi budu u sinhronizaciji.

### *TransactionService*

Funkcionalnost ovog servisa je koordinacija distribuirane transakcije kroz sistem. Kada podaci stignu na **NMS** i uspešno prođu validacije proslede se ostalim servisima. Nakon toga se servisi od interesa

prijavljuju transakcionom servisu i kada se svi prijave **NMS** obavesti transakcioni servis da distribuirana transakcija može da se pokrene.

### *Korisnički interfejs (UI)*

Korisnik za interakciju sa sistemom ima na raspolaganju korisnički interfejs. Funkcionalnosti mu omogućavaju manipulaciju uređajima koji se nalaze u sistemu, njihovo prebacivanje u funkciju i van funkcije, prikaz mnoštva izveštaja, popravku.

***Slika 1.2*** *Arhitektura celog sistema*

**Standardi i modeliranje elektroenergetskih sistema**

*Obezbediti istorijske podatke*

Baza podatak sadrži skup svih informacija datog sistema. Pri svakom ažuriranju podataka, podaci se unose u bazu, ne brišući prethodne.

Svaka promena koja se izvrši nad prekidačima i transformatorima, dodaje se u istorijsku bazu podataka ili HIST bazu. Istorijska baza sadrži šest tabela, od kojih su prve tri koje nose sve informacije, dok su druge tri pomoćne, radi bržeg rada samog programa.

Za pristup bazi koristimo *Connection String* koji se nalazi u *App.config* fajlu.

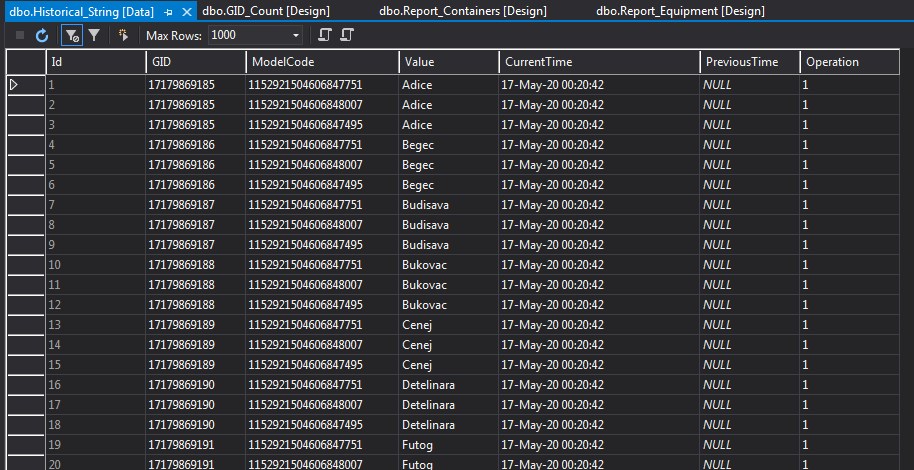
Tabele su:

* *Historical DateTime*
* *Historical Long*
* *Historical String*
* *GID Count*
* *Report Containers*
* *Report Equipment*

Prve tri tabele ( *DateTime,Long i String* ) sadrže ista polja, jedina je razlika jeste u tipu *Value* polja. Svaka od ovih tabela, u polje *Value*, upisuje vrednost koja je zapisana u nazivu (*DateTime je datetime2, Long je bigint, a za String je nvarchar* ). Polja u ovim tabelama su:

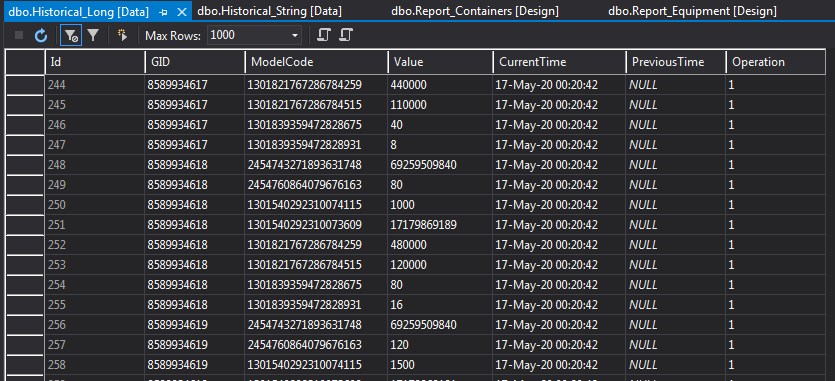
* *Id - Primarni ključ*
* *GID - globalni identifikator*
* *ModelCode - definisan u ModelCode-ovima u okviru NMS-a*
* *Value - vrednost objekta*
* *CurrentTime - vreme kada se desila promena*
* *PreviousTime - poslednje vreme prethodne promene*
* *Operation - tip operacije, može biti vrednost 0-delete,1-insert,2-update*

Sadržaj tih tabela izgleda ovako:

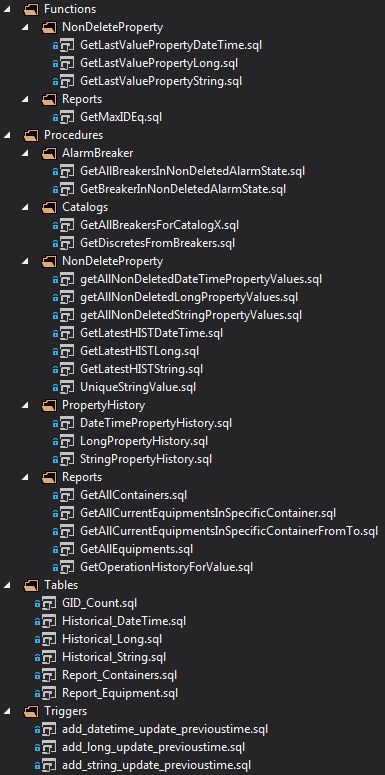
******

***Slika2.1****. Deo sadržaja tabele Historical String*

Zatim prikaz druge tabele Historical Long, gde polje Value ima drugačiju vrednost:

******

***Slika 2.2*** *Deo sadržaja tabele Historical Long*



Iz datih primera primećujemo sličnost u obe tabele. Isto je i sa trećom tabelom.

Kada obrišemo određeni element iz sistema, on će se ubeležiti u HIST bazu, a polje Operation će imati vrednost 0, što predstavlja delete operaciju.

Korišćenjem procedura i funkcija vadimo određene podatke iz baze koje su nam potrebne.

Procedure i funkcije su sortirane u foldere u zavisnosti koju namenu ima i svaka u svom nazivu opisuje koje informacije povlači iz baze.

Tri funkcije (triggers) služe da bi se izmenili određeni podaci u okviru tabela. U našem slučaju koristimo samo da bi izmenili polje *PreviousTime* u okviru sve tri *Historical* tabele, koje je do momenta izmene imalo vrednost null.

***Slika 2.3****. Lista svih procedura,funckija,tabela i trigger-a*

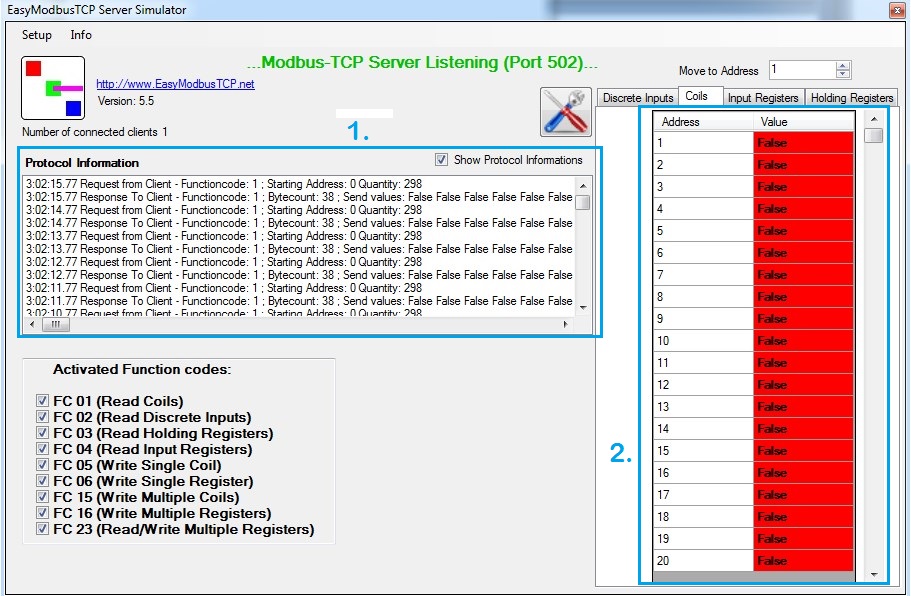
**Simulacija elektroenergetskih sistema sa kritičnom misijom**

*Simulator*

Za simulaciju sistema koristimo EasyModbusTCP Server Simulator, sa kojim vršimo komandovanje stanja elemenata. Sve izmene se šalju na Scada servis preko Modbus TCP protokola.

Vrednost na adresama se dvoklikom i dve moguće vrednosti su True i False.Inicijalna vrednost je False na svim elementima. Ove dve vrednosti u sistemu predstavljaju otvaranje i zatvaranje elemenata.

Čitanje vrednosti se vrši na svakih 1000 milisekundi.

***Slika3.1*** *Inicijalno stanje simulatora pri pokretanju sistema*

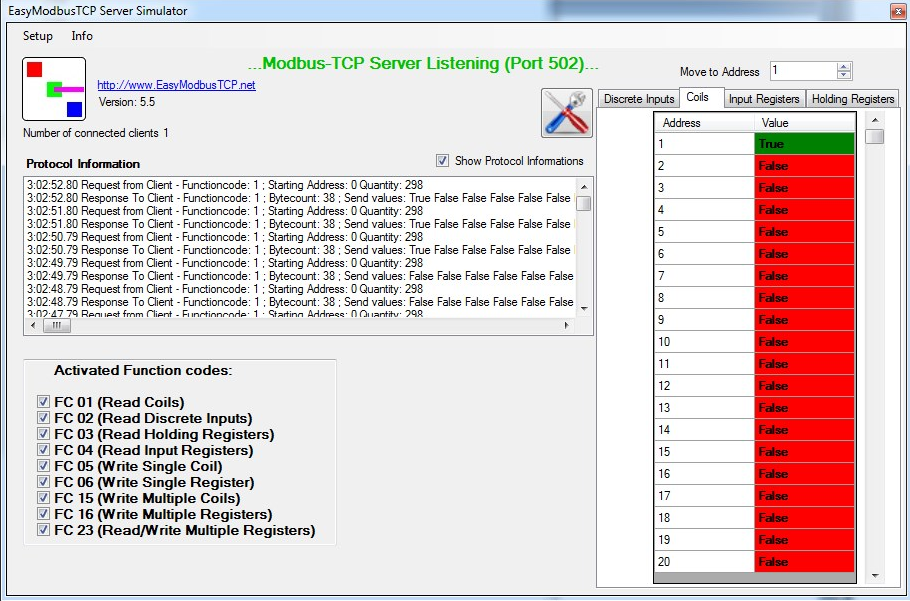
Na slici 4. :

1. Informacije koje simulator prima i šalje ( request i response )

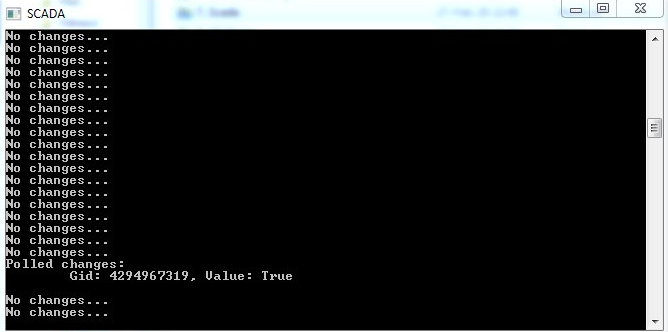
-Vrednost na svim adresama je False ( zatvoren )

1. Tab Coils sa mogućim adresama za komandovanje

Zatim nakon jedne manipulacije jednog elementa primećujemo da je na datoj adresi promenjena boja i vrednost na True i takva informacija se šalje dalje na skadu.



***Slika3.2*** *Prikaz simulatora i skade pri jednoj manipulaciji*



***Slika3.3****Manipulisana vrednost pristigla na Scada komandnom prozoru*

Primećujemo u okviru Protocol Information prozora na simulatoru da se pooling vrši svake sekunde i da se pri svakom pooling-u šalju sve moguće adrese, a ne samo one na kojima je izvršena manipulacija.

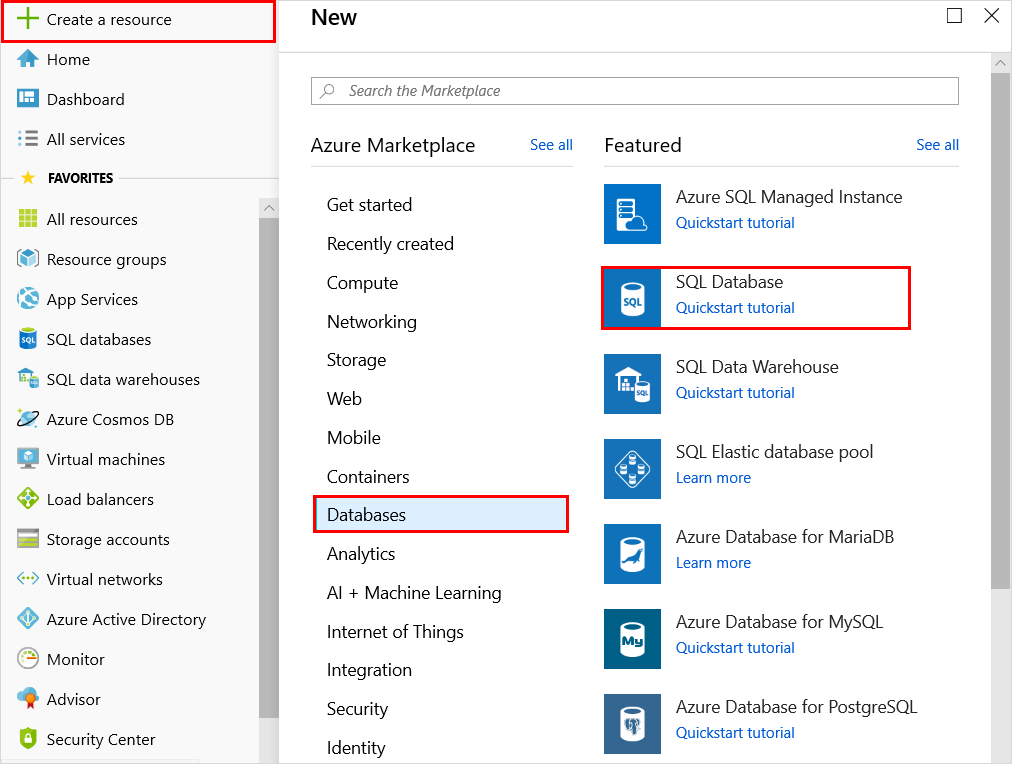
Ukupan broj adresa nad kojim je moguće komandovanje je od 1 do 65534. Svaka adresa je povezana sa GID-om. Princip rada ovakvog simulatora je zasnovana na PubSub modelu (Publish Subscribe). Scada se pretplati (Subscribe) na Network Model Service(NMS). NMS će zatim primljenu listu GID-ova da prosledi nazada na Scada servis.

**Održavanje i kontrola kvaliteta elektroenergetskog (CLOUD)**

*Cloud istorijska baza podataka*

Za potrebe Cloud-a koristili smo Azure servis koji je stvorio Microsoft, za pravljenje, testiranje, raspoređivanje i upravljanje aplikacijama i servisima. Sve podatke skladištili smo u HIST bazu podataka na datom cloud-u.

Na Azure portalu potrebno je kreirati SQL bazu. Ovako izgleda redosled koji treba ispratiti za kreiranje:



***Slika4.1*** *Prikaz Azure portala pri kreiranju baze podataka*

Nakon toga, otvoriće se prozor gde treba popuniti podatke kao sto su admin name i password za pristup bazi, ime baze itd., koje ćemo kredencijale koristiti u projektu prilikom pristupa cloud-u.

.............................

**Napredni računarski sistemi sa kritičnom misijom u elektroenergetici**

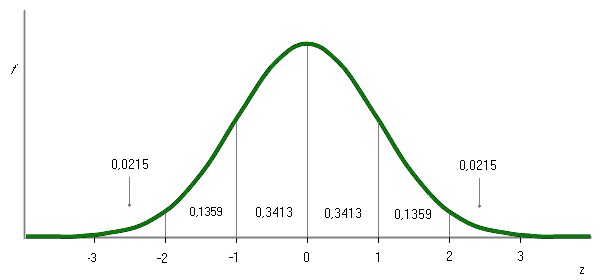
*Verovatnoća*

Da bi smo poboljšali sistem uvedena je verovatnoća, sa kojom računamo mogućnost ispada određenog elementa u sistemu. Sa verovatnoćom iskazujemo da li će se neki događaj desiti ili ne sa vrednostima u razmaku od 0 (0%) što predstavlja da je neki događaj skoro nemoguć, pa sve do 1(100%) što predstavlja da je neki događaj skoro siguran da će se dogoditi.

Verovatnoća koja je korištena u projektu jeste **normalna distribucija**.

Za izračunavanje procenta korišćena je formula:

**Z** vrednost predstavlja meru udaljenosti rezultata od aritmetičke sredine.

***Slika5.1*** *Izražena standardna devijacija*

Na **X osi** su prikazane sve vrednosti koje jedna varijabla može da ima u odnosu na aritmetičku sredinu. Na **Y os**i su prikazane frekvencije, odnosno koliko je puta neka vrednost opažena u radu sistema. Tamo gde je rezultat najintenzivniji biće bliže aritmetičkoj sredini, a što je manje intenzivan, to će se udaljavati srazmerno.

Procenat rezultata između -1z i +1z iznosi 68.26%, odnosno da se neki rezultat nalazi između te dve vrednosti je 0.6826.

Procenat rezultata između vrednosti -2z i +2z iznosi 95.44% odnosno 0.9544.

Procenat rezultata izmežu -3z i +3z iznosi 99.74%, odnosno 0.9974.

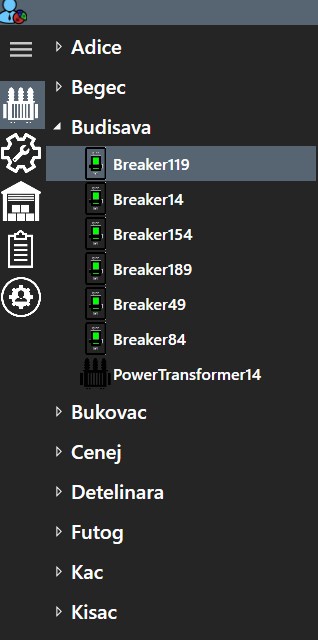
Aritmetička sredina predstavlja tačku **µ(mi)** i u toj tački se očekuje najviše rezultata.

Vrednost između dva susedna broja na X-osi predstavlja **Ơ(sigma)**

U *CalculationEngineService* koristimo klasu NormalDistribution gde se pozivaju metode i vrše izračunavanja.

Poziv te klase vrši se na svakih 75s, sa obzirom da ne dolazi do velikih promena pri svakom računanju, možemo dozvoliti to vreme, pri čemu će moguća greška biti zanemarljivo mala.

**Sigurnost i bezbednost u Smart Grid sistemima**



*Model - Elementi sistema*

Ceo sistem kojim upravljamo podeljen je na tri pozicije u kojima se nalaze elementi. Moguće pozicije su:

1. **Substation** - stanica (polje) na kojoj naši elementi prelaze u radno stanje
2. **Warehouse** - magacin u kom su elementi skladišteni
3. **Maintenance** - mesto gde se svi elementi popravljaju ili se vrši redovan servis

Svaki od ovih pozicija ima svoje kapacitete i posebne funkcije u okviru istih u zavisnosti od namene.

Elementi sistema koju su skladišteni su:

1. **Breaker** - *prekidači*
2. **Power** **Transformer** - *transformatori* ***Slika6.1*** *Prikaz stanica (substation)*

Svaki prekidač sadrži polja

* **GID** - *globalni identifikator*
* **SwitchLimit** - *rok trajanja po broju iteracija paljenja i gašenja prekidača*
* **ShortCircuitLimit** - *rok trajanja po broju iteracija posle kog treba na servis*
* **ExpirationTime** - *rok trajanja po katalogu*
* **ServiceTime** - *rok za odlazak na servis*
* **EquipmentContainer** - *referenca na kontejner*
* **CatalogName** - *naziv kataloga*
* **Price** - *cena*
* **Name** - *ime*

Svaki prekidač sadrži i po dva ***Discrete***-a za svako stanje.

Svaki transformator sadrži polja:

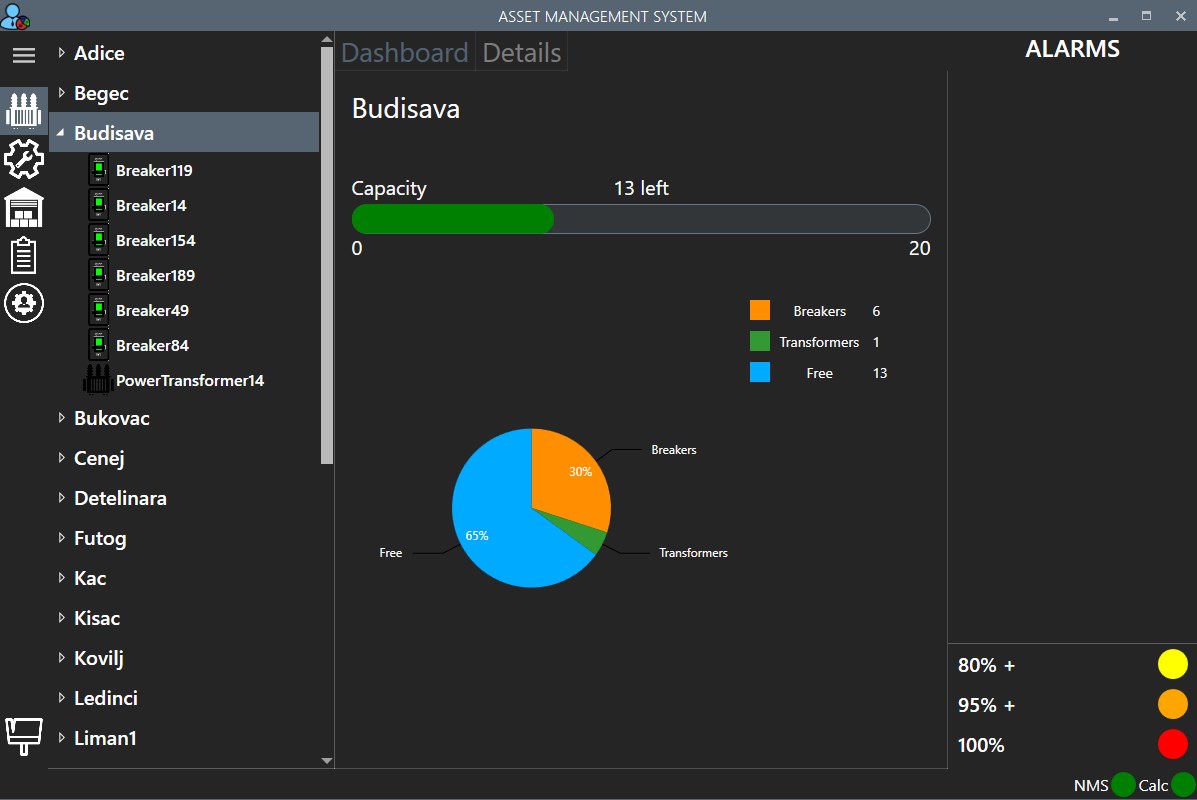
* **GID** - *globalni identifikator*
* **ExpirationTime** - *rok trajanja*
* **ServiceTime** - *rok za odlazak na servis*
* **EquipmentContainer** - *referenca na kontejner u kom se nalazi*
* **CatalogName** - *naziv kataloga*
* **Price** - *cena*
* **Name** - *ime*

Inicijalne podatke u sistem unosimo preko **.xml** fajla, koji je prethodno kreiran i popunjen.

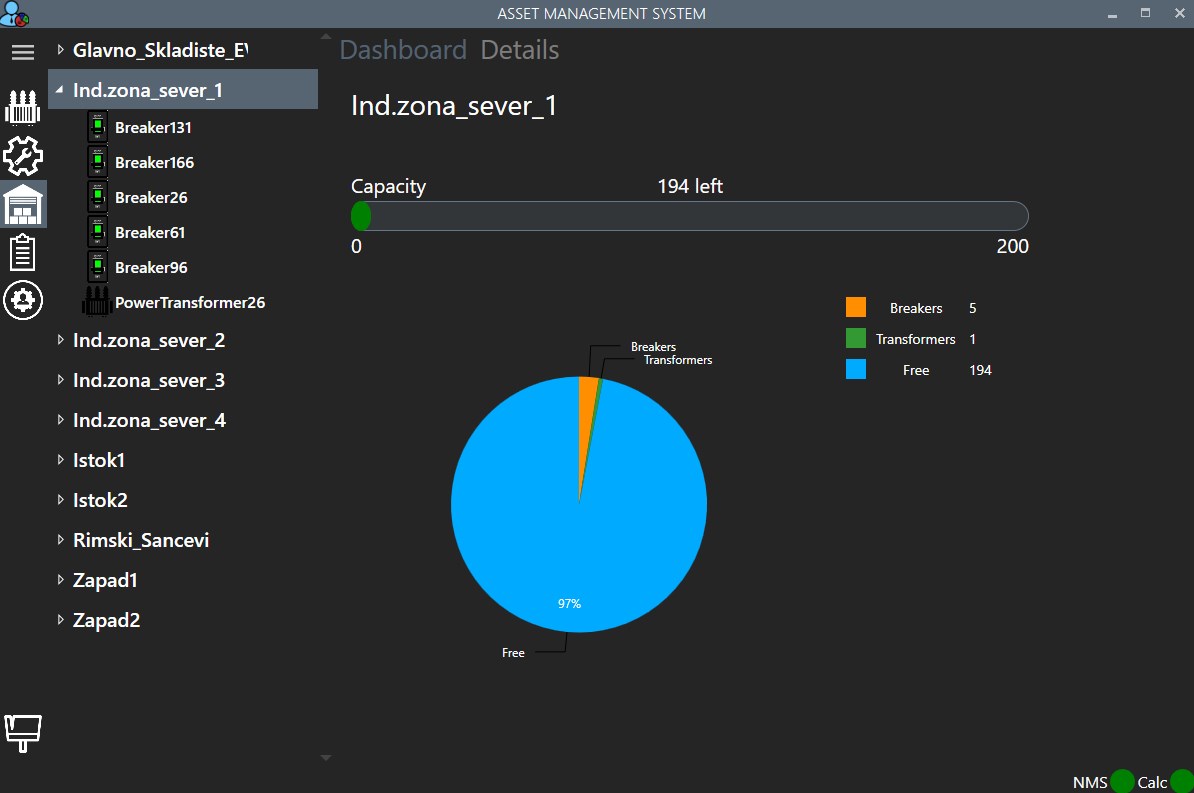
U svakoj stanici (*substation*) potrebno je da u svakom momentu bude minimalno **2 prekidača i 1 transformator**, u suprotnom će prikazati grešku.

Svi elementi počinju svoj vek u **Magacinu** (*warehouse*), nakon čega je potrebno da preko **Asset** **Manager**-a izvršimo premeštanje nekog elementa na **Stanicu** (*substation*).

U momentu kada je potrebno da dati element ide na servis, preko **Asset Manager**-a prebacujemo dati element na mesto gde se vrše **popravke** (*maintenance*). Element možemo da popravimo klikom na dugme *Repair*, pri čemu će nakon određenog vremena (10s) biti moguće prebaciti element u magacin i sa njim dalje manipulisati.



***Slika6.2*** *Prikaz stanice (substation) sa detaljima - ime, trenutni kapacitet, elementi unutar*



***Slika6.3****Prikaz magacina (warehouse) sa detaljima - ime,trenutni elementi i kapacitet sa grafičkim prikazom*