



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД**

Департман за рачунарство и аутоматику

Одсек за рачунарску технику и рачунарске комуникације

ЗАВРШНИ (BACHELOR) РАД

Кандидат: Елеонора Нан

Број индекса: РА180/2013

Тема рада: Управљање паметном кућом уз помоћ *Google* асистента

Ментор рада: проф. др Иштван Пап

Нови Сад, Јул, 2017



УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР:	
Идентификациони број, ИБР:	
Тип документације, ТД:	Монографска документација
Тип записа, ТЗ:	Текстуални штампани материјал
Врста рада, ВР:	Завршни (Bachelor) рад
Аутор, АУ:	Елеонора Нан
Ментор, МН:	Проф. др Иштван Пап
Наслов рада, НР:	Управљање паметном кућом уз помоћ Google асистента
Језик публикације, ЈП:	Српски / латиница
Језик извода, ЈИ:	Српски
Земља публикавања, ЗП:	Република Србија
Уже географско подручје, УГП:	Војводина
Година, ГО:	2017
Издавач, ИЗ:	Ауторски репринт
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, ФО: (поглавља/страница/ цитата/табела/слика/графика/прилога)	7/36/9/2/18/0/0
Научна област, НО:	Електротехника и рачунарство
Научна дисциплина, НД:	Рачунарска техника
Предметна одредница/Кључне речи, ПО:	IoT, кућна аутоматизација, гласовна команда, Гугл асистент
УДК	
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, ВН:	
Извод, ИЗ:	Већина постојећих алата за препознавање говора генеришу резултат у облику слободног текста. У циљу коришћења тог резултата за контролу уређаја гласовним командама у постојећем систему кућне аутоматизације, неопходно је детектовати шаблоне који одговарају информацијама као што су називи уређаја, њихове локације и називи команди. Стога је неопходно конвертовати слободан текст у структуриран објекат. У овом раду је представљено једно решење рашчлањивања текста коришћењем парсер генератора у циљу контроле „паметне“ куће, а уз помоћ Гугл асистента као алата за гласовно препознавање.
Датум прихватања теме, ДП:	
Датум одбране, ДО:	
Чланови комисије, КО:	Председник: проф. др Илија Башичевић
	Члан: доц. др Миодраг Ђукић
	Члан, ментор: проф. др Иштван Пап
	Потпис ментора



KEY WORDS DOCUMENTATION

Accession number, ANO :	
Identification number, INO :	
Document type, DT :	Monographic publication
Type of record, TR :	Textual printed material
Contents code, CC :	Bachelor Thesis
Author, AU :	Eleonora Nan
Mentor, MN :	Istvan Pap, PhD
Title, TI :	Smart Home Control with Google Assistant
Language of text, LT :	Serbian
Language of abstract, LA :	Serbian
Country of publication, CP :	Republic of Serbia
Locality of publication, LP :	Vojvodina
Publication year, PY :	2017
Publisher, PB :	Author's reprint
Publication place, PP :	Novi Sad, Dositeja Obradovica sq. 6
Physical description, PD : <small>(chapters/pages/ref./tables/pictures/graphs/appendixes)</small>	7/36/9/2/18/0/0
Scientific field, SF :	Electrical Engineering
Scientific discipline, SD :	Computer Engineering, Engineering of Computer Based Systems
Subject/Key words, S/KW :	IoT, home automation, voice control, Google Assistant
UC	
Holding data, HD :	The Library of Faculty of Technical Sciences, Novi Sad, Serbia
Note, N :	
Abstract, AB :	<p>Most of the existing speech recognition engines provide the results of the speech processing as the free form textual output. In order to use this output to enable voice commands within the existing home automation systems, it is necessary to detect patterns corresponding to the device names, their locations and actions that should be executed. Therefore, it is necessary to convert a free form textual output of a voice recognition services to a structured form. This paper presents one solution for parsing text in purpose of voice command for home automation using parser generator and with help of Google Assistant.</p>
Accepted by the Scientific Board on, ASB :	
Defended on, DE :	
Defended Board, DB :	President: Ilija Basicovic, PhD
	Member: Miodrag Djukic, PhD
	Member, Mentor: Istvan Pap, PhD
	Mentor's sign

Zahvalnost

Prvenstveno se zahvaljujem svojoj porodici na potpunoj podršci tokom školovanja. Hvala profesorima, asistentima i saradnicima na stečenom znanju. Hvala Mariji Antić i mentoru Ištvanu Papu na pruženoj pomoći tokom izrade ovog rada.

Posebnu zahvalnost upućujem kolegama iz tima na zajedničkoj saradnji prilikom razvoja projekta.



УНИВЕРЗИТЕТ У НОВОМ САДУ ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



SADRŽAJ

1.	Uvod	1
1.1	Pojam pametne kuće, IoT	1
1.2	Računarstvo u oblaku.....	2
2.	Teorijske osnove.....	3
2.1	Google Cloud Platforma	3
2.1.1	Google	3
2.1.2	Google Asistent (GA).....	3
2.1.3	Actions on Google (AoG)	5
2.1.3.1	Princip rada	5
2.1.3.2	Akcije	6
2.1.3.3	Kreiranje projekta.....	9
2.1.3.4	Validacija akcije.....	9
2.2	OBLO sistem	10
2.2.1	Centralni uređaj (OHM)	11
2.2.2	OBLO Cloud	11
2.2.3	Klijentska aplikacija	12
2.3	Parser generator	12
2.3.1	ANTLR parser generator	12
3.	Koncept rešenja	14
3.1	Ugrađeni moduli	14
3.1.1	Modul za prihvatanje glasovne komande u formi teksta.....	14
3.1.2	Modul za izvršavanje komande	15
3.1.3	Servis za parsiranje teksta	15

4. Programsko rešenje.....	17
4.1 Generisanje akcijskog paketa i otpremanje na AoG platformu	17
4.1.1 Komunikacija sa GA	18
4.1.1.1 Zahtev.....	18
4.1.1.2 Odgovor.....	19
4.2 Implementacija parser generatora	20
4.2.1 Definisane šablona podržanih komandi	20
4.2.2 Gramatika	21
4.2.2.1 Leksička pravila	21
4.2.2.2 Pravila parsera.....	22
4.3 Prilagođavanje generisanog parsera.....	23
5. Rezultati.....	25
6. Zaključak	26
7. Literatura	27

SPISAK SLIKA

Slika 1. Izgled GA aplikacije u pametnom telefonu	4
Slika 2. Google Home uređaj	4
Slika 3. Princip rada GA u integraciji sa GoA.....	6
Slika 4. Primer akcijskog paketa.....	7
Slika 5. Popunjavanje neophodnih podataka o autentikaciji u okviru kreiranog projekta na AoG platformi	8
Slika 6. Grafički prikaz AoG platforme	9
Slika 7. Izgled veb simulatora.....	10
Slika 8. Prikaz OBLO sistema za kućnu automatizaciju	11
Slika 9. Primer popunjavanja leksičkih pravila	13
Slika 10. Primer popunjavanja pravila parsera	13
Slika 11. Arhitektura sistema za glasovno upravljanje “pametnom” kućom	16
Slika 12. Akcijski paket prilagođen potrebama sistema	18
Slika 13. Format zahteva od strane GA.....	19
Slika 14. Format očekivanog odgovora za GA	19
Slika 15. Programski tok generisanja parsera.....	20
Slika 16. Primer leksičkih pravila korišćenih u implementaciji	22
Slika 17. Primer kratke gramatike	23
Slika 18. Izgled izlaznog objekta za LightIntent	24

SPISAK TABELA

Tabela 1. Definisani šabloni korišćenja.....	21
Tabela 2. Rezultati testiranja	25

SKRAĆENICE

ANTLR – *Another Tool for Language Recognition*, Alat za analiziranje jezika

SaaS – *Software as a Service*, Oblak kao servis

PaaS – *Cloud Platform as a Service*, Oblak kao platforma

IaaS – *Cloud Infrastructure as a Service*, Oblak kao infrastruktura

AI – *Artificial Intelligence*, Veštačka inteligencija

GA – *Google Assistant*, Gugl asistent

GH – *Google Home*, Uređaj integrisan sa Gugl asistentom

AoG – *Actions on Google*, Akcije na Guglu

IDE – *Integrated Development Environment*, Integrirano okruženje za razvoj

OHM - *OBLO Home Manager*, Centralni uređaj

JSON – *JavaScript Object Notation*, Standard za tekstualni opis podataka

IoT – *Internet of Things*, Internet stvari

HTTPS – *Hypertext Transfer Protocol Secure*, Komunikacioni protokol za sigurnu komunikaciju

1. Uvod

1.1 Pojam pametne kuće, IoT

Napredak moderne tehnologije omogućio je laku integraciju „pametnih“ uređaja u svakodnevni život. Ritam života ljudi postaje dinamičniji i brži, samim tim tehnološki noviteti se koriste u olakšavanju svakodnevnice. Internet stvari (engl. *Internet of Things*) predstavlja koncept povezivanja namenskih uređaja u postojeću mrežnu infrastrukturu. On omogućava povezivanje raznih sistema zasnovanih na računaru i možda jednog dana i apsolutnu povezanost između fizičkog i digitalnog sveta.

Jedan od najzanimljivijih napredaka u kućnoj automatizaciji jeste povećana upotreba glasovnih komandi za kontrolu svih vrsta namenskih uređaja. Očekuje se da kontrola glasom odigra ključnu ulogu u budućem razvitku *IoT*-a.

Pametna kuća predstavlja koncept modernog domaćinstva. Kuća se smatra „pametnom“ jer ume da se prilagodi članovima i automatizovana je. Komponente iz kojih se sistem pametne kuće najčešće sastoji su:

- Aktuatori
- Senzori
- Centralni uređaj – zadužen za kontrolu

Na tržištu postoji dosta tehnologija koje pružaju mogućnost korišćenja glasovnih komandi, kao što su biblioteka *Pocketsphinx* [2]-[3] i *Google Speech API* [1]. Oni pružaju *API* uz pomoć koji se vrši lako integrisanje u sistem.

Nakon detaljnog istraživanja dostupnih alata, najbolje rezultate pokazali su *Google* asistent (*GA*) i *Amazon Alexa* – virtuelni asistenti koji pored osnovnih, ugrađenih funkcionalnosti obezbeđuju mogućnost razvoja sopstvenih funkcionalnosti.

U daljem radu biće predstavljeno jedno rešenje integracije *Google* asistenta (*GA*) u postojeći sistem za automatizaciju kuće. Glavnu komponentu sistema čine servisi odgovorni za kontrolu dostupnih uređaja. S obzirom na specifičnosti koje sa sobom nosi *GA*, potrebno je implementirati parser koji će izvršiti detaljnu analizu zahtevane komande i raščlaniti je na neophodne delove.

1.2 Računarstvo u oblaku

Računarstvo u oblaku (engl. *Cloud Computing*) predstavlja isporuku računarskih resursa kao usluge grupi krajnjih korisnika. Koncept se oslanja na deljenje resursa putem mreže, najčešće interneta. Korisnici su u mogućnosti da pristupe aplikacijama unutar oblaka putem pretraživača, ali i preko aplikacija na mobilnim telefonima i drugim “pametnim” uređajima, gde se softver i korisnički podaci nalaze na serverima na udaljenoj lokaciji.

Postoje tri osnovna tipa računarstva u oblaku:

- Softver kao usluga, *SaaS*
- Platforma kao usluga, *PaaS*
- Infrastruktura kao usluga, *IaaS*

2. Teorijske osnove

2.1 Google Cloud Platforma

2.1.1 Google

Google je američka kompanija koja se bavi internet servisima i proizvodima, koji uključuju reklamne tehnologije, pretragu, skladištenje i softver. Deo usluga koje nudi nalaze se u okviru *Google Cloud* platforme [4] u obliku *IaaS*, *PaaS* i *SaaS* servisa. Ponuda servisa može se podeliti u četiri velike grupe:

- Usluge izvršavanja servisa na *Google* poslužiteljima
- Usluge skladištenja
- Mrežne usluge
- Usluge obrade podataka (engl. *Big Data*)

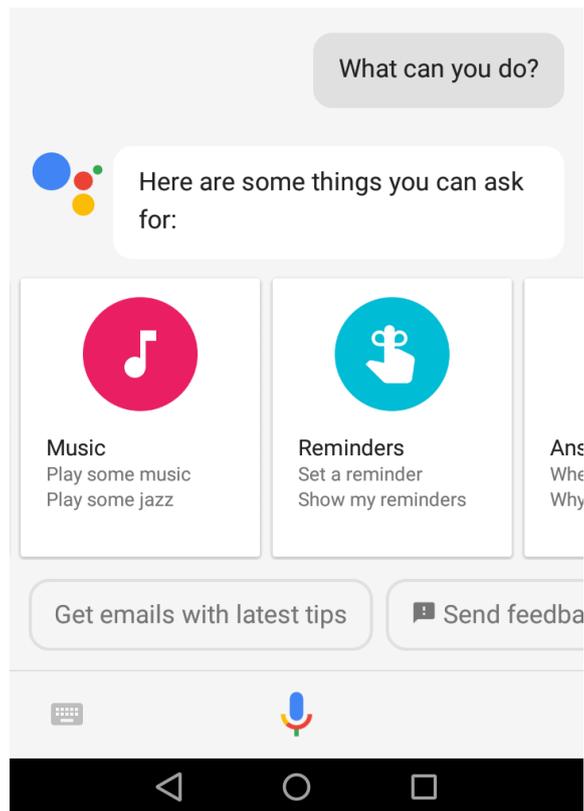
Lista dostupnih servisa [5] unutar ovih grupa je zaista velika i proširuje se svakim danom.

2.1.2 Google Asistent (GA)

Google asistent (*GA*) je virtualni, personalni *AI* (*Artificial Intelligence* – Veštačka inteligencija) asistent, predstavljen u maju 2016. godine. Ujedno i jedan od servisa unutar *Google* oblaka i smatra se *Google Now* nadogradnjom. Pruža mogućnost dvosmerne komunikacije – u vidu realne konverzacije. Korisnik može da interreaguje sa *GA* uz pomoć glasovnih komandi, gde asistent koristi algoritme za procesiranje prirodnog jezika, ali i unosom teksta sa tastature. Kao i njegov prethodnik *Google Now*, *GA* pretražuje Internet, zakazuje sastanke, podešava alarme, prilagođava hardversku konfiguraciju korisničkog uređaja, i prikazuje i koristi informacije o korisniku putem *Google* naloga. Pored osnovnih

funkcionalnosti, jedna od novina jeste upravo dvosmerni tok razgovora, gde pitanja i odgovori mogu da se nadovezuju bez ograničenja, na istu temu.

Ovaj proizvod prvobitno je plasiran samo u *Google Pixel* i *Pixel XL* telefonima, a zatim u *Google Home* uređaju, o kome će biti reči u daljem tekstu. Nakon kraćeg vremena, integrisan je i u Android uređaje sa operativnim sistemima: *Marshmallow*(6.0 & 6.0.1) i *Nougat*(7.0) uključujući i *Android Wear 2.0*. Još jedna od mogućnosti jeste integracija GA u sopstveni hardver korišćenjem *Google Assistant SDK*-a [6].



Slika 1. Izgled GA aplikacije u pametnom telefonu



Slika 2. *Google Home* uređaj

Kako bi omogućili što prirodiju interakciju sa ovim servisom, uvedeno je prepoznavanje ključnih reči “*OK Google*” i “*Hey Google*”. Stoga nije neophodno da servis bude konstanto aktivan, već samo nakon prepoznavanja “reči buđenja”. Nakon detektovanja iste, započinje se snimanje glasovne komande koja dalje biva procesirana pomoću pomenute *AI*. Dobijeni odgovor se reprodukuje u zavisnosti od uređaja posredstvom kojeg se *GA* koristi: u slučaju *Google Home* uređaja odgovor je u obliku audio zvuka, dok kod korišćenja pametnog telefona dobija se dodatno i tekstualni zapis istog. Trenutno je podržano prepoznavanje samo engleskog jezika.

2.1.3 Actions on Google (AoG)

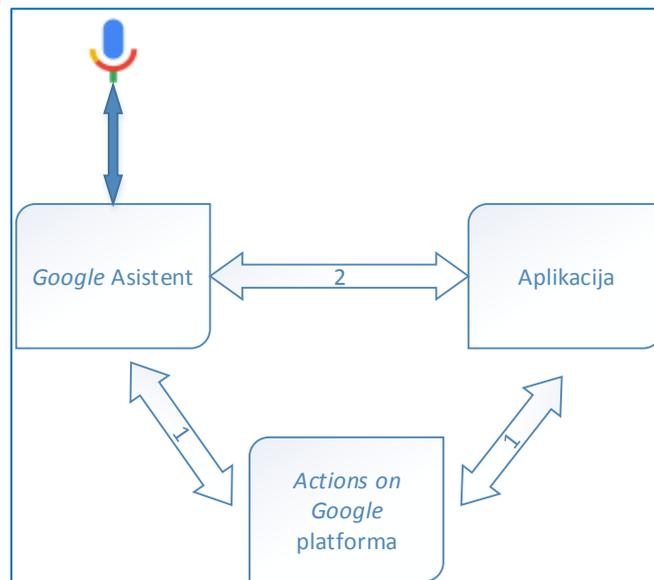
U decembru 2016. godine predstavljena je *Actions On Google* (Akcije na Guglu), programerska platforma za *GA*, što dodatno poboljšava korisničko iskustvo omogućavajući programerima da svoje akcije pridruže već postojećem nizu istih u okviru *GA*. Svaka akcija koja uspešno prođe verifikaciju postaje dostupna svim korisnicima, koji uz pomoć predefinisiranog naziva akcije mogu da je aktiviraju. Jedna od mogućnosti jeste implementacija sopstvene aplikacije koja realizuje željenu akciju.

2.1.3.1 Princip rada

Procesi koji se redom izvršavaju prilikom korisničkog zahteva za određenom akcijom su sledeći:

1. *GA* potražuje od *AoG* platforme da pokrene željenu aplikaciju
2. *AoG* šalje zahtev na krajnju tačku izvršavanja (engl. *endpoint*) odabrane aplikacije i dobija odgovor koji prosleđuje do *GA*
3. *GA* prikazuje odgovor korisniku.
4. Na dalje, *GA* unos korisnika direktno prosleđuje aplikaciji i aplikacija direktno odgovara asistentu bez posredstva *AoG*.

Na slici 3 grafički je predstavljen princip rada. Strelica označena brojem 1 označava putanju podataka prilikom inicijalizacije akcije, dok broj 2 označava dalji tok konverzacije.



Slika 3. Princip rada GA u integraciji sa GoA

2.1.3.2 Akcije

Actions on Google (AoG) je platforma koja pruža mogućnost proširivanja personalnog GA dodavanjem sopstvenih servisa koji se nazivaju akcijama. Postoje dva tipa interfejsa za razvoj akcija:

1. API.AI

Najčešće korišćen interfejs za razvijanje konverzijskih aplikacija. S obzirom na to da je razumevanje i parsiranje ljudskog govora veoma težak zadatak, API.AI to uspešno odrađuje. On takođe obavlja i funkcionalnost *Action SDK*-a u lak za korišćenje *IDE* koji nudi pogodnosti kao što su generisanje i otpremanje akcijskih paketa.

2. Actions SDK

Koristi se ukoliko aplikacija sadrži kratke konverzije sa ograničenim mogućnostima unosa od strane korisnika. Ovakvi tipovi akcija najčešće ne zahtevaju robustno razumevanje jezika i tipično pokrivaju male/jednostavne slučajeve korišćenja (engl. *Use Case*). S obzirom na to da ovaj paket ne poseduje pogodnosti *IDE*, neophodno je samostalno kreirati akcijske pakete i otpremiti ih na svoje naloge, o čemu će i biti još reči u daljem tekstu.

Akcija predstavlja ulaznu tačku i definiše model za pronalaženje i pokretanje aplikacije. Deklariše se unutar *JSON* datoteke koja se naziva akcijski paket, koja se otprema na *Google Cloud Platformu* u okviru prethodno napravljenog razvojnog projekta. U njemu se definišu osnovne informacije o akciji (slika 4). Dva polja su neophodna: „*actions*“ i

„*conversations*“. Unutar prvog polja, definišu se jedna ili više akcija, a unutar njih korisničke namere (*intent*) – funkcionalnosti koje se pokreću u zavisnosti od korisničkog unosa. Svaka od korisničkih namera može imati sopstvenu krajnju tačku izvršavanja, bez potrebe da sve budu na istom. Obavezna polja kojima se definiše jedna korisnička namera jesu:

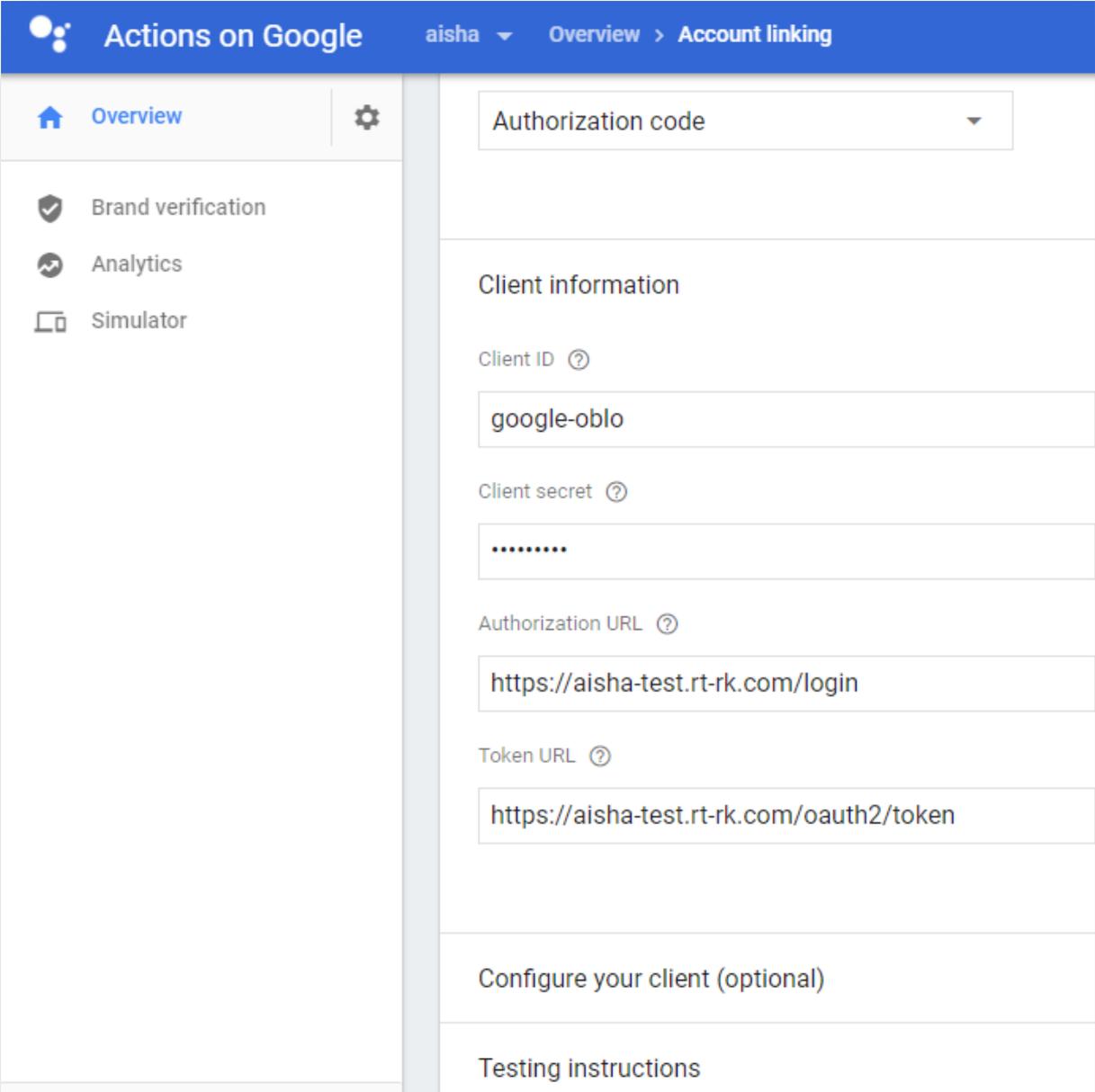
1. „*name*“ – jedinstven naziv korisničke namere
2. „*fulfillment*“ – polje koje sadrži URL na kom je dostupna aplikacija koja izvršava korisničku nameru
3. „*intent*“ – objekat koji sadrži informacije o korisničkoj nameri i šablone koji definišu rečenice koje će uzrokovati pokretanje akcije

```
{
  "actions": [
    {
      "name": "MAIN",
      "intent": {
        "name": "actions.intent.MAIN"
      },
      "fulfillment": {
        conversationName: "sekai-app"
      }
    },
    {
      "name": "BUY",
      "intent": {
        "name": "com.example.sekai.BUY",
        "parameters": [{
          "name": "color",
          "type": "SchemaOrg_Color"
        }],
        "trigger": {
          "queryPatterns": [
            {"queryPattern": "find some $SchemaOrg_Color:color sneakers"},
            {"queryPattern": "buy some blue suede shoes"}
          ]
        }
      },
      "fulfillment": {
        conversationName: "sekai-app"
      }
    }
  ],
  "conversations": {
    "sekai-app": {
      "name": "sekai-app",
      "url": "https://sekai.example.com/sekai-app"
    }
  }
}
```

Slika 4. Primer akcijskog paketa

Korisničke namere predstavljaju proste objekte koji bliže opisuju određen proces. U navedenom primeru, osnovna korisnička namera jeste *MAIN*, i pozvaće se prilikom prvog obraćanja aplikaciji. Druga definisana korisnička namera je *BUY*, pokrenuće se kada korisnički unos bude jedan od definisanih u polju „*queryPatterns*“.

Još jedna od mogućnosti jeste autentikacija – proces određivanja identiteta korisnika i povezivanje sa drugim nalogom. Postiže se dodavanjem polja „*signInRequired*“ u akcijski paket. Detalji autentikacije se upisuju prilikom kreiranja projekta na *AoG* platformi (slika 5).



The screenshot displays the 'Account linking' configuration interface. The top navigation bar includes the 'Actions on Google' logo, the user name 'aisha', and the current page path 'Overview > Account linking'. A left sidebar contains navigation options: 'Overview' (selected), 'Brand verification', 'Analytics', and 'Simulator'. The main content area is titled 'Authorization code' and contains the following fields:

- Client information**
 - Client ID: google-oblo
 - Client secret: [masked]
 - Authorization URL: https://aisha-test.rt-rk.com/login
 - Token URL: https://aisha-test.rt-rk.com/oauth2/token
- Configure your client (optional)
- Testing instructions

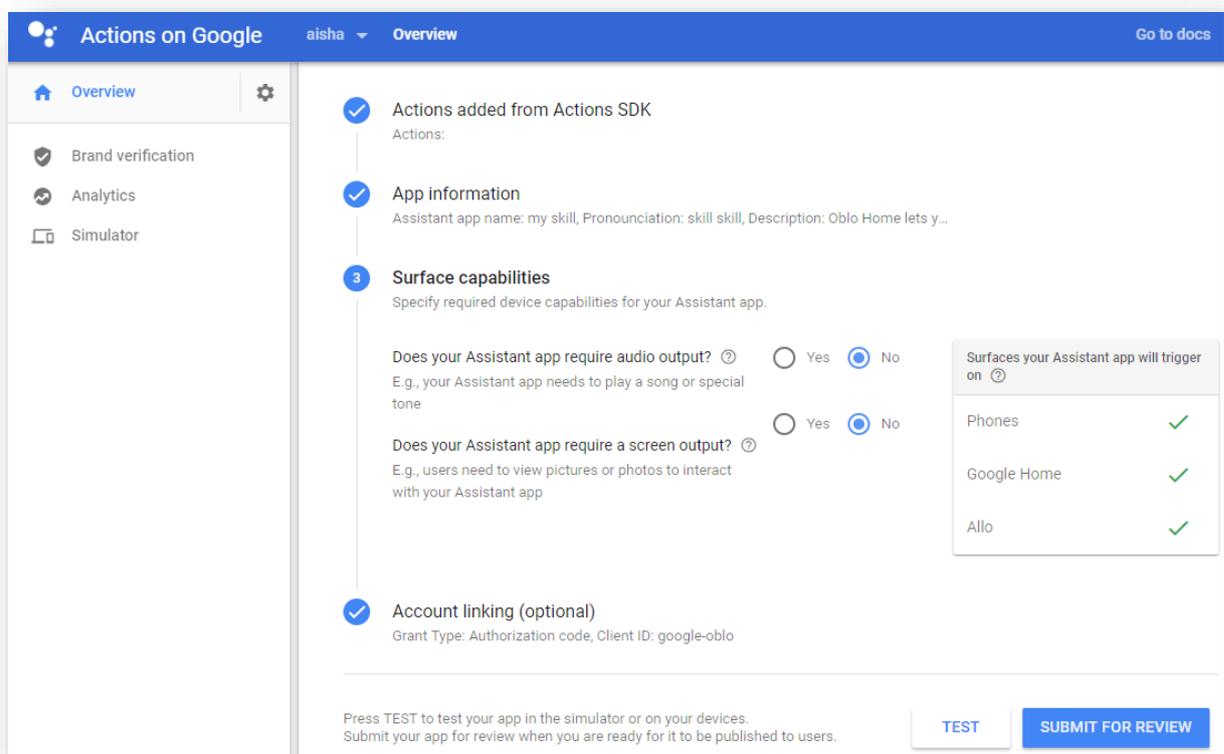
Slika 5. Popunjavanje neophodnih podataka o autentikaciji u okviru kreiranog projekta na *AoG* platformi

U jednom akcijskom paketu može da se definiše najviše deset različitih korisničkih namera. Neophodno je da *URL* koji se unosi u polju „*fulfillment*“ u okviru intenta koristi *HTTPS* protokol. Nakon kreiranja akcijskog paketa, potrebno je otpremiti isti na *AoG* platformu u okviru kreiranog projekta.

2.1.3.3 Kreiranje projekta

Osnovni zahtev za kreiranje projekta na *AoG* platformi je postojanje *Google* naloga. Kreiran projekat dozvoljava definisanje meta podataka o aplikaciji i praćenje procesa odobravanja akcije. Na samom početku potrebno je odabrati željeni interfejs (*API.AI* ili *Actions SDK*). Nakon odabiranja interfejsa, otpremljuje se akcijski paket na platformu. Ostatak informacija nije neophodno popuniti dok se ne pojavi potreba za javnom objavom akcije (engl. *Action Publishing*). Testiranje na simulatoru je moguće i bez objave.

Ime koje se odabira za akciju (engl. *Invocation Name*), a koje će ujedno biti i ključ za aktiviranje iste, trebalo bi da bude jasno i da se sastoji od najmanje dve reči. U primeru sa slike 7 *invocation name* je „my skill“.



Slika 6. Grafički prikaz *AoG* platforme

2.1.3.4 Validacija akcije

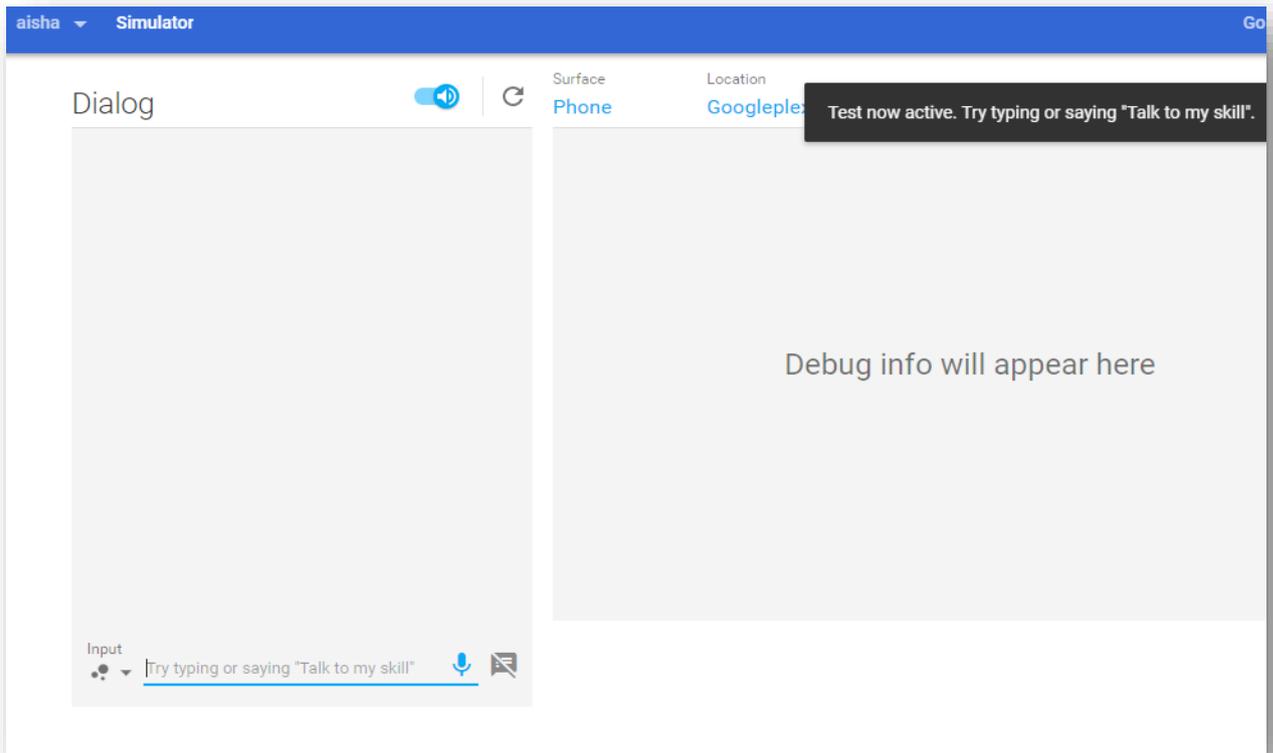
Testiranje akcije moguće je na tri načina: korišćenjem uređaja *Google Home*, pametnim telefonom sa Android sistemom novijim od 6.0 verzije, ili putem veb simulatora. Testiranje putem uređaja zahteva potpuno popunjavanje informacija o projektu, kako bi mogao da se iskoristi ključ za aktiviranje.

Rečenicama „*OK Google, talk to <imeAkcije>*“, „*OK Google, speak to <imeAkcije>*“ ili „*OK Google, I want to speak to <imeAkcije>*“ poziva se *MAIN* korisnička namera i

pokreće se akcija. Nadalje, u toku konverzacije, nije potrebno naglašavati željenu akciju. Sesija se prekida kada korisnik unese „*Goodbye*“.

U slučaju da nije neophodno održavati sesiju, već je dovoljno jedno obraćanje akciji, potrebno je uneti „*OK Google, ask <imeAkcije> to <željenaKomanda>*“. Željena komanda će se nalaziti unutar poslatog zahteva u obliku teksta.

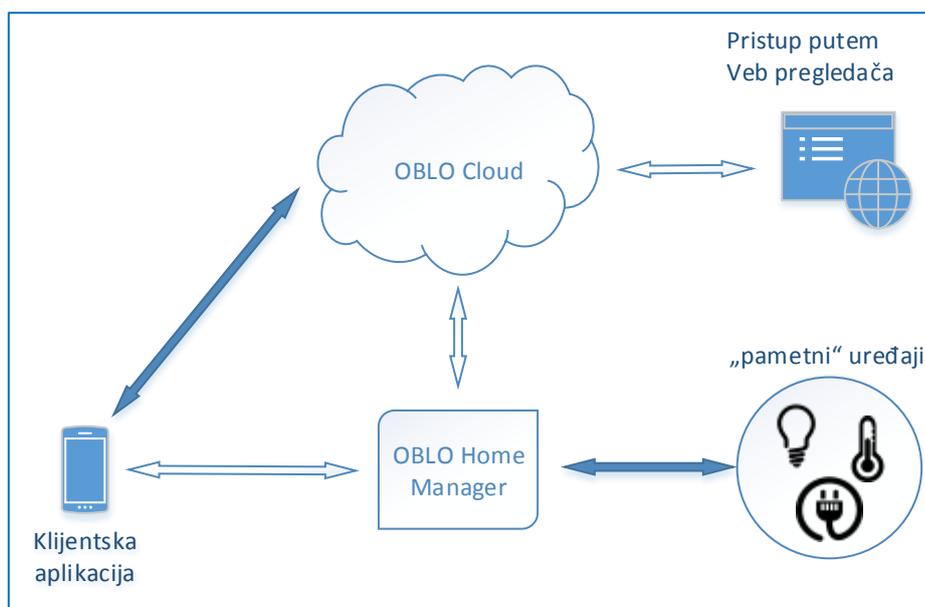
U veb simulatoru prikazuje se prozor sa *debug* informacijama. Prikazuje se poslat zahtev od strane GA, i odgovor koji je pristigao od aplikacije.



Slika 7. Izgled veb simulatora

2.2 OBLO sistem

OBLO sistem predstavlja sistem za kontrolu pametnih kuća. On se sastoji iz određenog broja nezavisnih komponenti koje komuniciraju putem različitih komunikacionih protokola. Komponente sistema su: centralni uređaj, OBLO *Cloud*, klijentska aplikacija i krajni “pametni” uređaji. Prikaz kompletnog sistema dat je na slici 8.



Slika 8. Prikaz OBLO sistema za kućnu automatizaciju

2.2.1 Centralni uređaj (OHM)

Centralni uređaj je softverska komponenta koja omogućuje funkcionalnost kućne automatizacije. Može da se izvršava ili na kontroleru kućne automatizacije (*gateway*) ili može da se integriše u okviru već postojećeg uređaja (npr. ruter, računar,...).

Njegov osnovni zadatak jeste da komande koje korisnik zadaje prosleđuje uređajima, i obratno, da informacije sa uređaja po zahtevu šalje korisniku. Osim osnovne funkcionalnosti, ima zadatak da izvršava dodatnu logiku, tj. da samostalno upravlja uređajima na osnovu predefinisanih zahteva (npr. scene). Bavi se i problematikom grupisanja uređaja na istu lokaciju/zonu radi grupnog kontrolisanja istih.

Kako bi uspešno izvršavao zadatke, centralni uređaj podržava određene komunikacione protokole: *ZigBee*, *ZWave*, *IP*, *BLE*. Omogućava i grupisanje uređaja na istoj lokaciji/zoni radi grupnog kontrolisanja. Komande od strane korisnika prihvata putem *MQTT* protokola [9].

2.2.2 OBLO Cloud

OBLO Cloud predstavlja spregu između centralnog kontrolera i korisnika u slučaju pristupa putem Interneta. Samim tim omogućava kontrolu “pametne” kuće van lokalne mreže. Komponente od kojih se sastoji jesu server aplikacije, baza podataka, *back-end* servisi i *front-end*. Grafička korisnička sprega *OBLO cloud*-a nudi mogućnost konfiguracije centralnog kontrolera, prijem obaveštenja i daljinsku kontrolu uređaja za kućnu automatizaciju.

Ovo programsko rešenje dodatno omogućava slanje obaveštenja korisniku putem elektronske pošte, SMS i *push* notifikacija.

2.2.3 Klijentska aplikacija

Posredstvom klijentske aplikacije korisnik vrši kontrolu svojih uređaja. Prilikom pokretanja povezuje se sa centralnim uređajem i dobavlja osnovne informacije o sistemu. Informacije sadrže dostupne uređaje i njihova stanja. Nakon uspešnog povezivanja dalja kontrola istih postaje dostupna.

Trenutno su podržane klijentske aplikacije za iOS i Android platforme.

2.3 Parser generator

Parser generator predstavlja alat koji olakšava kreiranje parsera. Parser ima zadatak da od određenog teksta izgradi organizovanu strukturu kao što je apstraktno sintaksko stablo (engl. *Abstract Syntax Tree*) koje se kasnije lako analizira.

2.3.1 ANTLR parser generator

ANTLR [8] predstavlja parser generator koji može da se koristi u cilju čitanja, obrade, izvršavanja ili prevođenja struktuiranog teksta ili binarnih datoteka. Zbog jednostavnog korišćenja našao je široku primenu. Korišćen je kako u akademskim ustanovama tako i u industriji za kreiranje programskih jezika, kompajlera, interpretera, raznih alata i radnih okruženja.

Princip rada zasniva se na generisanju dve klase: leksera i parsera na osnovu definisane gramatike. Gramatika sadrži leksička pravila (slika 9) i pravila parsiranja (slika 10) u obliku tokena koji su raspoređeni u različite šablone koji mogu da se očekuju.

Klasa lekser uzima individualne karaktere i transformiše ih u tokene koji su ujedno i ulazna tačka za parser. Na osnovu ulaznih tokena parser pronalazi izraz posredstvom pravila (šablona) i obavlja određene akcije za svaki detektovan. Svako pravilo koje je definisano u parseru korespondira jednoj akciji. Akcije su definisane kao klasne metode koje se pokreću prilikom detektovanja nekog pravila i omogućuju korisniku da obavi procesiranje. Korišćenje ovih metoda, koje se ponašaju kao slušaoci (engl. *listeners*), omogućio je *Tree Parser* koji je generisan uz pomoć gramatike koja je definisana kao *Tree grammar* (grananje u vidu krošnje drveta). To podrazumeva generisano sintaksko stablo koje pokazuje postupak izvođenja raščlanjivanja iz gramatike, gde svaki čvor stabla označava pojam ili simbol koji se pojavljuje u tekstu. Gramatika pruža mogućnost rekurzivnog definisanja šablona u smislu da jedno

pravilo može da se sastoji iz drugih pravila, a ta pravila se takođe mogu sastojati iz pravila nižeg nivoa, sve dok se ne dođe do samo određenog skupa reči.

```
/*  
*   Lexer Rules  
*/  
  
fragmentA   : ('A'|'a');  
fragmentB   : ('B'|'b');  
  
EXAMPLE     : E X A M P L E;  
  
WHITESPACE  : (' '|'\t');
```

Slika 9. Primer popunjavanja leksičkih pravila

```
/*  
*   Parser Rules  
*/  
  
primer1     : prviDeo | WHITESPACE | drugiDeo;  
prviDeo     : ( 'dobar' | 'bolji' );  
drugiDeo    : ( 'dan' | 'san' );
```

Slika 10. Primer popunjavanja pravila parsera

3. Koncept rešenja

Oslanjajući se na prethodne celine dolazimo do jednog od mogućih softverskih rešenja (slika 11) upravljanja pametnom kućom putem glasovnih komandi. U osnovi su neophodni servis za prepoznavanje govora i servis za izvršavanje naredbi. U zavisnosti od okruženja u kom se razvijaju servisi, rešenje je potrebno proširiti. Ovo rešenje u „oblaku“ zahteva i dodatni proces, tačnije međukorak koji bi priuštio dodatnu obradu naredbe ukoliko je to i potrebno.

S obzirom na to da različiti alati za prepoznavanje govora generišu različite izlaze, javila se potreba za raščlanjivanjem obrade kako bi se od određenog momenta izvršavanje komandi moglo svesti na isti proces. Servisi koji su implementirani u postojećem sistemu olakšavaju integrisanje bilo kog alata za glasovno prepoznavanje. Međusobno komuniciraju putem redova čekanja, a sve u cilju podrške višekorisničkog korišćenja. OBLO sistem u okviru komponente *OBLO Cloud* sadrži niz servisa neophodnih za realizaciju ovog sistema.

Postojeća arhitektura osmišljena je u cilju raspoređivanja zadataka između modula. Pruža mogućnost skaliranja modula uz pomoć implementiranih redova čekanja.

3.1 Ugrađeni moduli

3.1.1 Modul za prihvatanje glasovne komande u formi teksta

Predstavlja ulaznu tačku koja ima zadatak da pristigli zahtev u formi teksta uputi na dalju obradu. U zavisnosti od korišćenog alata za glasovno prepoznavanje, upućuje se ili u red čekanja za izvršavanje komande ili u red za dalje parsiranje.

Format dolazne poruke koji je podržan može da bude u vidu rečenice – u tom slučaju zahteva dodatno parsiranje. Idealni format jeste već rasčlanjena rečenica, čije semantičke celine nose informacije dovoljne za izvršavanje komande u sistemu pametne kuće.

3.1.2 Modul za izvršavanje komande

Neizostavan deo sistema za kućnu automatizaciju je servis koji vrši komunikaciju sa centralnim uređajem, izdaje komande i prima određene informacije o stanjima krajnjih uređaja.

Procesi koje izvršilac obavlja:

- Povezivanje sa centralnim uređajem u cilju dobavljanja liste dostupnih uređaja za kontrolu i izdavanja komandi.
- Validacija unosa od strane korisnika.
- Autentikacija u cilju povezivanja naloga i dobavljanja informacija o korisničkom centralnom uređaju, kako bi konekcija bila uspešno obavljena.
- Slanje odgovora servisu za glasovno prepoznavanje koje se reprodukuje korisniku u audio formatu.

3.1.3 Servis za parsiranje teksta

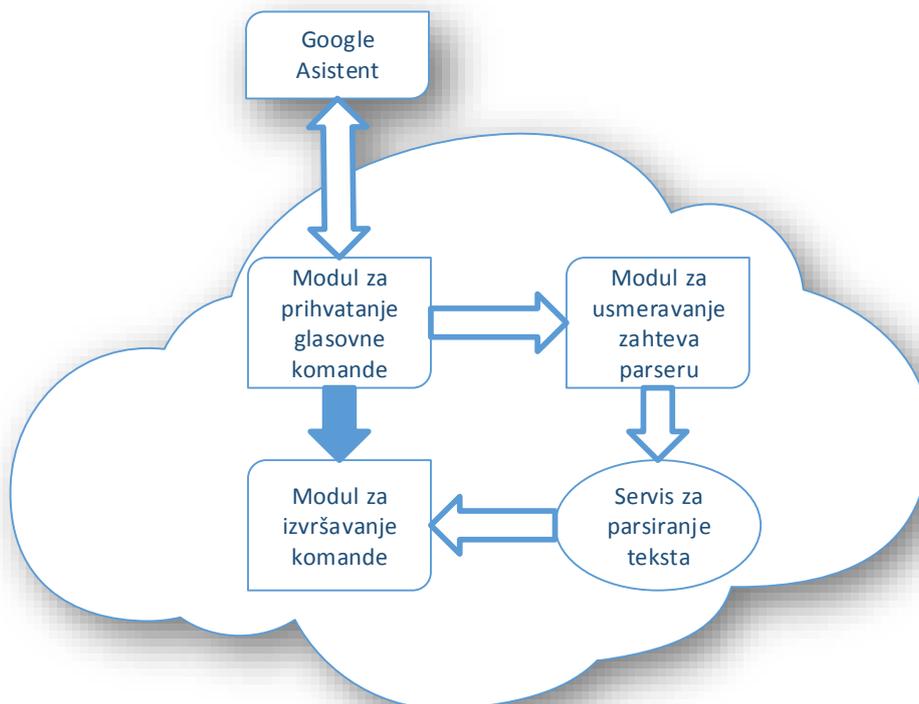
Servis za parsiranje teksta nalazi se u okviru implementiranog modula za usmeravanje zahteva ka parseru.

Većina postojećih alata za prepoznavanje govora generišu izlaz u obliku slobodnog teksta. U cilju uspešnog korišćenja generisanog teksta za glasovno upravljanje, neophodno je pronaći šablon koji sadrži akciju, naziv uređaja, njegovu lokaciju. S obzirom na to, potrebno je konvertovati slobodan tekst u struktuiranu formu sa određenom semantikom.

Implementiran servis predstavlja nezavisan modul. Za svrhu generisanja parsera, odabran je *ANTLR* parser generator. Neke od odlika jesu velika brzina obrade i realizovana podrška za Node.js platformu, što nam dodatno ollašava integraciju u postojeći sistem za automatizaciju kuće.

Prilikom dizajna sprege za kontrolu glasom, potrebno je unapred definisati šablone govora koji bi mogli da se prevedu u komandu. Skup tih šablona je prirodan za korisnika i sličan je ili isti svakodnevnom govoru. Sa druge strane, komande moraju da sadrže sve potrebne informacije neophodne za izvršavanje korisničkog zahteva. Na primer, svaka komanda mora da sadrži tip akcije (upali, ugasi, očitaj itd.). Jedinствeno definisanje uređaja može da se postigne dodavanjem lokacije ispred naziva, gde će se npr. kuhinjsko svetlo i trepezarijsko svetlo razlikovati tokom izvršavanja.

Izlaz koji je generisan od strane ovog servisa je strukturiran objekat koji je ulazna tačka za modul za izvršavanje komande.



Slika 11. Arhitektura sistema za glasovno upravljanje "pametnom" kućom

4. Programsko rešenje

Jedno od mogućih rešenja integrisanja *GA* u okviru sistema za automatizaciju pametne kuće predstavljeno je u prethodnom poglavlju. Kao alat za glasovno prepoznavanje odabrali smo *Google* asistenta, kako zbog brzine odziva i korektnosti prepoznavanja, tako i zbog formata konvertovanog teksta.

U nastavku poglavlja biće predstavljene implementirane komponente u okviru sistema.

4.1 Generisanje akcijskog paketa i otpremanje na AoG platformu

Akcijski paket u obliku *JSON* datoteke je od ključne važnosti prilikom implementacije ovog sistema. On sadrži informacije poput naziva akcije, opisa iste, šablone koji olakšavaju prepoznavanje govora ali i informacije o aplikaciji koja se pokreće prilikom poziva akcije. U ovom rešenju nije potrebno definisati šablone s obzirom na to da parser uspešno raščlanjuje komandu, samim tim akcijski paket jeste vrlo jednostavan (slika 13). Definisanje dodatnih korisničkih namera nije nužno jer je parser omogućio prepoznavanje istih, o čemu će biti reči prilikom definisanja šablona podržanih komandi.

Polje unutar datoteke koje obezbeđuje autentikaciju prilikom pokretanja je „*signInRequired*“. Ovaj proces se smatra veoma bitnim jer povezivanje *Google* naloga i naloga u *OBLO* sistemu obezbeđuje implementiranoj aplikaciji informacije o centralnom uređaju korisnika.

```
{
  "actions": [
    {
      "description": "Default Welcome Intent",
      "name": "MAIN",
      "fulfillment": {
        "conversationName": "oblo-voice-command"
      },
      "intent": {
        "name": "actions.intent.MAIN"
      },
      "signInRequired": true
    }
  ],
  "conversations": {
    "oblo-voice-command": {
      "name": "oblo-voice-command",
      "url": "https://aisha-test.rt-rk.com/vci/google"
    }
  }
}
```

Slika 12. Akcijski paket prilagođen potrebama sistema

4.1.1 Komunikacija sa GA

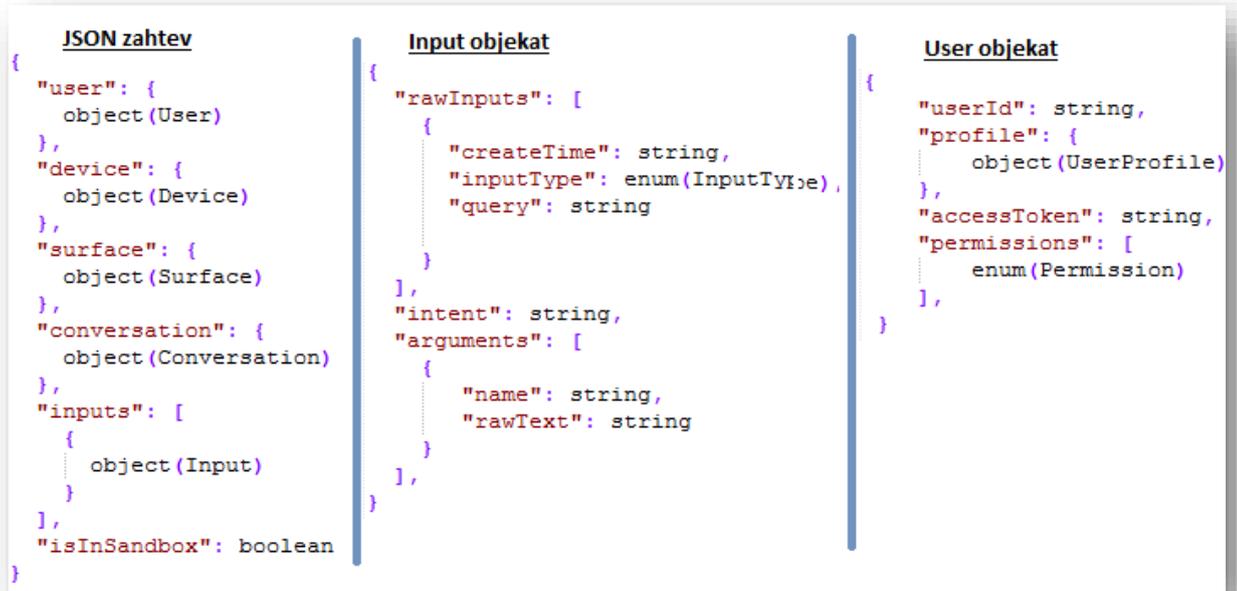
4.1.1.1 Zahtev

Komunikacija između *GA* i naše aplikacije se odvija putem *HTTPS* protokola. *GA* formira *JSON* objekat sa određenim poljima koji su nam neophodni. Osnovni objekti u zahtevu nalaze se na slici 14.

Polja koja su nama potrebna za realizaciju su „*user*“ i „*inputs*“ koja sadrže objekte *user* i *inputs*. Oni sadrže osnovne informacije u korisniku, kao i token koji je potreban za autentikaciju prilikom povezivanja sa centralnim uređajem.

Prepoznat tekst nalazi se u okviru objekta *user* i trenutno sadrži tekst u slobodnoj formi koji je spreman za parsiranje.

U slučaju korišćenja intenta i šablona za prepoznavanje, ostatak polja bi takođe bio od velike važnosti.



Slika 13. Format zahteva od strane GA

4.1.1.2 Odgovor

Odgovor poslat GA mora da zadovoljava određeni format. To bi bio predefinisani *JSON* objekat koji poseduje određena polja. Poštovanje izloženog formata omogućava reprodukciju željenog odgovora od strane GA. Dakle, ispravno strukturiran odgovor obezbeđuje mogućnost da korisnik dobije odgovor u obliku prirodnog ljudskog govora.

Za potrebe implementacije ovog rešenja dovoljno je definisati samo dva polja navedena na slici 15. Svakako, GA podržava mnoštvo drugih polja koje nismo naveli s obzirom na to da se ne koriste u ovom rešenju.

Odgovor koji se nalazi u okviru polja „*text_to_speech*“ generisan je u modulu za izvršavanje komandi. Takođe, taj modul je zadužen kako za prihvatanje zahteva, tako i za slanje odgovora do GA.

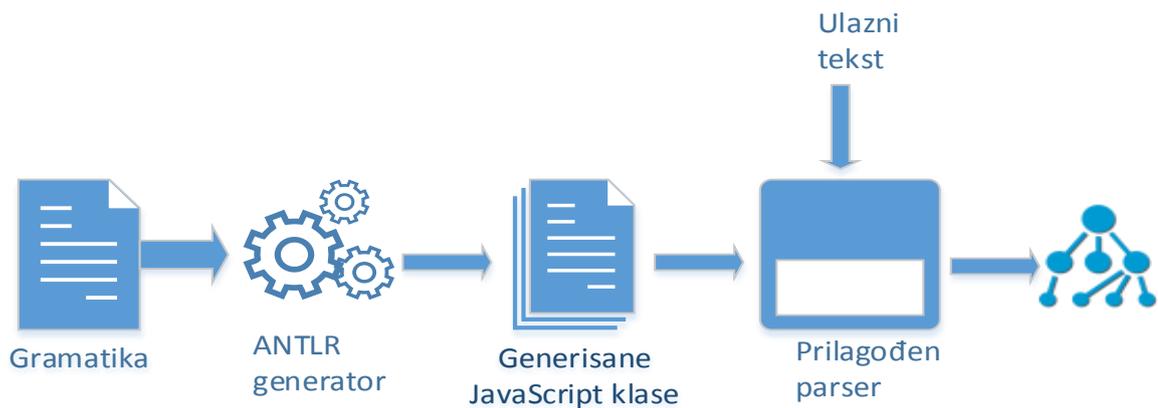
```

{
  "expect_user_response": false,
  "final_response": {
    "speech_response": {
      "text_to_speech": "The light is turned on."
    }
  }
}

```

Slika 14. Format očekivanog odgovora za GA

4.2 Implementacija parser generatora



Slika 15. Programski tok generisanja parsera

Programski tok generisanja parsera predstavljen je na slici 12. Ulazna tačka jeste gramatika koja detaljno opisuje izgled željenog parsera. Zatim *ANTLR* parser generator na osnovu ulaza produkuje izvorni kod u obliku *JavaScript* klasa.

Prilagođavanje parsera predstavlja primenu generisanog parsera. Postoji potpuna sloboda prilikom obrade dobijenih podataka. Cilj jeste da krajnji izlaz bude u obliku struktuiranog objekta koji se šalje na dalju obradu.

4.2.1 Definisane šablona podržanih komandi

Kako bi kontrolisao svoje „pametne“ uređaje, korisnik ima potrebu da koristi što prirodniji jezik za izgovaranje naredbi. Naredba treba da bude kratka i jasna, bez dodatnih stilskih ulepšavanja. U cilju olakšavanja i korisniku i projektantu sistema, napravljeni su skupovi šablona koji će biti podržani (tabela 1). Oni su neophodni kako bi gramatika mogla da se definiše.

Komanda se najčešće sastoji od tipa akcije i uređaja, gde je svaka akcija mapirana na određenu korisničku nameru, a sve u cilju olakšanog izvršavanja komandi. U Tabeli 1. prikazani su formati dozvoljenih naredbi. Ciljani uređaj opisan je imenom, dok opis lokacija označava poziciju unutar određene sobe, kako bi uređaji sa istim imenom mogli da se razlikuju (npr. stona lampa i podna lampa). Numerička vrednost opisana poljem vrednost označava željeni nivo npr. osvetljenja. Boja svetla se mapira na polje boja. Takođe, ukoliko korisnik želi da očita stanje nekog senzora, željeno stanje (npr. temperatura ili vlažnost vazduha) opisano je poljem stanje senzora. Za svako od navedenih polja podržano je oko pedeset reči što omogućava raznovrsno definisanje istih.

Obavezna polja koja se očekuju su naziv akcije i ime uređaja. Dakle, ukoliko oni ne postoje, sama naredba se smatra potpuno neispravnom, dok neke od naredbi zahtevaju i dodatna polja, kao što su npr. „promeni“ akcija, koja zahteva promenu boje na lampi i definisano polje boje.

Komanda	Primer korišćenja	Tip korisničke namere
Request + Device_Name	Turn on/off the light, Turn on/off the plug.	LightIntent
Request + Device_name + Floor+ Room+ Value	Set dimmer on second floor in kitchen to 20.	DimIntent
Request + Device_name+ Room + Color	Change light color in living room to red.	HueIntent
Request + Scene_name	Run relax scene.	SceneIntent
Request+ Sensor_service+ Device_name	Get temperature from kitchen sensor.	SensorIntent

Tabela 1. Definisani šabloni korišćenja

4.2.2 Gramatika

Gramatika predstavlja preciznu definiciju nekog jezika, ili u posmatranom slučaju definiciju pravila po kojima se formiraju moguće komande. Gramatika opisuje skup mogućih sekvenci koje formiraju validne reči i rečenice, ali ne definiše njihovo značenje. Ona se započinje startnim simbolom a potom se primenjuju pravila u bilo kojem broju, u bilo kojem redosledu.

4.2.2.1 Leksička pravila

Prva faza parsiranja predstavlja eksrtahovanje tokena, i u njoj se vrši leksička analiza kojom se ulazni niz karaktera deli u osnovne simbole čije je značenje definisano gramatikom leksičkih pravila.

```

/*
 * Lexer Rules
 */
fragment LOWERCASE : [a-z] ;
fragment UPPERCASE : [A-Z] ;
NUMBER              : [0-9] ;
WORD                : (LOWERCASE | UPPERCASE)+ ;
WHITESPACE          : (' ' | '\t')+ ;
NEWLINE             : ('\r'? '\n' | '\r')+ ;
TEXT                : ('[' | '(' | '~[\]]')+ ('[' | '(') ;

```

Slika 16. Primer leksičkih pravila korišćenih u implementaciji

Na primer, program za izračunavanje će tražiti ulaz kao što je $12 * (3 + 4)$ i podeliti ga na tokene 12, *, 3 i 4, od kojih svaki predstavlja simbol koji ima značenje u kontekstu aritmetičkog izraza. Parser treba da sadrži pravila po kojima će znati da slova * i + označavaju početak novog tokena.

U opisanom rešenju leksička pravila su korišćenja sa ciljem da se dozvoli i greška u unosu korisničke komande. Naime, ukoliko se pojavi reč koja predstavlja višak u naredbi, zanemaruje se, osim u slučaju da zahtev potpuni izgubi smisao.

4.2.2.2 Pravila parsera

Sledeća faza je sintaksna analiza, kojom se proverava da li otpremljeni tokeni formiraju izraz koji je dozvoljen. Dozvoljeni izrazi definišu se pravilima parsera koje rekruzivno definišu komponente koje mogu činiti izraz i redosled kojim one mogu da se pojavljuju.

Početni simbol definiše osnovnu komandu koju definiše niz intenta koji su predstavljeni u Tabeli 1. Sledeći nivo u okviru intenta sadrži naziv akcije, ime uređaja, naziv scene, sobe ili sprata. Definisano je da nije nužno da se sva polja moraju naći u izrazu, osim tipa akcije, bez koje nema smisla dalje parsiranje. Kratak primer osnovne gramatike koja zadovoljava minimalne potrebe dat je na slici 17.

```
grammar VoiceCommand;

/*
 * Parser rules
 */

command      :LightIntent|DimIntent;

LightIntent  :actionLight|nameOfDevice;

DimIntent    :actionDim|nameOfDevice|VALUE;

actionLight  :'turn on'|'turn off';

actionDim    :'set'|'dim';

VALUE        :NUMBER+;

nameOfDevice:'lamp'|'plug';

/*
 * Lexer rules
 */
NUMBER       : [0-9] ;
```

Slika 17. Primer kratke gramatike

Potrebno je omogućiti širok dijapazon mogućih naziva uređaja, lokacija, soba, scena. To podrazumeva zantno proširenje gramatike velikim brojem reči.

4.3 Prilagođavanje generisanog parsera

Prilagođavanje generisanog parsera jeste treći korak, koji predstavlja semantičko raščlanjivanje. Obrađuju se implikacije pronađenih izraza i preduzimaju se odgovarajuće radnje. Parser koji je generisan sastoji se iz *JavaScript* klasa, *Parser* i *Lexer*. U okviru njih postoje metode koje se pokreću prilikom prolaska kroz sintaksno stablo. Prolaženjem kroz sintaksno stablo parser proverava postojanje reči i izraza koji su definisani u gramatici.

Cilj uvođenja parsera je da pravilno raščlani naredbu i otpremi je u struktuiran objekat koji je unapred definisan. Kreiran je objekat sa određenim poljima koji se popunjava prolaskom kroz sintaksno stablo. Generisane metode su koncipirane tako da svaka može da se mapira na jedno pravilo gramatike, kao što su intenti, akcije, naziv uređaja.

<u>Izlazni objekat</u>	<u>Objekat Request</u>
<pre data-bbox="311 241 622 488"> { "session": { object(Session) }, "request": { object(Request) }, "version": "1.0" } </pre>	<pre data-bbox="821 253 1321 1099"> "type": "IntentRequest", "requestId": "", "locale": "en-US", "timestamp": "2017-06-30T11:16: "intent": { "name": "SceneIntent", "slots": { "Request": { "name": "Request" }, "Things": { "name": "Things" }, "Scene": { "name": "Scene", "value": "scene" }, "Rooms": { "name": "Rooms", "value": "bedroom" }, "SceneRequests": { "name": "SceneRequests", "value": "run" }, "Floors": { "name": "Floors" } } } </pre>
<p data-bbox="352 555 555 584"><u>Objekat Session</u></p> <pre data-bbox="311 600 683 936"> { "sessionId": "", "application": { "applicationId": "" }, "attributes": {}, "user": { "userId": "", "accessToken": "" }, "new": false } </pre>	

Slika 18. Izgled izlaznog objekta za LightIntent

Na primer, ako se detektuje LightIntent, generisaće se objekat sa poljima koji sadrže naziv akcije, ime uređaja, sobu, sprat. Sva polja ne moraju nužno da se popune, ukoliko ne postoje u izrečenoj komandi. Dakle, implementacija svake od metoda sastoji se samo iz popunjavanja određenog polja unutar izlaznog objekta. Primer objekta za LightIntent nalazi se na slici 18.

5. Rezultati

Za svrhu testiranja odabrano je pet različitih komandi na engleskom jeziku, izvršena su dva test slučaja. Svaka je ponovljena deset puta.

Prvi test slučaj svodio se na merenje odziva kompletnog sistema u zbiru sa obradom koja se izvršava van našeg dometa, tj. obrada koju vrši *Google* asistent. Cilj ovakve validacije jeste provera da li je vreme od momenta izgovaranja komande do izvršavanja, prihvatljivo za korisnika.

Drugi test slučaj je zadužen za testiranje brzine odziva parsera i izvršilaca komandi.

Neophodno je obratiti pažnju na kompleksnost sintakse jezika koji se koristi jer ona može da utiče na nivo prepoznavanja govora. Prosečno vreme odziva u prvom testnom slučaju jeste 2,25 sekundi, dok za samo izvršavanje prosek je 1,6 sekundi.

Rezultati su prikazani u tabeli 2. Predstavljene vrednosti su prosečna vremena po izvršenoj naredbi komandi.

Komanda	Prvi testni slučaj	Drugi testni slučaj
Turn on lamp.	2,06 sec	1,562 sec
Turn on all lights in ground floor.	2,98 sec	2,003 sec
Run relax scene.	1,23 sec	0,79 sec
Get temperature from sensor.	2,9 sec	2,089 sec
Set dimmer to fiftz.	2,08 sec	1,602 sec

Tabela 2. Rezultati testiranja

6. Zaključak

U ovom radu predstavljeno je jedno rešenje integracije *Google* asistenta u OBLO sistem za kućnu automatizaciju. Napredak tehnologije zahteva konstatno proširivanje postojećih rešenja a samim tim i *IoT* tehnologija – u smislu glasovne kontrole. Trenutno postojeća rešenja su najčešće ograničena na određen broj uređaja.

Rezultati pokazuju da korišćenje odabranog parser generatora olakšava integraciju različitih alata za glasovno prepoznavanje. Gramatika je uvek dostupna za brze i lake promene u vidu dodavanja novih leksičkih pravila i pravila parsiranja. Jedna od glavnih odlika ovog rešenja je mogućnost integracije bilo kog dostupnog alata za prepoznavanje govora, ukoliko nude izlaz u formatu slobodnog teksta.

Pokazano vreme odziva parsera je zadovoljavajuće, čak i neprimetno. Ono što utiče na brzinu parsiranja je dužina naredbe – što je komanda duža, prolaz kroz sintaksno stablo je kompleksnije a samim tim zahteva više vremena za procesiranje.

Dalji pravac unapređena usmeren je ka proširivanju korisničke sprege i dodavanju još podržanih komandi, kao i ka potpunoj integraciji u OBLO sistem.

7. Literatura

- [1] *Google Speech API* dokumentacija: <https://cloud.google.com/speech/docs>
- [2] *Pocketsphinx API* dok.: <http://cmusphinx.sourceforge.net/doc/pocketsphinx/>
- [3] C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatwy and D. Suendermann-Oeft, “Comapring Open-Source Speech Recognition Toolkits”, Technical Report of the Project OASIS, Germany, 2014.
- [4] Google Cloud Platform: <https://cloud.google.com/docs/overview>
- [5] Google Cloud Platform services: <https://cloud.google.com/products>
- [6] Google Assistant SDK: <https://developers.google.com/assistant/sdk>
- [7] Actions on Google: <https://developers.google.com/actions/components>
- [8] T. Parr, *The Definitive ANTLR 4 Reference*, Pragmatic Bookshelf, 2013
- [9] Milan Tucić, „Protokol za rezmenu poruka u sistemima pametnih kuća“, Master rad, 2016