

UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET U SARAJEVU
ODSJEK ZA TELEKOMUNIKACIJE

PROJEKTNI ZADATAK 1
Smart FastFood

Elma Adžanelić
Melisa Čehajić
Azra Topčagić

17.2.2019., Sarajevo

Sadržaj

Uvod.....	3
1. Teorijska obrada teme.....	4
1.1 SOAP (Simple Object Access Protocol) protokol	5
1.2 M2M (eng. Machine to Machine) protokol.....	6
1.3 SoapUI	7
1.4 Web servisi	7
1.5 Web aplikacija	8
1.6 MySQL	9
2. Izrada aplikacije.....	10
2.1 Pregled funkcionalnosti	10
2.1.1 Baza podataka	10
2.1.2 Web servisi	14
3. Radno okruženje, korišteni alati i tehnologije	15
3.1 Java	15
3.2 Eclipse	15
3.3 Apache server	16
3.4 Senzori	16
4. Softverski dizajn i implementacija aplikacije (arhitektura, UML dijagrami: dijagram paketa, dijagram klase, sekvencijalni dijagram).....	17
4.1 MSC (Message Sequence Chart) dijagrami.....	17
4.1.1 Dodavanje nove narudžbe	17
4.1.2 Dodavanje nove namirnice	18
4.2 Dijagram paketa	19
4.3 Dijagram klase	20
5. Interaktivni pregled aplikacije	23
Zaključak	34
Popis skraćenica	37
Literatura	39

Uvod

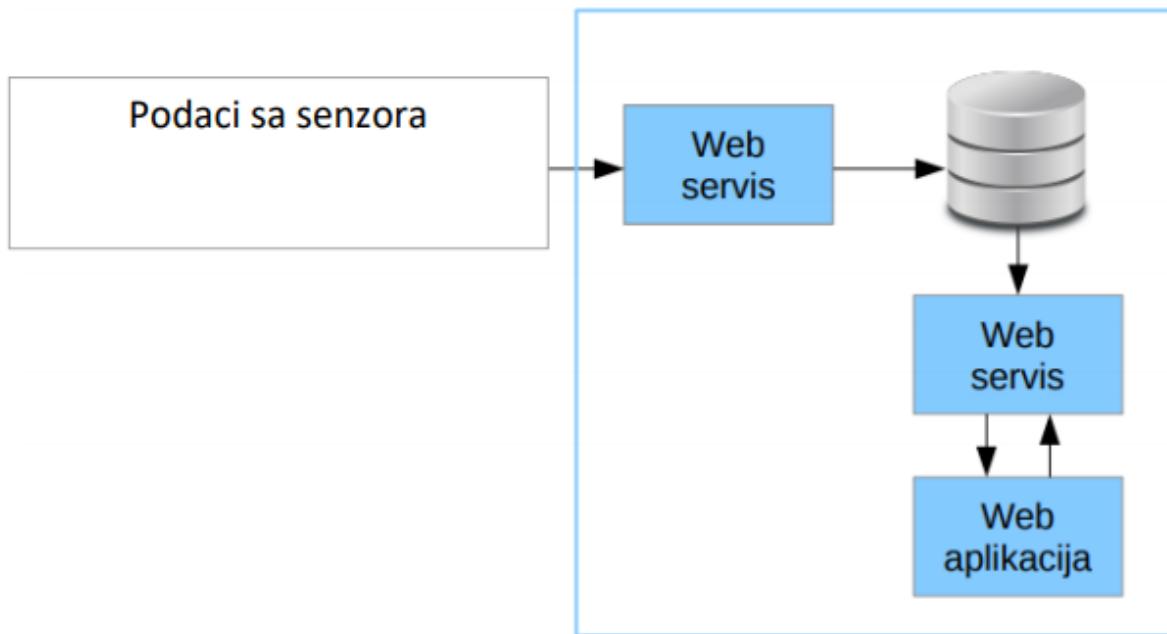
Projektni zadatak pod nazivom *SmartFastFood* predstavlja integraciju *smart* rješenja u funkcionalnu cjelinu, u kojoj su objedinjena teorijska i praktična znanja vezana za SOAP *web* servise, kojim će se simulirati vrijednosti senzora koji se prosljeđuju na *web* servis, *web* aplikacije koje se realiziraju kao *Java Spring* projekat, kao i korištena MySQL baza podataka.

U slučaju *SmartFastFood* rješenja, radi demonstracije različitih nivoa hijerarhije korisnika, potrebno je kreirati administrator korisnika (uposlenici), koji će na raspolaganju imati sve funkcionalnosti, te običnog korisnika, koji će imati reducirani set funkcionalnosti na raspolaganju. Konkretno, korisnik ima mogućnost izvršiti narudžbu, a mogućnost dodavanja namirnica, pregled korisničkih narudžbi, mogućnosti njihovog brisanja nakon dostavljanja istih, pregled vremenskog trenutka logiranih korisnika te nasilnog upada, kao i pregleda vrijednosti senzora, koji se očitavaju u prostorijama u kojima se čuvaju namirnice, kao i frižiderima i zamrzivačima, ima jedino administrator korisnik, a sve ove mogućnosti se nude zahvaljujući *web* servisima. Ostale funkcionalnosti i detaljniji način njihove realizacije će biti opisan u nastavku.

1. Teorijska obrada teme

U ovom poglavlju opisani su osnovni principi koji se tiču SOAP protokola, web servisa i web aplikacije. Također prikazana je osnovna arhitektura te su ukratko opisani entiteti koji učestvuju u samoj komunikaciji.

U sklopu projektnog zadatka implementirani su web servisi koji su vršili ažuriranje odnosno manipulaciju bazom podataka. Web servisi bazirani su na SOAP protokolu odakle potreba za detaljnijom analizom rada samog protokola. Entiteti koji učestvuju u realizaciji kompletног projektnog zadatka su web aplikacija, web servisi, baza podataka čiji je zadatak čuvanje podataka o narudžbama korisnika, dodanih namirnica te čuvanje informacija prikupljenih od strane senzora i SOAP UI. Soap UI (podaci sa senzora) simulira senzore koji prikupljaju informacije o temperaturi, nivou gasa, oksigena i zasićenosti prostorije, kao i temperaturi i vremenu radnih sati frižidera i hladnjаче, te poziva implementirane web servise koji vrše upis prikupljenih podataka u bazu.



Slika 1: Arhitektura

1.1 SOAP (Simple Object Access Protocol) protokol

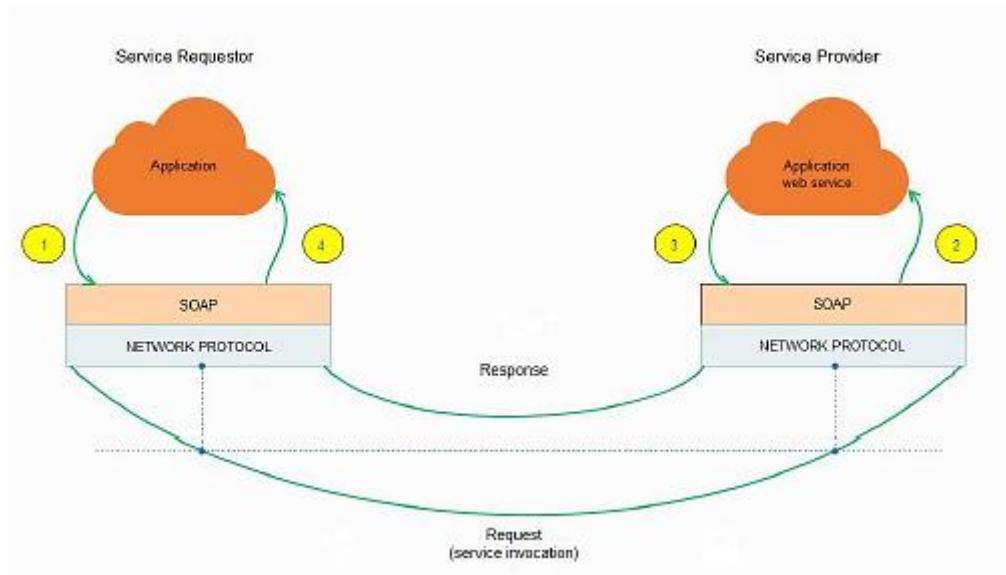
SOAP je komunikacijski protokol namijenjen za razmjenu strukturiranih podataka u distribuiranom okruženju. Koristi se XML-om za prenos strukturiranih poruka.

- SOAP protokol definiše [kako te poruke trebaju izgledati i kako ih prenijeti](#). SOAP ne definiše sadržaj i redoslijed razmjene poruka jer to zavisi o Web servisu koji ga koristi.
- SOAP protokol može biti [vezan uz protokole nižeg nivoa](#) poput SMTP (engl. Simple Mail Transport Protocol) ili TCP/IP protokol, iako se najčešće koristi vezan uz HTTP (engl. Hypertext Transfer Protocol) protokol.
- [HTTP protokol pridonosi jednostavnosti korištenja SOAP poruka](#) te omogućava slobodnu komunikaciju korištenjem porta 80 tako da se mogu izbjegići problemi propusnosti vezani za firewall-e.
- SOAP poruka je [XML dokument zadanog formata](#) koji može prenositi parametre koje treba predati programskom bloku na obradu ili rezultate obrade podataka.

SOAP poruka mora zadovoljavati nekoliko pravila te mora biti ispravno formatirani XML dokument i sastoji se od nekoliko elemenata [2]:

- [Omotnica](#) (engl. Envelope) koristi `xmlns:soap` područje imena (engl. namespace) uvijek mora imati vrijednost <http://www.w3.org/2001/12/soap-envelope>. Definiše okosnicu za određivanje onoga što se nalazi u poruci, ko je treba koristiti i kako.
- [Zaglavljje](#) (engl. Header) nosi odgovarajuće informacije zaglavljia. U zaglavljju se nalaze informacije o primjeni poruke, identifikatoru prenosa te podaci o pošiljaocu i primaocu poruke. Format i sadržaj podataka zavisi o vrsti SOAP poruke. Ako se ovaj element koristi unutar SOAP poruke, mora biti neposredno podređen Envelope elementu.
- [Tijelo](#) (engl. body) poruke s podacima o pozivu i odgovoru te porukama o grešci koje su se dogodile u toku obrade poruka. Porukama o greškama. Ako se tokom komunikacije dogodila nekakva greška, informacije o nastalim greškama bilježe se unutar Fault elementa. Kodovi grešaka su propisani.

Resursi se adresiraju primjenom URL-a koji vraća informaciju korisniku. Njihovo funkcionisanje se temelji na zahtjev-odgovor ([request-response](#)) paradigm gdje [request](#) i [response](#) mogu biti dio jedne operacije (*synchronous model*) ili se mogu pojaviti odvojeno. Podržavaju i P2P oblik komuniciranja. [1]



Slika 2: SOAP komunikacija[1]

1.2 M2M (eng. Machine to Machine) protokol

Tehnički, to je komunikacioni protokol za upotrebu između klijentskog softvera na M2M uređaju i serverskog softvera na M2M platformi za upravljanje i osposobljavanje usluga.

M2M protokol ima najmanje tri bitne osobine:

- 1) da ima **moderan dizajn** na osnovu REST pogodnosti za programere
- 2) da **definiše model resursa i podataka** koji se može proširivati
- 3) **dizajniran sa prepostavkom performansi i ograničenja M2M uređaja** i sa efikasnim standardom za bezbjedan prenos podataka *Constrained Application Protocol* (CoAP) koji je standardizovan od strane IETF kao varijacija HTTP protokola (prikladan za prenos podataka sa povezanih IOT uređaja).

Ovaj novi **klijent-server protokol** je predložen od strane *Open Mobile Alliance* (OMA), a zasniva se na protokolima i bezbjednosnim standardima IETF-a. M2M *Enabler* definiše komunikacioni protokol aplikativnog sloja između servera i klijenta. M2M server se obično nalazi u privatnom ili javnom *data centru*. M2M klijent se nalazi na uređaju i obično je integriran kao softverska biblioteka ili *built-in* funkcija modula ili uređaja. [1]

Četiri logička interfejsa su definisana između servera i klijenta [3]:

- 1) *Bootstrap Device Discovery*
- 2) *Registration*
- 3) *Device Management and Service Enablement*
- 4) *Information Reporting*

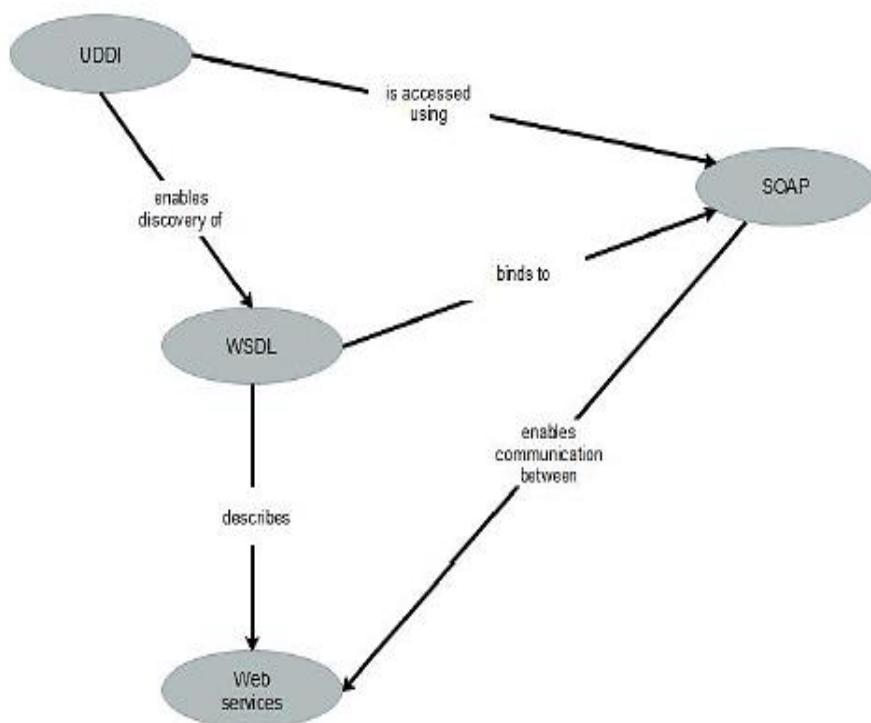
za bilo koju vrstu M2M slučajeva upotrebe.

1.3 SoapUI

SoapUI je *open-source* aplikacija za testiranje web servisa za servisno-orientisane arhitekture (SOA) i reprezentacijske transfere stanja REST (engl. *Representational State Transfer*). Njena funkcionalnost pokriva inspekciju, pozivanje, razvoj, simulaciju te razne vrste testiranja web servisa. SoapUI može testirati SOAP i REST web servise, JMS (*Java Message Service*), AMF (*Action Message Format*), te takođe može slati HTTP(s) i JDBC (*Java Database Connectivity*) zahtjeve. U kontekstu projektnog zadataka, ovaj alat se koristio da bi simulirao senzore koji mijere odgovarajuće parametre npr. vlažnost, gas, temperatura itd. [4]

1.4 Web servisi

Komunikacijski model (slika 3) se temelji na postojanju tzv. *service provider-a* i *service requestor-a* koji komuniciraju korištenjem SOAP protokola. Web servisni opisi interfejsa su dovoljno detaljni, u obliku XML dokumenata koji su pisani u WSDL jeziku. Prednosti web servisa u odnosu na klasične web aplikacije su **fleksibilnost, štedljivost i jednostavnost**. [5]



Slika 3: Web servisi – komunikacijski model [5]

Sloj protokola Web servisa se koristi da definiše, otkriva i implementira web servise.

Osnova ovih protokola leži u sljedeća četiri nivoa [5]:

- *ServiceTransport* - nivo odgovoran za prijenos poruka između aplikacija i u njega su trenutno uključeni HTTP, SMTP i FTP, te novi protokoli kao što je BEEP (engl. *Blocks Extensible Exchange Protocol*, skraćeno BEEP)
- *XMLMessaging* - nivo odgovoran za razumijevanje poruka koje se implementiraju u XML formatu i trenutno se koristi XML-RPC (engl. *XMLRemoteProcedureCall*, skraćeno XML-RPC) i SOAP
- *Service Description* - definiše javne interfejs za određeni Web servis i trenutno se opisuje kroz WSDL (engl. *WebService Description Language*)
- *Service Discovery* - centralizovanje servisa u zajednički i jedinstven registar koji osigurava jednostavno objavljivanje i pronalaženje servisa, a to se postiže pomoću protokola UDDI (engl. *Universal Description, Discovery and Integration*).

1.5 Web aplikacija

Web aplikacija koristi kreirane web servise da bi putem njih upisivala u bazu podataka, te da bi čitala podatke – odnosno pretraživala po ID-u i parametru *amount* unutar baze podataka. Dakle komunikacija se zasniva na sljedećem: podaci se upisuju u bazu i čitaju iz baze putem web servisa, a web aplikacija služi kao korisnički interfejs putem kojeg se zapravo koriste funkcionalnosti implementiranih web servisa. [6]

Za razvoj web interaktivne aplikacije korišten je *Spring* okvir. *Spring* okvir se zasniva na *model-view-controller* obrascu koji pojednostavljuje proces kreiranja interaktivnih web aplikacija. Umjesto manuelnog manipulisanja sa kompleksnostima *Servlet-a*, *HttpServletRequests-a*, *HttpServletResponses-a* te proslijđivanja ka JSP-ovima (*Java Server Pages*), *Spring* upravlja ovim zadacima. Svaka metoda u klasi *Controller* se mapira na različit URL zahtjev ili metod. Model se prenosi iz kontrolera ka pogledu u formi mape. *View* koji vraća *Controller* uzrokuje da *Spring* prikaže odgovarajući JSP pogled. Zahtjev i URL (*Uniform Resource Locator*) Path parametri se automatski konvertuju u primitivne ili kompleksne argumente metoda kontrolera. [6]

Kao dodatak tipičnim HTML (*Hypertext Markup Language*) pogledima, *Spring* može automatski da generiše jednostavne tekst preglede, poglede za download fajlova ili XML (*Extensible Markup Language*) odnosno JSON (*JavaScript Object Notation*) entitetske preglede. Kroz sve ove značajke, *Spring* okvir značajno pojednostavljuje rad u *Servlet* kontejneru.

1.6 MySQL

MySQL je popularan izbor baze podataka za korištenje u mrežnim aplikacijama, te u centralnoj komponenti naširoko korištene LAMP stog mrežne aplikacije otvorenog koda (i ostali 'AMP' stogovi). LAMP je akronim za *Linux, Apache, MySQL, Perl/PHP/Python*, besplatni projekti otvorenog koda koji zahtijevaju potpuno opremljen sistem upravljanja bazom podataka obično koriste MySQL. Na svim platformama osim *Windows-a*, MySQL se isporučuje bez GUI alata za administriranje MySQL baza podataka ili upravljanje podacima koji se nalaze unutar baza podataka. [7]

2. Izrada aplikacije

U ovom poglavlju je detaljno razmotreno okruženje korišteno za razvoj web aplikacije, te navedene funkcionalnosti koje nudi sama aplikacija. Način implementacije te softverski dizajn prikazan je kroz UML dijagrame.

2.1 Pregled funkcionalnosti

2.1.1 Baza podataka

Kreirana je baza podataka pod nazivom *smartdb*, koja se sastoji od šest tabela.

Tables_in_smartdb
foodstuffs
logins
orders
sensor_values
users
wrong_entry_attempts

Slika 4: Implementirane tabele u bazi podataka *smartdb*

Tabela *foodstuffs* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator korisnika
- *name* – naziv namirnice
- *type* – vrsta namirnice
- *amount* – količina namirnice(kg)
- *regDate* – prikazuje datum i vrijeme naručivanja namirnice

Tabela *logins* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator korisnika
- *username* – korisničko ime
- *login_time* – datum i vrijeme ulogovanja korisnika

Tabela *orders* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator narudžbe
- *typeoforder* – vrsta narudžbe
- *purchaser* – naručilac
- *contact* – kontakt telefon naručioca
- *note* – dodatne informacije o narudžbi

Tabela *sensor_values* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator senzora
- *name* – naziv senzora
- *type* – vrsta senzora (soba, frižider, hladnjaka)
- *temperature* – izmjerena temperatura
- *gas*
- *oxygenlevel*
- *humidity* – nivo zasićenosti
- *working_time_hours* – vrijeme rada frižidera ili hladnjake u periodu od godinu dana
- *regDate* – datum i vrijeme kada su izmjerene vrijednosti senzora

Tabela *users* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator običnog korisnika ili *admin* korisnika
- *username* – korisničko ime
- *password*
- *last_login_fail* – trenutak posljednjeg pogrešnog logovanja
- *login_fail_attempt* – broj neuspješnih login-a zaredom
- *is_admin* – označava da li se radi o običnom ili *admin* korisniku

Tabela *wrong_entry_attempts* se sastoji iz polja, koja su redom unesena na sljedeći način:

- *id* – predstavlja jedinstveni identifikator korisnika
- *username* – korisničko ime
- *timestamp* – vremenski trenutak izvršenja *brute force* napada

Opis svake tabele dat je u nastavku:

mysql> describe foodstuffs;					
Field	Type	Null	Key	Default	Extra
<i>id</i>	int(6) unsigned	NO	PRI	NULL	auto_increment
<i>name</i>	varchar(30)	NO		NULL	
<i>type</i>	varchar(30)	NO		NULL	
<i>amount</i>	int(11)	NO		NULL	
<i>regDate</i>	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Slika 5: Opis tabele *foodstuffs*

mysql> describe orders;					
Field	Type	Null	Key	Default	Extra
<i>id</i>	int(6) unsigned	NO	PRI	NULL	auto_increment
<i>typeoforder</i>	varchar(30)	NO		NULL	
<i>purchaser</i>	varchar(30)	NO		NULL	
<i>contact</i>	varchar(30)	NO		NULL	
<i>note</i>	varchar(30)	NO		NULL	
<i>regDate</i>	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Slika 6: Opis tabele *orders*

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
username	varchar(30)	NO		NULL	
login_time	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Slika 7: Opis tabela *logins*

Field	Type	Null	Key	Default	Extra
id	int(6) unsigned	NO	PRI	NULL	auto_increment
name	varchar(30)	NO		NULL	
type	varchar(30)	NO		NULL	
temperature	varchar(30)	NO		NULL	
working_time_hours	int(11)	NO		NULL	
regDate	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Slika 8: Opis tabele *sensor_values*

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
username	varchar(30)	NO		NULL	
password	varchar(30)	NO		NULL	
last_login_fail	datetime	YES		NULL	
login_fail_attempt	int(11)	YES		NULL	
is_admin	tinyint(1)	YES		NULL	

Slika 9: Opis tabele *users*

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
username	varchar(30)	NO		NULL	
timestamp	timestamp	NO		CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP

Slika 10: Opis tabele *wrong_entry_attempts*

Komande koje su korištene u izradi projektnog zadatka a koje obuhvataju kreiranje baze podataka, kreiranje tabele baze podataka, čitanje podataka iz baze te brisanje unosa u bazi su prikazane u nastavku.

Kreiranje baze podataka izvršeno je komandom: `CREATE DATABASE smartdb;`

Korištenje napravljene baze: `use smartdb;`

Brisanje baze podataka vrši se komandom: `DROP DATABASE smartdb;`

Kreiranje tabela baze podataka izvršeno je sljedećim komandama:

```
CREATE TABLE orders ( id INT(11) AUTO_INCREMENT PRIMARY KEY, typeoforder  
VARCHAR(30) NOT NULL, purchaser VARCHAR(30) NOT NULL, contact VARCHAR(30) NOT  
NULL, note VARCHAR(30) NOT NULL, regDate TIMESTAMP);  
  
CREATE TABLE foodstuffs ( id INT(11) AUTO_INCREMENT PRIMARY KEY, name  
VARCHAR(30) NOT NULL, type VARCHAR(30) NOT NULL, amount VARCHAR(30) NOT NULL,  
regDate TIMESTAMP);  
  
CREATE TABLE wrong_entry_attempts ( id INT(11) AUTO_INCREMENT PRIMARY KEY, username  
VARCHAR(30) NOT NULL, timestamp TIMESTAMP);  
  
CREATE TABLE logins ( id INT(11) AUTO_INCREMENT PRIMARY KEY, username  
VARCHAR(30) NOT NULL, login_time TIMESTAMP);  
  
CREATE TABLE users ( id INT(11) AUTO_INCREMENT PRIMARY KEY, username  
VARCHAR(30) NOT NULL, password VARCHAR(30) NOT NULL, last_login_fail DATETIME,  
login_fail_attempt INT(11), is_admin TINYINT(1));  
  
CREATE TABLE sensor_values( id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY, name  
VARCHAR(30) NOT NULL, type VARCHAR(30) NOT NULL, temperature VARCHAR(30) NOT  
NULL, gas VARCHAR(30) NOT NULL, oxygenlevel VARCHAR(30) NOT NULL, humidity  
VARCHAR(30) NOT NULL, working_time_hours INT NOT NULL, regDate TIMESTAMP);
```

Brisanje tabele: **DROP table ime_tabele;**

Primjer unosa u tabelu *sensor_values*:

```
insert into sensor_values(name,type,temperature, gas, oxygenlevel, humidity,  
working_time_hours) values('Main room','Room',22,10,95,60,0);  
  
insert into sensor_values(name,type,temperature, gas, oxygenlevel, humidity,  
working_time_hours) values('Main refrigerator','Refrigerator',2,0,0,0,100);  
  
insert into sensor_values(name,type,temperature, gas, oxygenlevel, humidity,  
working_time_hours) values('Main fridge','Fridge',1,0,0,0,100);
```

Čitanje pojedinih unosa iz tabele podataka omogućavaju sljedeće komande:

```
SELECT * FROM orders;  
  
SELECT * FROM logins;  
  
SELECT * FROM foodstuffs;  
  
SELECT * FROM sensor_values  
  
SELECT * FROM wrong_entry_attempts;
```

2.1.2 Web servisi

Nakon kreiranja baze podataka, implementirani su web servisi koji omogućavaju upisivanje namirnica, narudžbi i korisnika u bazu, kao i servisi koji omogućavaju čitanje istih iz baze. Svaki servis je implementiran putem odgovarajuće metode. Klase u kojim se nalaze metode koje služe za implementaciju web servisa za dodavanje podataka u bazu su pobrojane i objašnjene u nastavku.

Klase u kojim se nalaze metode koje služe za implementaciju web servisa za čitanje podataka iz baze su pobrojane i objašnjene u nastavku

- ***addFoodStuff*** – klasa sadrži metodu *createFoodStuff(int Id, String Name, String Type, String Amount)*, koja služi za kreiranje web servisa za dodavanje namirnica u bazu podataka
- ***addOrder*** – klasa sadrži metodu *createOrder(int Id, String TypeOfOrder, String Purchaser, String Contact, String Note)*, koja služi za kreiranje web servisa za dodavanje narudžbe u bazu
- ***addLogin*** – klasa sadrži metodu *createLogin(int LoginId, String UserName, String Password)*, koja služi za kreiranje web servisa za dodavanje korisnika u bazu
- ***addFridgeInfo*** – klasa sadrži *createFridgeInfo(int id, String name, String type, int temperature, int gas, int oxygenlevel, int humidity, Integer workingTimeHours, Timestamp timestamp)* koja služi za kreiranje web servisa za dodavanje podataka sa senzora na frižideru u bazu
- ***addRefrigeratorInfo*** – klasa sadrži metodu *createRefrigeratorInfo(int id, String name, String type, int temperature, int gas, int oxygenlevel, int humidity, Integer workingTimeHours, Timestamp timestamp)* koja služi za kreiranje web servisa za dodavanje informacija sa senzora na hladnjači u bazu
- ***addRoomInfo*** - klasa sadrži metodu *createRoomInfo(int id, String name, String type, int temperature, int gas, int oxygenlevel, int humidity, Integer workingTimeHours, Timestamp timestamp)* koja služi za kreiranje web servisa za dodavanje informacija sa senzora na prostoriju u bazu

Klase u kojim se nalaze metode koje služe za implementaciju web servisa za čitanje podataka iz baze su pobrojane i objašnjene u nastavku

- ***getOrder*** – klasa sadrži metodu *getOrderInfo(int Id)* koja na osnovu unesenog ID-a čita podatke o narudžbi iz baze
- ***getLogin*** - klasa sadrži metodu *getLoginInfo(String username)* pomoću koje dobivamo podatke o ulogovanim korisnicima
- ***getFoodStuff*** - klasa sadrži metodu *getFoodStuffInfo(int Id)*, koja na osnovu unesenog ID-a čita podatke o namirnici iz baze
- ***getRefrigeratorInfo*** - klasa sadrži metodu *getRefrigeratorInfoStatus(int refrigeratorinfoid)*, koja na osnovu unesenog ID-a čita podatke sa senzora na hladnjači iz baze
- ***getFridgeInfo*** - klasa sadrži metodu *getFridgeInfoStatus(int fridgeinfoid)*, koja na osnovu unesenog ID-a čita podatke sa senzora na frižideru iz baze
- ***getRoomInfo*** - klasa sadrži metodu *getRoomInfoStatus(int roomid)* koja na osnovu unesenog ID-a čita podatke sa senzora u sobi iz baze

3. Radno okruženje, korišteni alati i tehnologije

Radno okruženje korišteno tokom razvoja web aplikacije detaljno je opisano u tabeli 1. U tabeli 2 prikazani su osnovni alati korišteni tokom pravljenja aplikacije. Detaljnije informacije o navedenim alatima i korištenom okruženju su navedene u nastavku ovog poglavlja.

Radno okruženje	Verzija radnog okruženja
OS Windows	Windows 10
Eclipse Java EE IDE	eclipse-jee-kepler
Java Development Kit	jre1.8.0_191
Aplikacijski server	apache-tomcat-7.0.64

Tabela 1.

Korišteni alat	Verzija korištenog alata
MySQL Database	MySQL 5.5 Command Line Client
SOAUI Client	SoapUI-x64-5.2.1

Tabela 2

3.1 Java

Java je objektno orijentirani programski jezik koji su razvili *James Gosling* i drugi inženjeri u firmi u *Sun Microsystems*. Razvoj je počeo 1991., kao dio projekta *Green*, a objavljen je u decembru 1995. *Sun* posjeduje *trademark* na ime *Java*, ali samo okruženje je moguće bez plaćanja skinuti sa *Sun*-ovih internet poslužitelja. Velika prednost u odnosu na većinu dotadašnjih programskih jezika je to što se programi pisani u *Javi* mogu izvoditi bez promjena na svim operativnim sistemima za koje postoji *JVM* (eng. *Java Virtual Machine*), dok je primjerice klasične programe pisane u C-u potrebno prilagođavati platformi(OS-u) na kojem se izvode. [8]

3.2 Eclipse

Eclipse (od engleske riječi *eclipse* što znači "eklipsa", ili jednako, "pomračenje") je višejezička integrirana razvojna okolina *Eclipse* fondacije napisana u *Javi* i koristi se za razvoj raznih aplikacija. *Eclipse* se sastoji od radnog mjesta (*workspace*) i proširivog sistema za priključke (eng. *plug-in*) za podešavanje okoline. Sa *Eclipse*-om je moguće praviti programe u *Javi* i, uz pomoć raznih priključaka, također u drugim programskim jezicima uključujući jezike Ada, C, C++, COBOL, Fortran, Haskell, Javascript, Lasso, Perl, PHP, Python, R, Ruby, Scala, Clojure, Groovy, Scheme i Erlang. Također se može koristiti za razvoj paketa za program *Mathematica*. [9]

Eclipse sadrži i integrisano razvojno okruženje (engl. *Integrated Development Environment*, skraćeno *IDE*) za *Java* sa potpuno integrisanim funkcijama. Također *Eclipse* omogućava zajednički prostor za organizaciju zadataka razvojnih programera, te rada drugima u zajedničkom direktoriju. Obzirom na prethodno navedeno *Eclipse* predstavlja jedan od najpovoljnijih softvera *Java IDE*-a koji su dostupni na tržištu [5].

3.3 Apache server

Apache Tomcat softver je otvorena implementacija *JAVA Servlet*, *JAVA Servlet Pages*, *JAVA Expression Language* i *JAVA Web Socket* tehnologije. *Apache* se koristi za posluživanje statičkih i dinamičkih stranica na Internetu, čime se programerima znatno olakšava razvoj novih aplikacija. Besplatan je i dostupan na mnogim operativnim sistemima, kao što su *Unix*, *Linux*, *Windows*, *Mac OS X*, itd. [9]

3.4 Senzori

Senzori predstavljaju srce IoT rješenja.

Pet razloga zašto se sve više i više primjenjuju IoT rješenja su:

- smanjenje cijena hardvera
- univerzalna Internet povezanost
- IoT gotovi uređaji i senzori
- jednostavno za upravljanje S tehnologijama
- jeftiniji *cloud*-bazirani servisi

4. Softverski dizajn i implementacija aplikacije (arhitektura, UML dijagrami: dijagram paketa, dijagram klasa, sekvencijalni dijagram)

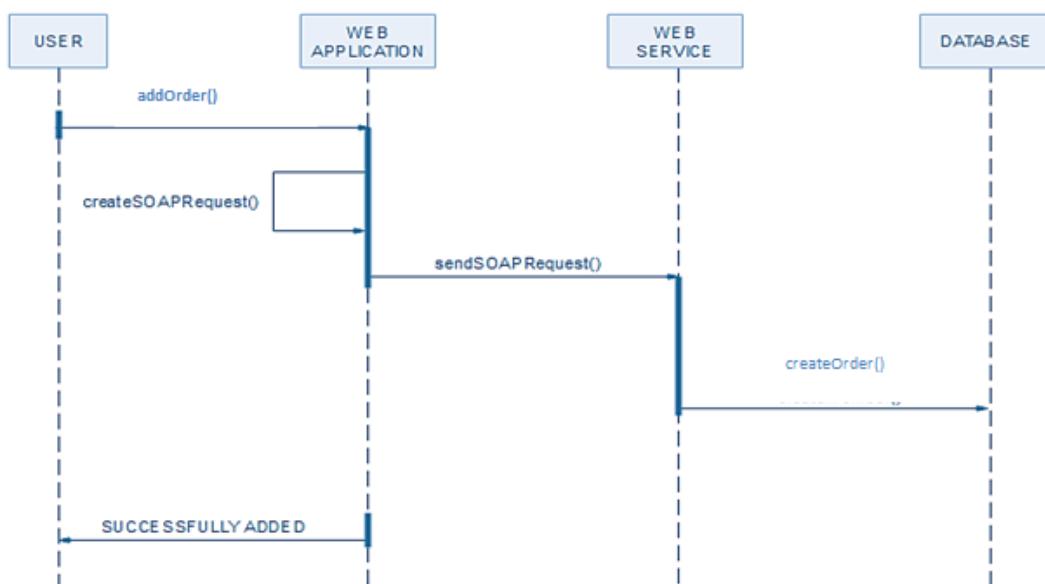
U ovom poglavlju navedeni su osnovni principi vezani za komunikaciju između entiteta sa tehničko aspekta. Navedene su funkcionalnosti koje pruža aplikacija kroz UML dijagrame. Interakcija korisnika i usluga koje su mu omogućene odvija se kroz komunikaciju više entiteta. Entiteti koji učestvuju u komunikaciji su web aplikacija, SOAP web servisi, SOAPUI klijent te odgovarajuće baze podataka. Interakcija između pojedinih entiteta prikazana je na MSC (eng. *Message Sequence Chart*) dijogramima. Takođe, prilikom implementacije web aplikacije kreirane su različite klase i pripadajuće metode koje su prikazane na dijagramu klasa. [11]

4.1 MSC (Message Sequence Chart) dijagrami

Interakcija između pojedinih entiteta za različite slučajeve upotrebe prikazani su na MSC dijagramu. Entiteti koji učestvuju u interakciji su sam korisnik, web aplikacija, web servisi, baze podataka te SOAP UI klijent čiji je zadatak slanja informacija prikupljenih od strane senzora. Za svaki slučaj upotrebe aplikacije prikazan je odgovarajući dijagram.

4.1.1 Dodavanje nove narudžbe

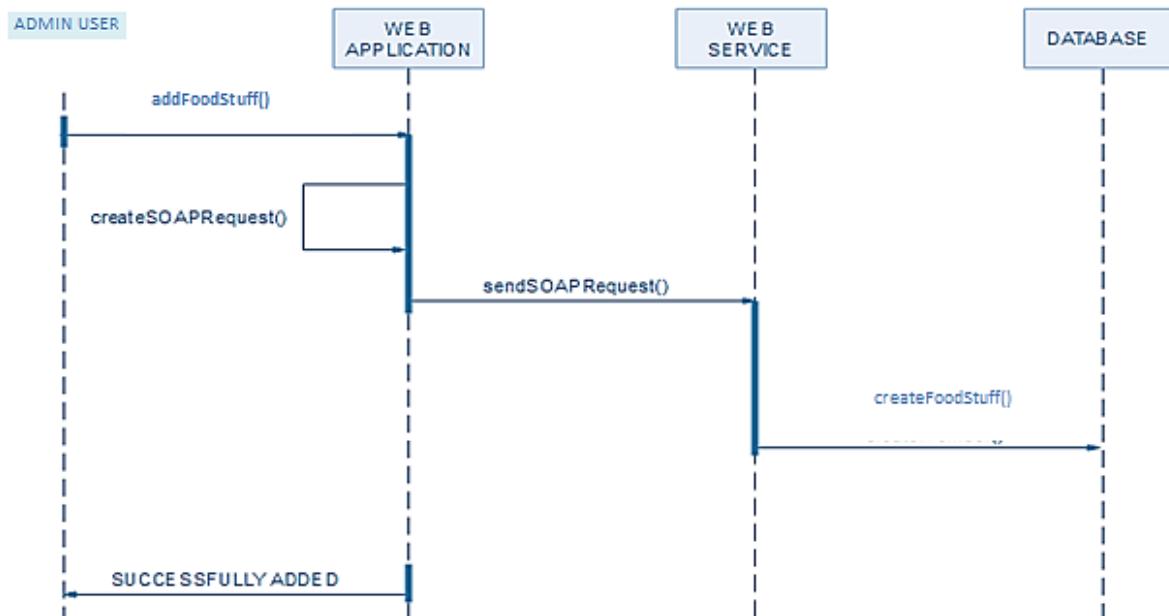
MSC dijagram prikazuje entitete koji učestvuju u interakciji kojom je omogućena usluga dodavanja narudžbi u bazu podataka. Interakcije se odvija na način da korisnik web aplikacije upućuje zahtev web aplikaciji za dodavanje nove narudžbe. Zatim, web aplikacija putem metode *sendSOAPRequest* upućuje zahtev ka web servisu za dodavanje nove narudžbe. Web servis pozivom metode *createOrder()* vrši upis narudžbi u bazu podataka.



Slika 11: Dodavanje nove narudžbe

4.1.2 Dodavanje nove namirnice

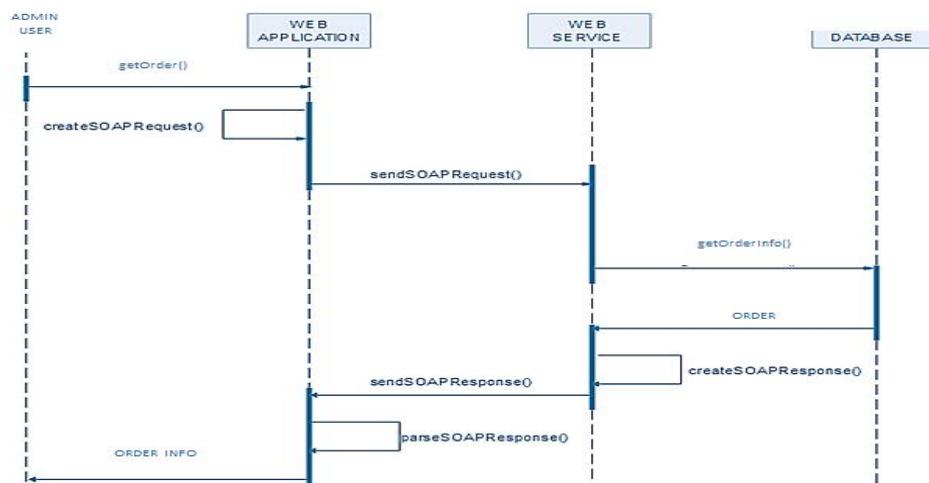
MSC dijagram prikazuje entitete koji učestvuju u interakciji kojom je omogućena usluga dodavanja namirnice u bazu podataka. Interakcije se odvijaju na način da korisnik web aplikacije upućuje zahtjev web aplikaciji za dodavanje nove namirnice. Zatim, web aplikacija putem metode `sendSOAPRequest()` upućuje zahtjev ka web servisu. Web servis pozivom metode `createOrder()` vrši upis namirnice u bazu podataka.



Slika 12: Dodavanje nove namirnice

4.1.3 Dobijanje informacija o narudžbi za uneseni ID parametar

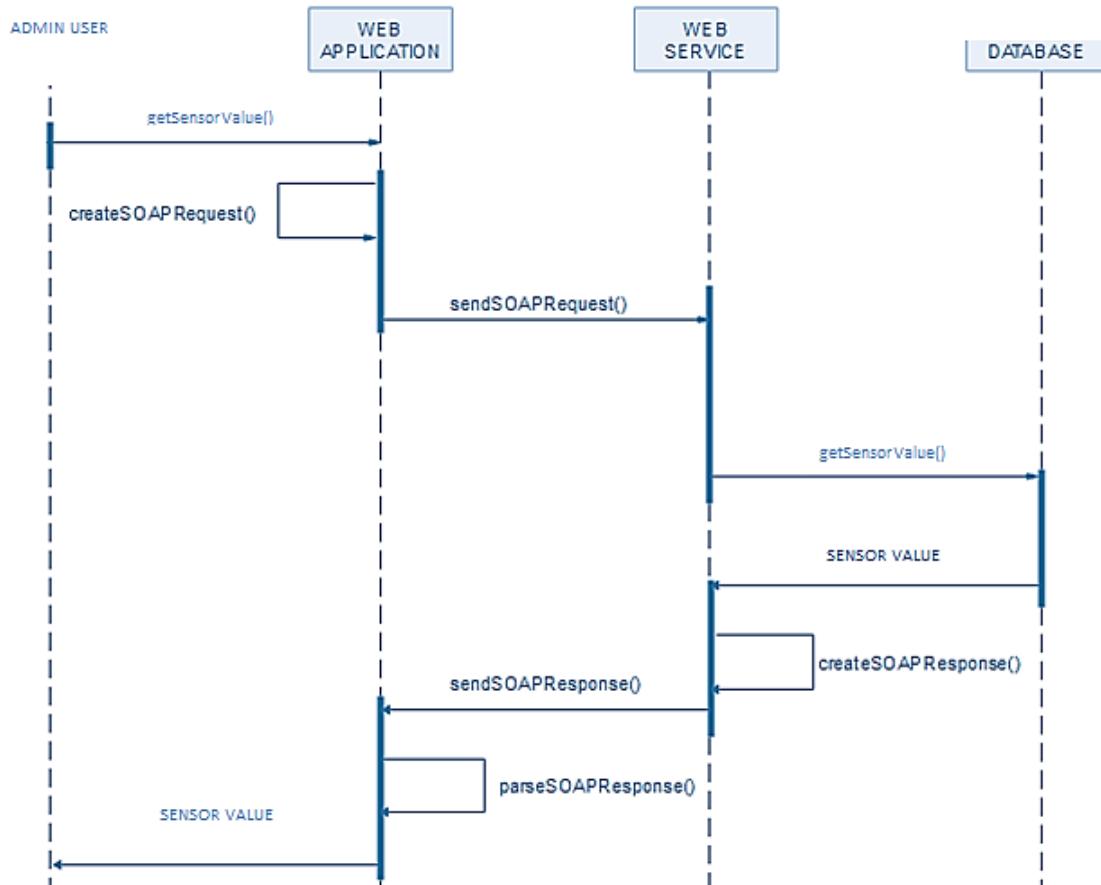
Interakcija se odvija na način da administrator korisnik unosom ID parametra narudžbe, upućuje zahtjev kroz web aplikaciju. Web aplikacija upućuje SOAP zahtjev ka web servisu koji pozivom metode `getFoodStuffInfo()` iz baze podataka preuzima informacije sa navedenim ID parametrom te tu informaciju kroz SOAP odgovor proslijedi ka Web aplikaciji, koja na osnovu dobijenog SOAP odgovora vrši prikaz informacija korisniku web aplikacije. Baza podataka je *orders*.



Slika 13: Dobijanje informacija o narudžbi za uneseni ID parametar

4.1.4 Dobijanje informacija prikupljenih od strane senzora

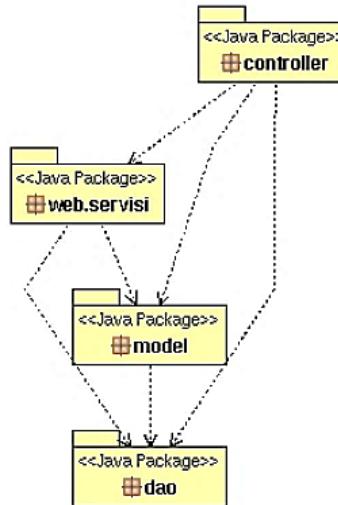
Senzori prikupljaju informacije, *web* servis pomenute informacije upisuje u bazu podataka. Web aplikacija pozivom web servisa pristupa informacijama sa senzora. Zatim, web aplikacija pomenute informacije proslijeđuje korisniku na uvid. Baza podataka je *sensor_values*.



Slika 14: Dobijanje informacija prikupljenih od senzora

4.2 Dijagram paketa

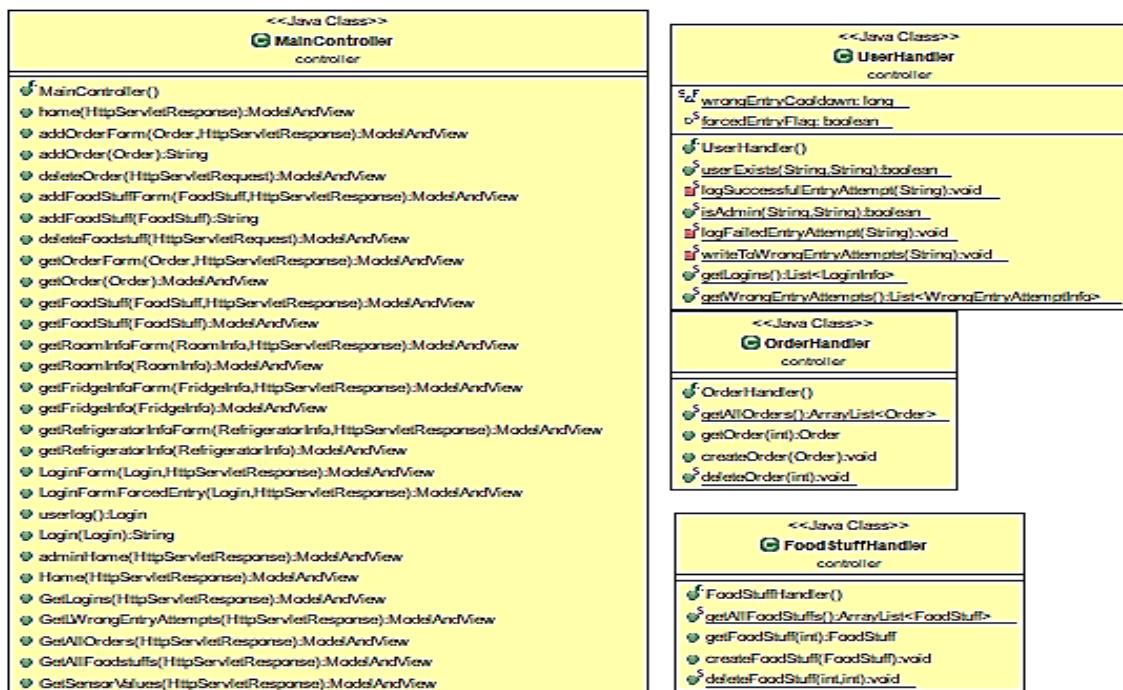
Paketi sadrže klase koje su zadužene za izvršenje pomenutih funkcionalnosti koje nudi web aplikacija. Implementirana su 4 paketa pod nazivom: *dao*, *controller*, *model* i *web.services*. U sklopu *dao* paketa implementirane su klase koje omogućavaju povezivanje sa bazom podataka. Paket *controller* sadrži klase koje omogućavaju različite vizuelne prikaze (forme) koje se nude korisniku web aplikacije te koje definiraju način na koji aplikacija reaguje na različite akcije korisnika. Paket *model* sadrži klase koje simuliraju različite modele koji implementiraju narudžbe, namirnice, podatke prikupljene od strane senzora, korisnike, pogrešno ulogovane korisnike i sl.



Slika 15: Dijagram paketa

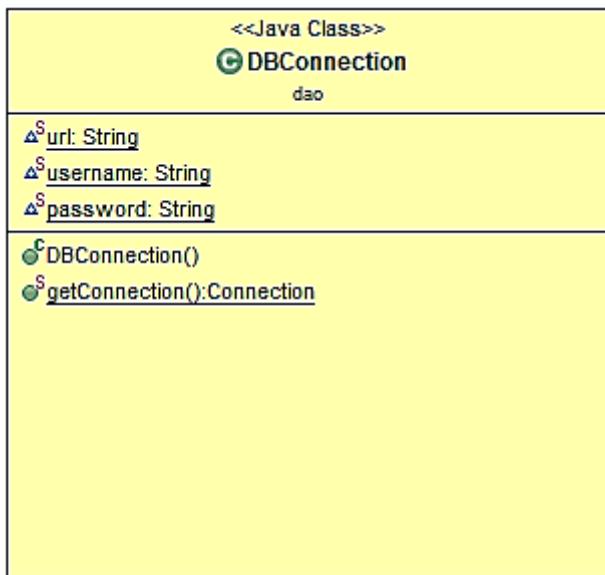
4.3 Dijagram klasa

Naime, zbog složenosti same implementacije te velikog broja atributa i metoda, prikaz klasnih dijagrama izvršen je na osnovu paketa kojem klasa pripada. Naredna slika prikazuje dijagram klasa koje se nalaze u sklopu paketa *controller*. U sklopu paketa *controller* implementirane su 4 klase: *MainController*, *UserHandler*, *OrderHandler* te *FoodStuffHandler*. Zadatak pomenutih klasa jeste upravljanje narudžbom i korisnicima na nivou aplikacije, dok upravljanje narudžbama i korisnicima na nivou *web servisa* implementirano je u sklopu paketa *web.services*.

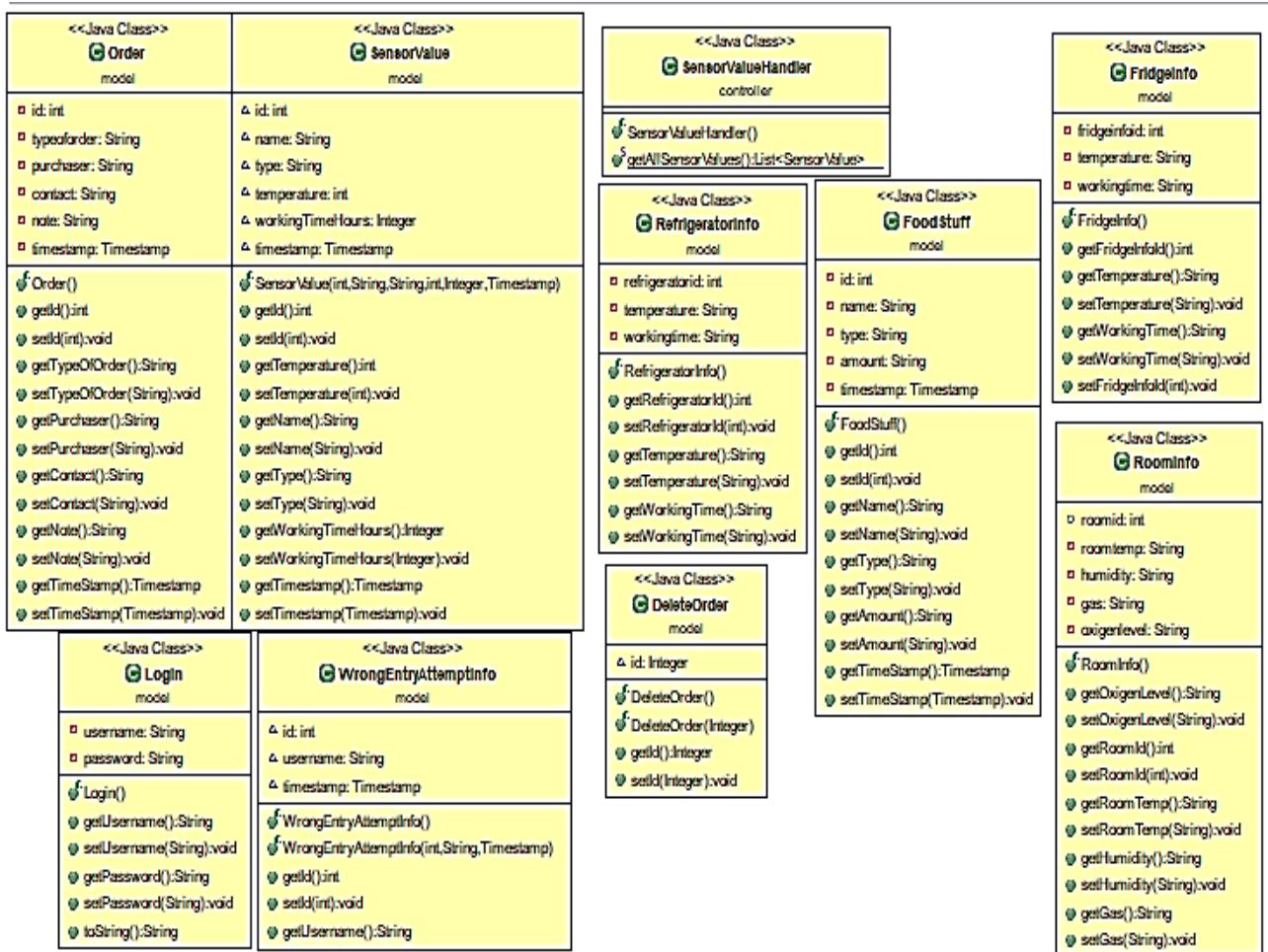


Slika 16: Dijagram klasa za paket controller

U sklopu paketa *dao* nalazi se klasa *DBConnection* koja omogućava povezivanje sa bazom podataka. Paket *web.services* sadrži klase koje omogućavaju unos elemenata u bazu podataka te čitanje podataka iz baze podataka (slika 18).



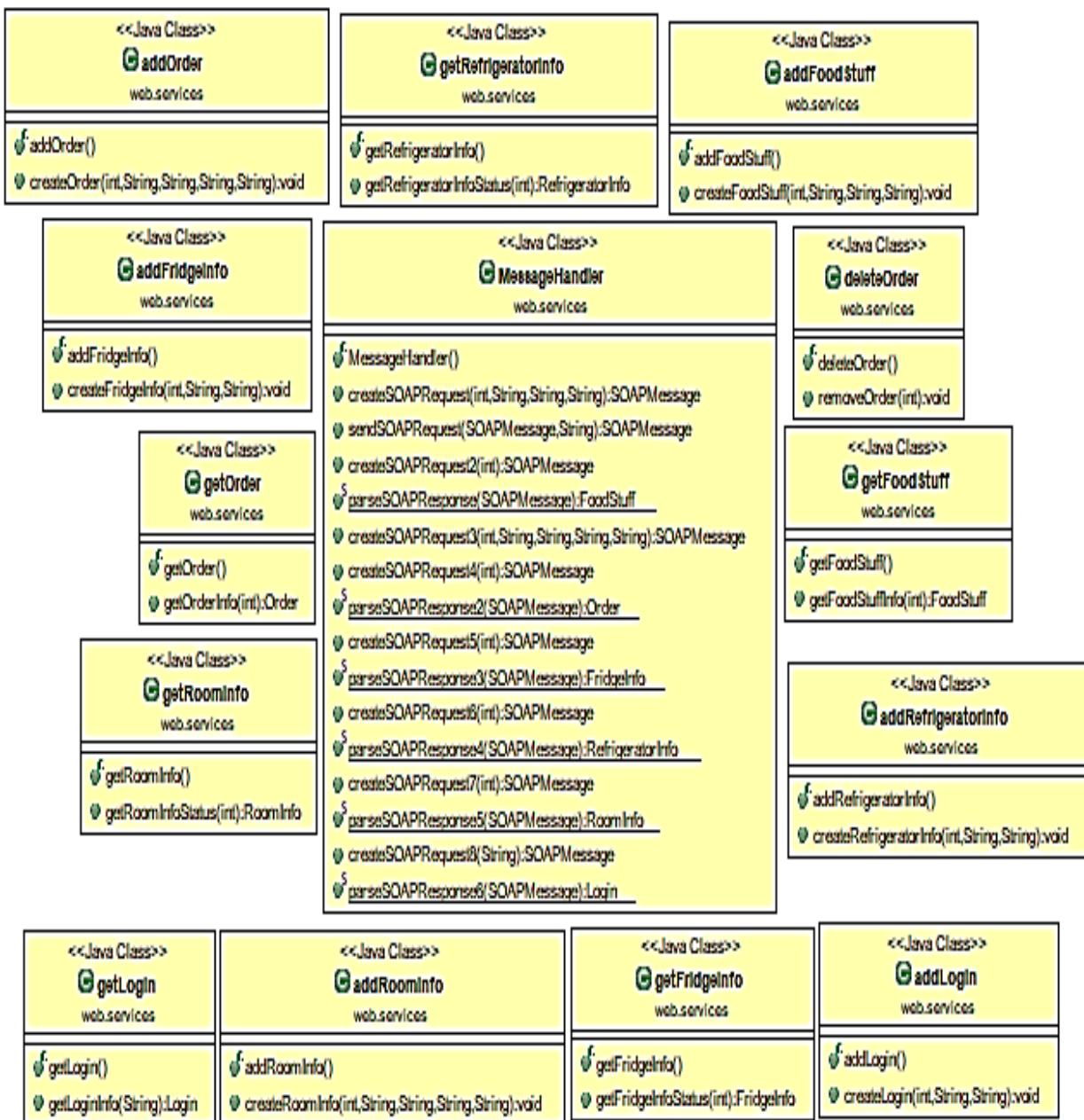
Slika 17: Dijagram klasa za paket dao



Slika 18: Dijagram klasa za paket model

Dijagram klasa koje pripadaju paketu *model*, prikazan je na slici 18. Paket *model* sastoji se iz klasa *DeleteOrder*, *FoodStuff*, *FridgeInfo*, *Login*, *LoginInfo*, *Order*, *RefrigeratorInfo*, *RoomInfo*, *SensorValue*, te *WrongEntryAttemptInfo*, sa odgovarajućim *get* i *set* metodama koje omogućavaju promjenu i očitanje parametara koji su karakteristični za određene klase.

Dijagram klasa za paket `web.services` je prikazan na narednoj slici.



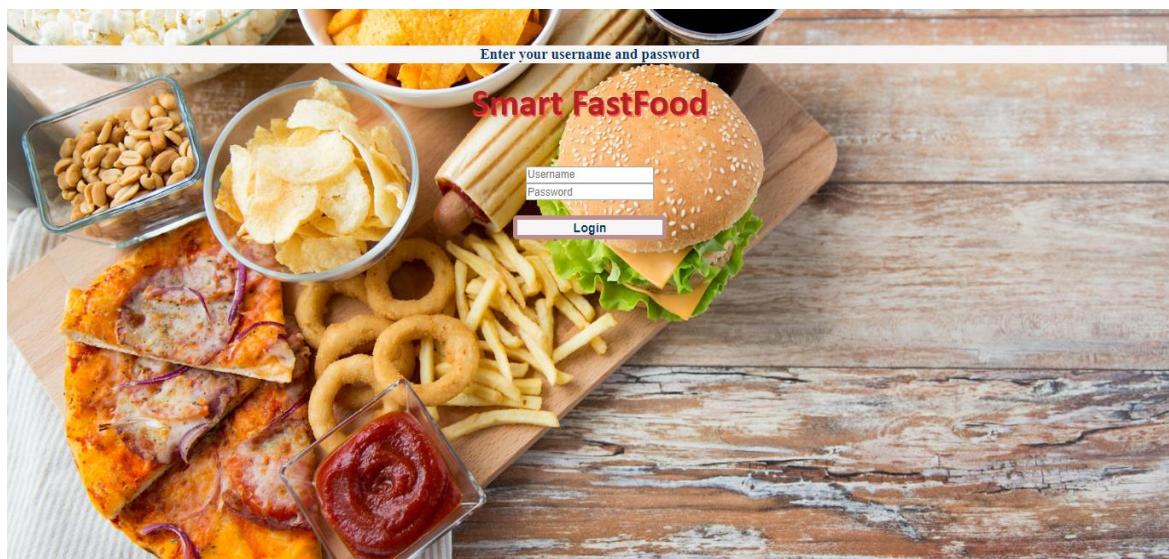
Slika 19: Dijagram klasa za paket model web.services

5. Interaktivni pregled aplikacije

Funkcionalnosti koje ima naša web aplikacija su sljedeće:

- Zahtjev za unosom korisničkog imena i lozinke, nakon čega se, za autorizovanog korisnika, prikazuje glavna naslovna stranica aplikacije. U svrhu demonstracije različitih nivoa hijerarhije korisnika, potrebno je proizvoljno kreirati administrator korisnika, koji će na raspolaganju imati sve funkcionalnosti, te običnog korisnika, koji će imati reducirani set funkcionalnosti na raspolaganju. Vremenski trenutak svakog uspješnog pristupanja aplikaciji od strane običnog korisnika treba biti evidentiran u bazi podataka (u zasebnoj tabeli).

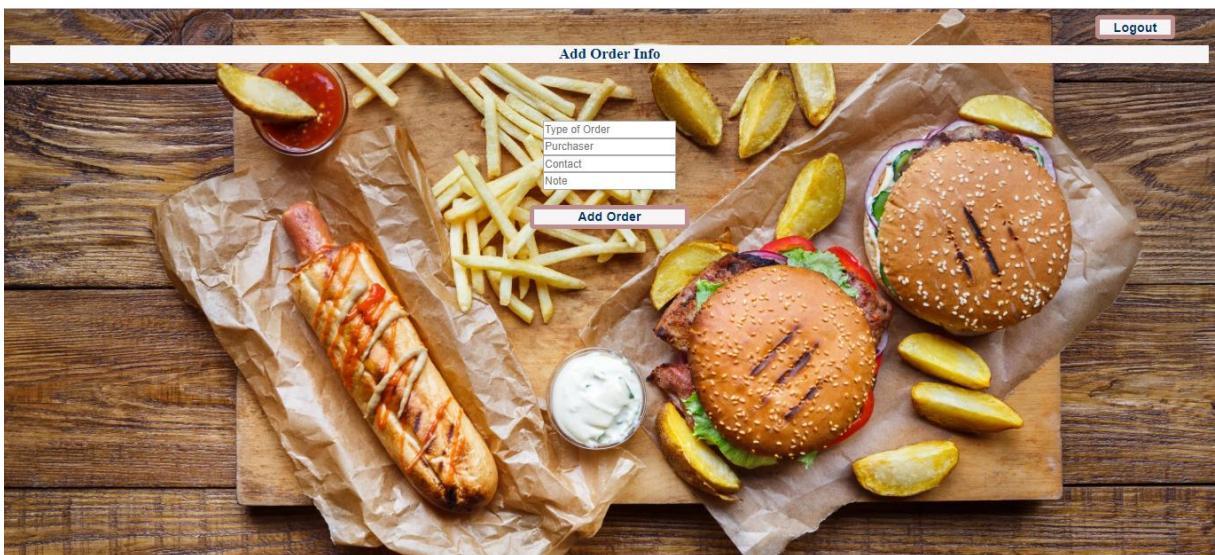
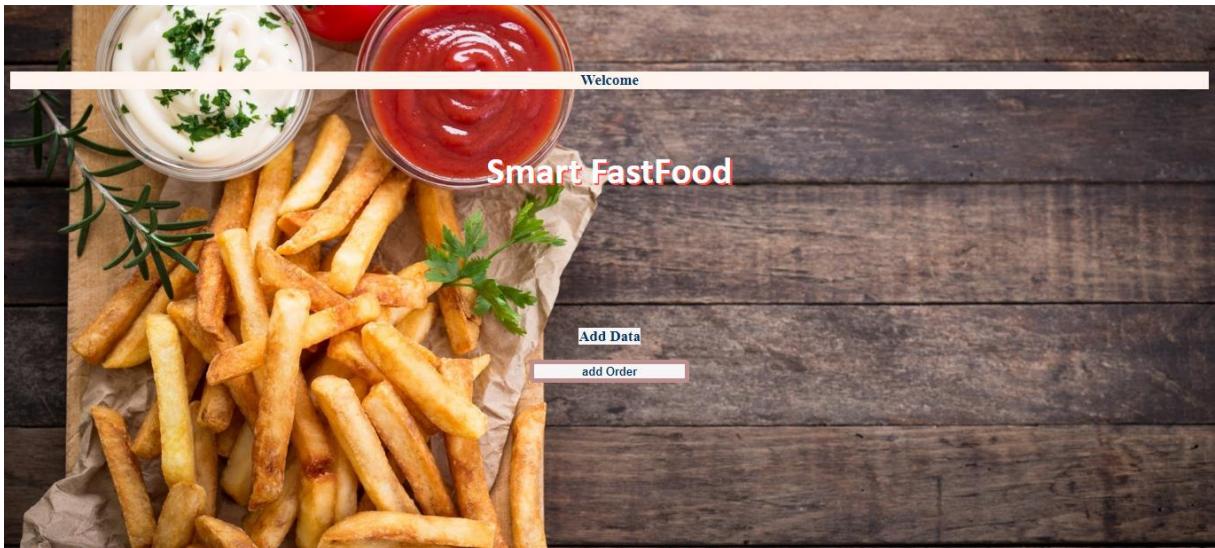
Početna stranica aplikacije je *Login page*, gdje imamo zahtjev za unosom imena i lozinke, a naše ulogovane korisnike možemo vidjeti u bazi podataka u tabeli *logins*.



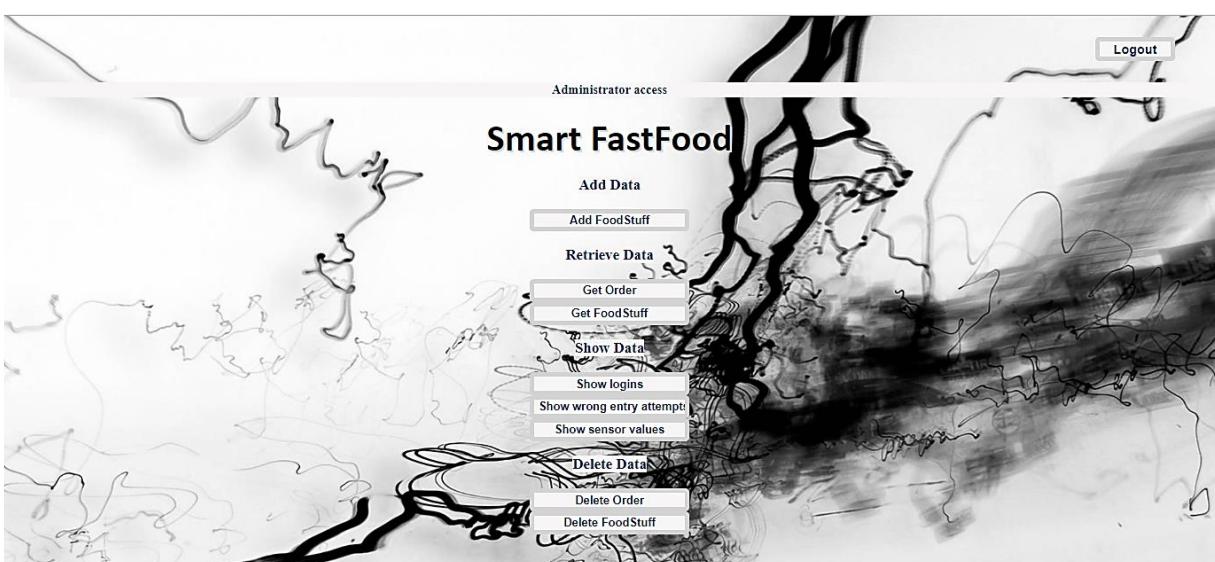
Slika 20: Login stranica

mysql> select * from users;					
id	username	password	last_login_fail	login_fail_attempt	is_admin
1 elma elmapass NULL NULL 0					
2 azra azrapass NULL NULL 0					
3 administrator 12345 NULL NULL 1					
4 tarik tariktarik NULL NULL 0					
5 melisa melisapass NULL NULL 0					

Slika 21: Ulogovani korisnici i administrator korisnik



Slika 22: Dodavanje narudžbi koje su omogućene samo ulogovanom korisniku



Slika 23: Set funkcionalnosti koje su omogućene samo administrator korisniku

Vremenski trenutak svakog uspješnog pristupanja aplikaciji od strane običnog korisnika je prikazan u tabeli *logins* u bazi podataka.

id	username	login_time
36	elma	2019-02-15 23:04:18
37	melisa	2019-02-15 23:13:30
38	melisa	2019-02-15 23:14:27
39	azra	2019-02-15 23:16:32
40	azra	2019-02-16 01:29:07
41	melisa	2019-02-16 01:33:05
42	elma	2019-02-16 13:13:06
43	elma	2019-02-16 13:13:37
44	elma	2019-02-16 16:13:25
45	elma	2019-02-16 16:30:52
46	elma	2019-02-16 16:33:34
47	elma	2019-02-16 16:39:29
48	elma	2019-02-16 17:07:18
49	azra	2019-02-16 17:32:59
50	elma	2019-02-16 19:05:47
51	elma	2019-02-16 20:12:15
52	elma	2019-02-16 20:22:21
53	melisa	2019-02-16 20:35:32
54	elma	2019-02-16 22:13:07
55	melisa	2019-02-16 23:48:13
56	elma	2019-02-17 00:09:11
57	elma	2019-02-17 02:31:17
58	elma	2019-02-17 03:14:49
59	administrator	2019-02-17 20:54:24
60	elma	2019-02-17 21:07:27

Slika 24: Vremenski trenutak svakog uspješnog pristupanja aplikaciji od strane običnog korisnika

- Nakon unosa pogrešnog korisničkog imena ili lozinke tri puta uzastopno (*brute force*), potrebno je u bazi (u zasebnoj tabeli) zapisati vremenski trenutak kada se desio pokušaj „nasilnog“ pristupa aplikaciji, a na ekranu je potrebno da se ispiše odgovarajući tekst, koji govori da je takav pokušaj evidentiran.

Pogrešan unos *password-a* tri puta uzastopno od strane običnog korisnika se zapisuje u tabelu *wrong_entry_attempts* u bazi podataka.



Slika 25: Poruka koja se ispisuje prilikom nasilnog upada

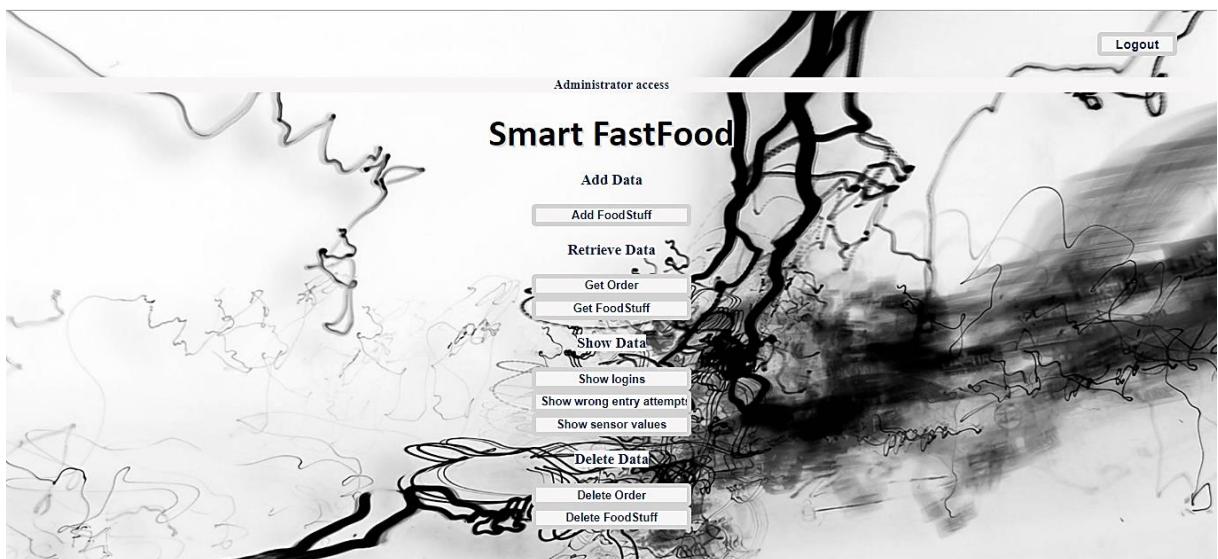
id	username	timestamp
10	elma	2019-02-15 23:40:20
11	azra	2019-02-15 23:42:33
12	elma	2019-02-16 16:23:01
13	elma	2019-02-17 21:06:21
14	melisa	2019-02-17 21:08:42

Slika 26: Vremenski trenutak kada se desio pokušaj „nasilnog“ pristupa aplikaciji

- Svim korisnicima treba biti omogućeno da u svakom trenutku izađu iz aplikacije (da se odjave sa aplikacije) klikom na opciju „Logout“, koja se može nalaziti na proizvoljnem mjestu.



Slika 27: Opcija Logout za običnog korisnika



Slika 28: Opcija Logout za administrator korisnika

- Administrator korisnik, nakon uspješnog pristupa aplikaciji, treba da vidi sve opcije kao i obični korisnik, s tim da dodatno može vidjeti i sljedeće opcije:
 - pregled uspješnih pristupa aplikaciji
 - pregled nasilnih pokušaja pristupa aplikaciji

Napomena: Evidentiranje uspješnih i „sumnjivih“ pristupa trebaju voditi u različitim tabelama unutar baze. Obični korisnik ne može vidjeti ovakve opcije.

Successful Login Info		
Id	Username	SuccessfulLoginTime
62	melisa	2019-02-17 21:53:55.0
61	elma	2019-02-17 21:22:55.0
60	elma	2019-02-17 21:07:27.0
59	administrator	2019-02-17 20:54:24.0
58	elma	2019-02-17 20:31:49.0
57	elma	2019-02-17 20:23:17.0
56	elma	2019-02-17 20:09:11.0
55	melisa	2019-02-16 23:48:13.0
54	elma	2019-02-16 22:13:07.0
53	melisa	2019-02-16 20:35:32.0
52	elma	2019-02-16 20:22:21.0
51	elma	2019-02-16 20:12:15.0
50	elma	2019-02-16 19:05:47.0
49	azra	2019-02-16 17:32:58.0
48	elma	2019-02-16 17:07:18.0

Slika 29: Pregled uspješnih pristupa aplikaciji omogućen samo administrator korisniku

Wrong Entry Attempts Info		
Id	Username	WrongEntryAttemptTime
14	melisa	2019-02-17 21:08:42.0
13	elma	2019-02-17 21:06:21.0
12	elma	2019-02-16 16:23:01.0
11	azra	2019-02-15 23:42:33.0
10	elma	2019-02-15 23:40:20.0

Slika 30: Pregled nasilnih pokušaja pristupa aplikaciji omogućen samo administrator korisniku

- Dodavanje narudžbe u bazu podataka – nakon što korisnik napravi narudžbu, potrebno je informacije o narudžbi unijeti u bazu podataka (šta je naručeno, ko je naručio, kontakt telefon, napomene). **Primjer unosa:** „Narudžba: Cheesburger, Coca-Cola, Naručilac: Mujo Mujić, Kontakt: 062123456, Napomena: Bez sira“ (moguće je i drugačije definisati unos).

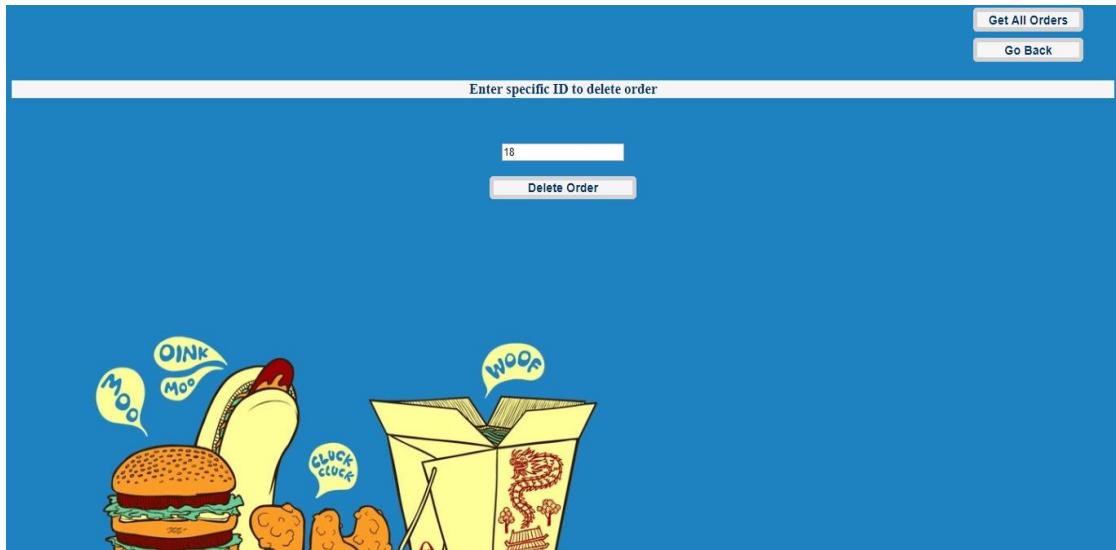


Slika 31: Korisnik pravi narudžbu

```
mysql> select * from orders;
+----+-----+-----+-----+-----+
| id | typeoforder | purchaser | contact | note      | regDate   |
+----+-----+-----+-----+-----+
| 18 | Cheesburger, Coca-Cola | Elma     | 061311244 | bez sira  | 2019-02-17 22:13:15 |
+----+-----+-----+-----+-----+
```

Slika 32: Dodavanje narudžbe u bazu podataka

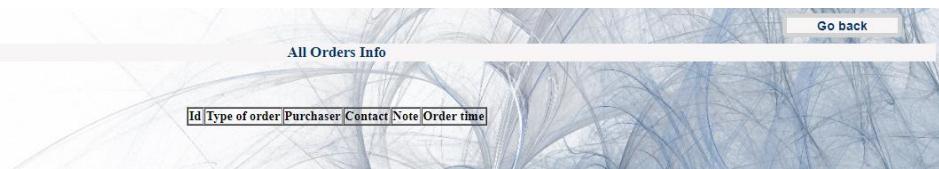
- Brisanje narudžbe iz baze podataka – nakon isporuke narudžbe, potrebno je da se narudžba izbriše iz baze podataka. Brisanje narudžbe treba da se vrši pomoću ID narudžbe. **Primjer unosa:** „ID narudžbe: 1“.



Slika 33: Mogućnost brisanja narudžbe od strane administrator korisnika

```
mysql> select * from orders;
Empty set (0.00 sec)
```

Slika 34: Brisanje narudžbe iz baze podataka



Slika 35: Uvid u sve narudžbe nakon brisanja

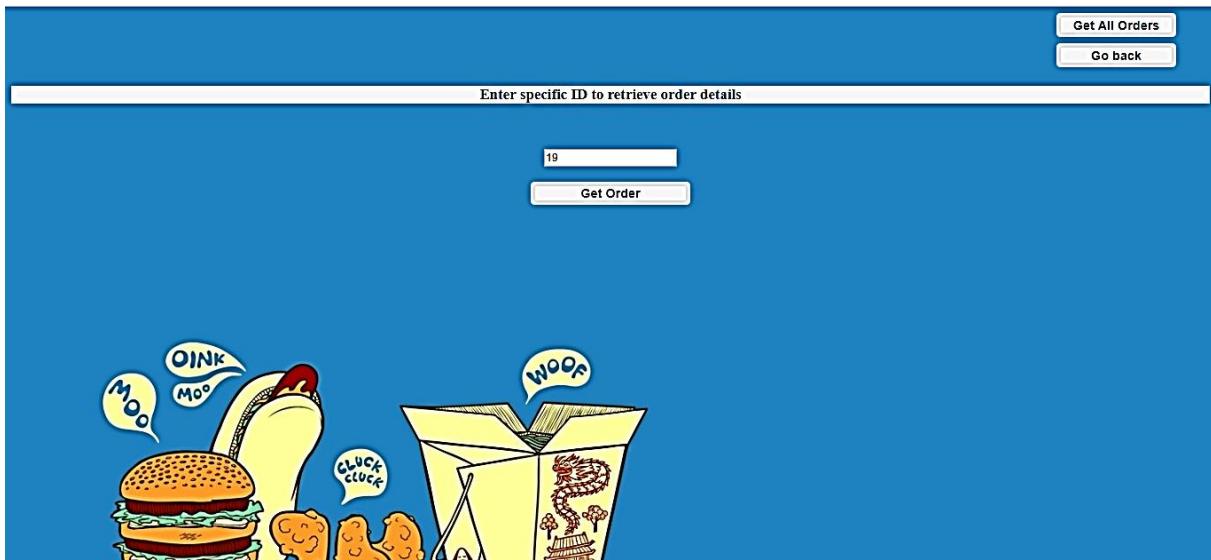
- Pregled svih narudžbi koje se nalaze u bazi podataka, na onakav način kakav su zapisani i u bazi podataka (prepisivanje iz baze).

All Orders Info					
<u>Id</u>	<u>Type of order</u>	<u>Purchaser</u>	<u>Contact</u>	<u>Note</u>	<u>Order time</u>
19	Cheesburger, Coca-Cola	Melisa	061111333	pomfrit	2019-02-17 22:34:08:0
20	Hot dog	Azra	060123456	kecap i majoneza	2019-02-17 22:34:32:0
21	Pomfrit	Tarik	063123456	majoneza	2019-02-17 22:34:57:0
22	Topli sendvic	Melisa	06123456	sir	2019-02-17 22:35:26:0

Slika 36: Pregled svih narudžbi koji se nalaze u bazi podataka

```
mysql> select * from orders;
+-----+-----+-----+-----+-----+
| id | typeoforder | purchaser | contact | note | regDate |
+-----+-----+-----+-----+-----+
| 19 | Cheesburger, Coca-Cola | Melisa | 061111333 | pomfrit | 2019-02-17 22:34:08 |
| 20 | Hot dog | Azra | 060123456 | kecap i majoneza | 2019-02-17 22:34:32 |
| 21 | Pomfrit | Tarik | 063123456 | majoneza | 2019-02-17 22:34:57 |
| 22 | Topli sendvic | Melisa | 06123456 | sir | 2019-02-17 22:35:26 |
+-----+-----+-----+-----+-----+
```

Slika 37: Prikaz svih narudžbi u bazi podataka

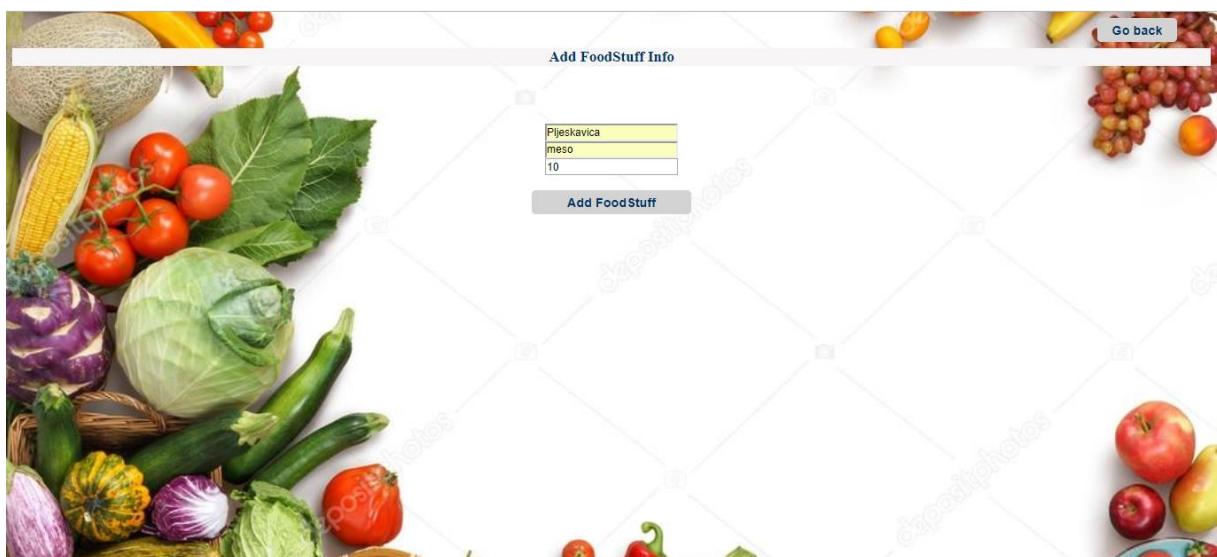


Slika 38: Mogućnost pregleda jedne narudžbe ½



Slika 39: Mogućnost pregleda jedne narudžbe 2/2

- Dodavanje novih namirnica u bazu podataka – nakon nabavke namirnica, potrebno je unijeti informacije o namirnici u bazu podataka (naziv namirnice, tip namirnice, količina). **Primjer unosa:** „Naziv namirnice: Pljeskavica, Tip namirnice: Meso, Količina (kg): 4“.



Slika 40: Dodavanje novih namirnica

mysql> select * from foodstuffs;				
id	name	type	amount	regDate
24	Pljeskavica	meso	10	2019-02-17 22:57:35
25	Luk	Povrce	5	2019-02-17 22:57:59
26	Paradajz	Povrce	5	2019-02-17 22:58:17
27	Krastavac	Povrce	5	2019-02-17 22:58:30
28	Cili	Zacin	0	2019-02-17 22:59:14

Slika 41: Dodavanje novih namirnica u bazu podataka

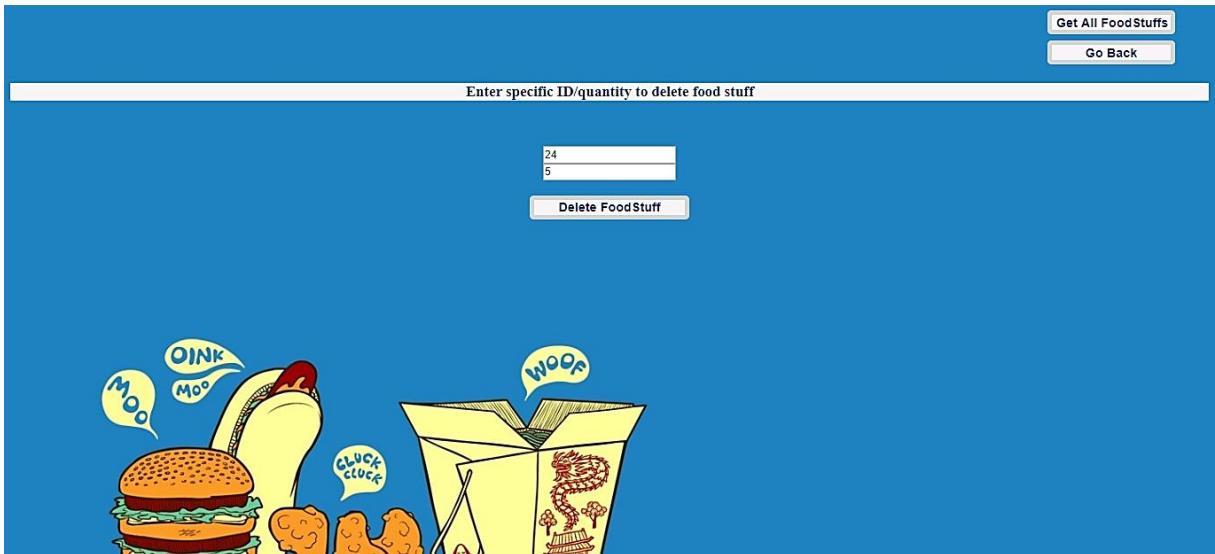
- Brisanje namirnice iz baze podataka – nakon prodaje, ili eliminacije artikla (kompletног ili dijela tog artikla), potrebno je obrisati ili reducirati količinu namirnice u bazi. **Primjer unosa:** „ID namirnice: 1, Količina (kg): 3“. Ukoliko se ne postavi vrijednost količine koja se želi ukloniti, ta namirnica treba biti u potpunosti uklonjena iz baze.



Slika 42: Brisanje namirnice iz baze podataka, ukoliko se ne postavi vrijednost količine koja se želi ukloniti

All FoodStuffs Info				
Id	FoodStuff name	FoodStuff type	Amount	FoodStuff purchase time
24	Pljeskavica	meso	10	2019-02-17 22:57:35.0
25	Luk	Povrce	5	2019-02-17 22:57:59.0
26	Paradajz	Povrce	5	2019-02-17 22:58:17.0
27	Krastavac	Povrce	5	2019-02-17 22:58:30.0

Slika 43: Prikaz namirnica nakon brisanja



Slika 44: Reduciranje količine namirnica

All FoodStuffs Info				
Id	FoodStuff name	FoodStuff type	Amount	FoodStuff purchase time
24	Pljeskavica	meso	5	2019-02-17 23:13:47.0
25	Luk	Povrce	5	2019-02-17 22:57:59.0
26	Paradajz	Povrce	5	2019-02-17 22:58:17.0
27	Krastavac	Povrce	5	2019-02-17 22:58:30.0

Slika 45: Prikaz trenutnog stanja namirnica nakon reduciranja količine

- Pregled svih namirnica koje se nalaze u bazi podataka, na onakav način kakav su zapisani i u bazi podataka (prepisivanje iz baze).

```
mysql> select * from foodstuffs;
+----+-----+-----+-----+-----+
| id | name      | type   | amount | regDate        |
+----+-----+-----+-----+-----+
| 24 | Pljeskavica | meso   |      5 | 2019-02-17 23:13:47 |
| 25 | Luk         | Povrce |      5 | 2019-02-17 22:57:59 |
| 26 | Paradajz    | Povrce |      5 | 2019-02-17 22:58:17 |
| 27 | Krastavac   | Povrce |      5 | 2019-02-17 22:58:30 |
+----+-----+-----+-----+-----+
```

Slika 46: Pregled svih namirnica u bazi podataka

All FoodStuffs Info				
ID	FoodStuff name	FoodStuff type	Amount	FoodStuff purchase time
24	Pjेकavica	meso	5	2019-02-17 23:13:47.0
25	Luk	Povrce	5	2019-02-17 22:57:59.0
26	Paradajz	Povrce	5	2019-02-17 22:58:17.0
27	Krastavac	Povrce	5	2019-02-17 22:58:30.0

Slika 47: Pregled svih namirnica koje se nalaze u bazi podataka

- Prikaz ambijentalnih informacija sa senzora iz hladnjače, frižidera, te prostorija u prodavnici.

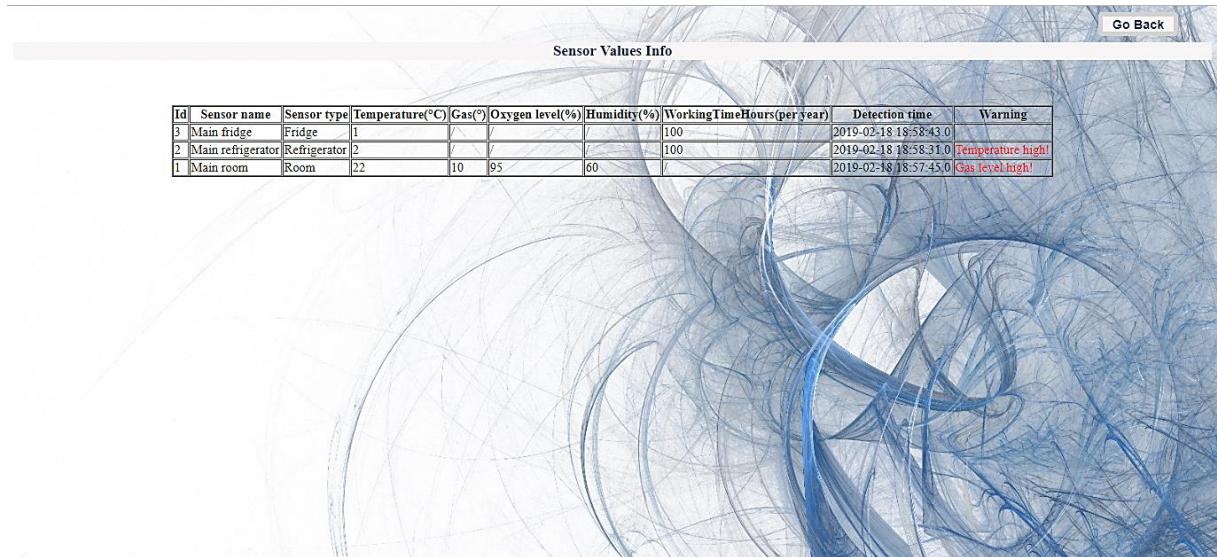
The screenshot shows two SoapUI windows side-by-side. Both windows have 'Raw' selected in the left pane and 'Headers' selected in the right pane. The left window displays an XML request for creating a room sensor info. It includes fields for ser:id (1), ser:room (room1), ser:gas (true), ser:temperature (21), ser:humidity (2), and various warning thresholds for gas, smoke, and humidity. The right window shows the corresponding XML response, which includes the created sensor info with an ID of 1, a name of room1, and a creation date of 2019-02-18 18:57:45.

Izgled SoapUI zahtjeva i odgovora

Sve informacije sa senzora iz hladnjače, frižidera te prostorija u prodavnici su prikazane u jednoj tabeli, koja se naziva *sensor_values*. Takođe, postoji odgovarajuća upozorenja u slučaju prekoračenja vrijednosti parametara, koja se takođe vidi na slici 49.

mysql> select * from sensor_values;								
id	name	type	temperature	gas	oxygenlevel	humidity	working_time_hours	regDate
1	Main room	Room	22	10	95	60	0	2019-02-18 18:57:45
2	Main refrigerator	Refrigerator	2	0	0	0	100	2019-02-18 18:58:31
3	Main fridge	Fridge	1	0	0	0	100	2019-02-18 18:58:43

Slika 48: Prikaz ambijentalnih informacija u bazi podataka



Go Back

Sensor Values Info

Id	Sensor name	Sensor type	Temperature(°C)	Gas(%)	Oxygen level(%)	Humidity(%)	WorkingTimeHours(per year)	Detection time	Warning
3	Main fridge	Fridge	1	/	/	/	100	2019-02-18 18:58:43,0	
2	Main refrigerator	Refrigerator	2	/	/	/	100	2019-02-18 18:58:31,0	Temperature high!
1	Main room	Room	22	10	95	60		2019-02-18 18:57:45,0	Gas level high!

Slika 49: Prikaz ambijentalnih informacija

Zaključak

Prikazani način predstavlja samo jedan od načina realizacije web aplikacija, uz korištenje prethodno navedenog alata, okruženja i tehnologije tj. uz korištenje Java aplikacije uz MySQL bazu podataka, SOAP web servise, *Spring framework* te *Tomcat* aplikacijski server na kojem se aplikacija izvodi. Navedeno predstavlja integrirano rješenje koje omogućava funkcionalnosti koje su neophodne za realizaciju jedne aplikacije. Takođe, korištenje senzora daje uvid u stanje pojedinih prostorija što, u značajnoj mjeri, može poboljšati kvalitet usluge koju nudi jedan *fastfood* restoran.

Na samom kraju opisan je interaktivni pregled aplikacije. Istiće se bitnost grafičkog interfejsa, jer upravo on igra važnu ulogu u ispunjavanju očekivanja projektanta, odnosno *developer-a* aplikacije.

Popis slika

Slika 1: Arhitektura	4
Slika 2: SOAP komunikacija[1].....	6
Slika 3: Web servisi – komunikacijski model [1]	7
Slika 4: Baza podataka smartdb	10
Slika 5: Opis tabele foodstuffs.....	11
Slika 6: Opis tabele orders	11
Slika 7: Opis tabela logins	12
Slika 8: Opis tabele sensor_values	12
Slika 9: Opis tabele users	12
Slika 10: Opis tabele wrong_entry_attempts	12
Slika 11: Dodavanje nove narudžbe	17
Slika 12: Dodavanje nove namirnice	18
Slika 13: Dobijanje informacija o narudžbi za uneseni ID parametar	18
Slika 14: Dobijanje informacija prikupljenih od senzora	19
Slika 15: Dijagram paketa	20
Slika 16: Dijagram klasa za paket controller	20
Slika 17: Dijagram klasa za paket dao.....	21
Slika 18: Dijagram klasa za paket model.....	21
Slika 19: Dijagram klasa za paket model web.services.....	22
Slika 20: Login stranica	23
Slika 21: Ulogovani korisnici i administrator korisnik.....	23
Slika 22: Dodavanje narudžbi koje su omogućene samo ulogovanom korisniku	24
Slika 23: Set funkcionalnosti koje su omogućene samo administrator korisniku	24
Slika 24: Vremenski trenutak svakog uspješnog pristupanja aplikaciji od strane običnog korisnika	25
Slika 25: Poruka koja se ispisuje prilikom nasilnog upada	25
Slika 26: Vremenski trenutak kada se desio pokušaj „nasilnog“ pristupa aplikaciji	25
Slika 27: Opcija Logout za običnog korisnika	26
Slika 28: Opcija Logout za administrator korisnika	26
Slika 29: Pregled uspješnih pristupa aplikaciji omogućen samo administrator korisniku	27
Slika 30: Pregled nasilnih pokušaja pristupa aplikaciji omogućen samo administrator korisniku.....	27
Slika 31: Korisnik pravi narudžbu	27
Slika 32: Dodavanje narudže u bazu podataka	27
Slika 33: Mogućnost brisanja narudže od strane administrator korisnika	28
Slika 34: Brisanje narudže iz baze podataka	28
Slika 35: Uvid u sve narudžbe nakon brisanja.....	28
Slika 36: Pregled svih narudžbi koji se nalaze u bazi podataka.....	28
Slika 37: Prikaz svih narudžbi u bazi podataka	28
Slika 38: Mogućnost pregleda jedne narudžbe %	29

Slika 39: Mogućnost pregleda jedne narudžbe 2/2	29
Slika 40: Dodavanje novih namirinica.....	29
Slika 41: Dodavanje novih namirnica u bazu podataka.....	30
Slika 42: Brisanje namirnice iz baze podataka, ukoliko se ne postavi vrijednost količine koja se želi ukloniti	30
Slika 43: Prikaz namirnica nakon brisanja.....	30
Slika 44: Reduciranje količine namirnica	31
Slika 45: Prikaz trenutnog stanja namirnica nakon reduciranja količine	31
Slika 46: Pregled svih namirnica u bazi podataka	31
Slika 47: Pregled svih namirnica koje se nalaze u bazi podataka.....	32
Slika 48: Prikaz ambijentalnih informacija u bazi podataka	32
Slika 49: Prikaz ambijentalnih informacija	33

Popis skraćenica

SKRAĆENICA	ZNAČENJE
URL	Uniform Resource Locator
HTML	Hypertext Markup Language
XML	Extensible Markup Language
JSON	JavaScript Object Notation
SOAP	Simple Object Access Protocol
HTTP	Hypertext Transfer Protocol
DTD	Document Type Definition
REST	Representational State Transfer
JMS	Java Message Service
AMF	Action Message Format
JDBC	Java Database Connectivity
UDDI	Universal Description, Discovery and Integration
WSDL	Web Service Description Language
JVM	Java Virtual Machine

RFID	Radio-Frequency Identification
JAPI	Java Application Programming Interface
J2EE	Java to Enterprise Edition
IDE	Integrated Development Environment
M2M	Machine to Machine
OMA	Open Mobile Alliance
IoT	Internet of Things
CoAP	Constrained Application Protocol
JSP	Java Server Pages

Literatura

- [1] M. ŠKRBIĆ, *Sistemi i servisi mobilnih telekomunikacija*, Predavanja i dodatni materijali u akademskoj 2017/2018. Sarajevo.
- [2] Margaret Rouse, *SOAP (Simple Object Access Protocol)*. dostupno na: <https://searchmicroservices.techtarget.com/definition/SOAP-Simple-Object-Access-Protocol>
- [3] Margaret Rouse ,M2M, dostupno na: <https://internetofthingsagenda.techtarget.com/definition/machine-to-machine-M2M>
- [4] SOAPUI, dostupno na: <https://www.soapui.org/soapui-projects/soapui-projects.html>
- [5] What areWebServices? . dostupno na : https://www.tutorialspoint.com/webservices/what_are_web_services.htm
- [6] Improve your understanding of web-based application programs. dostupno na: <https://www.lifewire.com/what-is-a-web-application-3486637>
- [7] Why MySQL?. dostupno na: <https://www.mysql.com/why-mysql/>
- [8] Java Powers Our Digital World. dostupno na: <https://go.java/index.html?intcmp=gojava-banner-java-com>
- [9] J. D'ANJOU, *The Java developer's guide to Eclipse*, Addison-Wesley Professional, 2005.
- [10] Apache. dostupno na: <https://httpd.apache.org/>
- [11] Dodatak Eclipse okruženju za crtanje klasnog dijagrama «ObjectAid UML Explorer for Eclipse » dostupno na: <http://www.objectaid.com/>