**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ**



**ФАКУЛТЕТ КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛОГИИ**

Специалност: Информационни технологии в индустрията

**ДИПЛОМНА РАБОТА**

*Тема:* ***Разработване на платформа за търсене и предлагане на стоки и услуги***

Студент: Асен Георгиев Николаев

Фак. No: 501214054

Научен ръководител: гл. ас. д-р Иван Станков

София, 2018

[Въведение 4](#_Toc523690197)

[Какво е интернет магазин? 5](#_Toc523690198)

[История на интернет пазаруването : 5](#_Toc523690199)

[Видове интернет магазини 5](#_Toc523690200)

[Предимства при пазаруване онлайн. 6](#_Toc523690201)

[Платформи за изграждане на интернет магазини 6](#_Toc523690202)

1. [Преглед на вече съществуващи платформи 7](#_Toc523690203)

[Amazon 7](#_Toc523690204)

[eBay 7](#_Toc523690205)

[Etsy 8](#_Toc523690206)

[AliExpress 8](#_Toc523690207)

[Alibaba.com 8](#_Toc523690208)

[OLX group (OnLine Exchange) 8](#_Toc523690209)

2. [Преглед на използваните технологии 9](#_Toc523690210)

[Клиентска част 10](#_Toc523690211)

[SPA (Single page application) архитектура (Lifecycle) 11](#_Toc523690212)

[Сравнение между SPA и традиционната архитектури 11](#_Toc523690213)

[Angular framework 12](#_Toc523690214)

[Модули 13](#_Toc523690215)

[Компоненти 13](#_Toc523690216)

[Шаблони ( Templates ) 14](#_Toc523690217)

[Директиви 14](#_Toc523690218)

[Директиви за компонентите 14](#_Toc523690219)

[*Структурни директиви* 14](#_Toc523690220)

[*Директиви за атрибутите* 14](#_Toc523690221)

[Angular Services 14](#_Toc523690222)

[Инжектиране на зависимости (Dependency Injection) 14](#_Toc523690223)

[Typescript 15](#_Toc523690224)

[Node.js 16](#_Toc523690225)

[Java 17](#_Toc523690226)

[SHA-256 17](#_Toc523690227)

[Spring framework 18](#_Toc523690228)

[Apache Maven 18](#_Toc523690229)

[Жизнен цикъл на Maven 19](#_Toc523690230)

[POM.xml 20](#_Toc523690231)

[Apache Tomcat 20](#_Toc523690232)

[WAR файл 20](#_Toc523690233)

[Предимства на WAR файловете 20](#_Toc523690234)

[Representational State Transfer (REST) архитектура 21](#_Toc523690235)

[База данни 22](#_Toc523690236)

[Oracle database 22](#_Toc523690237)

[MVC (Model-View-Controller) архитектура 23](#_Toc523690238)

[Предимства на MVC 24](#_Toc523690239)

[Недостатък на MVC 24](#_Toc523690240)

3. [Проектиране на базата от данни и програмна реализация 25](#_Toc523690241)

[Описания на таблиците 25](#_Toc523690242)

[таблица „Потребители“ 25](#_Toc523690243)

[Таблица „Обяви“ 26](#_Toc523690244)

[таблица „групи на обявите“ 27](#_Toc523690245)

[таблица „ снимки“ 27](#_Toc523690246)

[таблица „ харесвания„ 28](#_Toc523690247)

[таблица „ харесвания от потребители „ 29](#_Toc523690248)

[ER диаграма на базата данни 30](#_Toc523690249)

[Програмна реализация 30](#_Toc523690250)

[Клиентска част (front-end) – имплеметация 31](#_Toc523690251)

[Компоненти 31](#_Toc523690252)

[RegisterLogin компонент 31](#_Toc523690253)

[Add advertisement компонент 31](#_Toc523690254)

[home компонент 31](#_Toc523690255)

[my-advertisement компонент 32](#_Toc523690256)

[profile компонент 32](#_Toc523690257)

[Admin компонент 32](#_Toc523690258)

[create-new-advertisement-group компонент 32](#_Toc523690259)

[Advertisement-full-information компонент 32](#_Toc523690260)

[Search-by-advertisement-group компонент 32](#_Toc523690261)

[Сървърна част 33](#_Toc523690262)

[Back-end REST контролери 33](#_Toc523690263)

[Login контролер 33](#_Toc523690264)

[Admin контролер 33](#_Toc523690265)

[Edit контролер 33](#_Toc523690266)

[Back-end модели 34](#_Toc523690267)

[Back-end services 34](#_Toc523690268)

[Back-end repositories 34](#_Toc523690269)

[Кохезия 35](#_Toc523690270)

4. [Ръководство на потребителя 36](#_Toc523690271)

[Роли на потребителя 36](#_Toc523690272)

[Екрани и техните функционалности 37](#_Toc523690273)

5. [Приложения 37](#_Toc523690274)

[Регистриране/логин 37](#_Toc523690275)

[Регистриране 37](#_Toc523690276)

[Логин 38](#_Toc523690277)

[Забравена парола 39](#_Toc523690278)

[Създаване на обява 40](#_Toc523690279)

[Административна част 42](#_Toc523690280)

[Търсене по група 44](#_Toc523690281)

[Моите обяви 45](#_Toc523690282)

# Въведение

Online web shop е уеб приложение, което се използва за популяризиране на всякакви обяви и услуги, одобрени от администраторско лице. Регистрираният потребител ,трябва да въведе информация за услугата или продукта който предлага, цена и снимка( по желание). Тази обява става видима в уеб приложението след като администратор я одобри. Администраторът ако я одобрява, трябва да избере група продукти, към която спада съответната услуга/продукт ( коли, имоти, аксесоари и др.)

## Какво е интернет магазин?

**Интернет магазин** е [магазин](https://bg.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD), изграден с набор от компютърни програми ([софтуер](https://bg.wikipedia.org/wiki/%D0%A1%D0%BE%D1%84%D1%82%D1%83%D0%B5%D1%80)), регистриран и достъпен на собствен [уникален адрес в Интернет](https://bg.wikipedia.org/wiki/IP_%D0%B0%D0%B4%D1%80%D0%B5%D1%81), чрез който търговците предлагат на потенциални потребители стоки и услуги от разстояние, извършват възмездно [електронна търговия](https://bg.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D1%82%D1%8A%D1%80%D0%B3%D0%BE%D0%B2%D0%B8%D1%8F) с тях, доставят избраните стоки и изпълняват избраните услуги, като в замяна получават заплащане по законно утвърдени начини.

С все по-мащабното навлизане на [Интернет](https://bg.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) в света на хората неизменно се появяват нови неща, които улесняват живота на обикновените потребители. Сред тях е и пазаруването в Интернет, което спестява много време на обикаляне и търсене по магазините, спестява пари, а закупените стоки са доставяни до вратата на получателя.

## История на интернет пазаруването :

През 1990 година, [Тим Бърнърс-Лий](https://bg.wikipedia.org/wiki/%D0%A2%D0%B8%D0%BC_%D0%91%D1%8A%D1%80%D0%BD%D1%8A%D1%80%D1%81-%D0%9B%D0%B8%D0%B9) създава първият [уеб сървър](https://bg.wikipedia.org/wiki/%D0%A3%D0%B5%D0%B1_%D1%81%D1%8A%D1%80%D0%B2%D1%8A%D1%80) и [браузър](https://bg.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D1%8A%D1%80). През 1991 година, Интернет вече е свободно достъпен и за обикновения потребител. През 1994 година се появяват средства за онлайн разплащане и банкиране, а Netscape представят [SSL](https://bg.wikipedia.org/wiki/SSL) криптиране на данни за осъществяване на сигурна Интернет връзка.

## Видове интернет магазини

Съществуват много различни видове Интернет магазини. Едни от най-разпространите са специализираните. В тях даден търговец предлага определена стока (стоки) от дадена област( автомобили, телефони, компютри, хранителни стоки и др.). Производител на парфюми предлага в своята страница стоката, произвеждана от него или търговец на парфюми предлага своя асортимент.

Интернет [книжарница](https://bg.wikipedia.org/wiki/%D0%9A%D0%BD%D0%B8%D0%B6%D0%B0%D1%80%D0%BD%D0%B8%D1%86%D0%B0) е виртуална книжарница, вид интернет магазин, който предлага [книги](https://bg.wikipedia.org/wiki/%D0%9A%D0%BD%D0%B8%D0%B3%D0%B0) и други книжарски стоки на различни издателства, производители и вносители.

Интернет [хипермаркетите](https://bg.wikipedia.org/wiki/%D0%A5%D0%B8%D0%BF%D0%B5%D1%80%D0%BC%D0%B0%D1%80%D0%BA%D0%B5%D1%82) предлагат десетки хиляди видове стоки, като потребителят може да намери техника, козметика, играчки, приспособления за градината, асортименти за колата и други продукти. [[7]](https://bg.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82_%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD#cite_note-7) [[8]](https://bg.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82_%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD#cite_note-8).

Разновидност на интернет магазините са интернет [моловете](https://bg.wikipedia.org/wiki/%D0%A2%D1%8A%D1%80%D0%B3%D0%BE%D0%B2%D1%81%D0%BA%D0%B8_%D1%86%D0%B5%D0%BD%D1%82%D1%8A%D1%80). В тези Интернет магазини, всеки търговец, който се регистрира в системата може да предложи на пазара своя собствена стока. Така потребителят осъществява връзка директно с търговците или производителите и може да купува от първоизточника, а от своя страна, производителите и търговците спестяват много от изработката и поддръжката на свои собствени Интернет магазини.

Разпространен вид на онлайн магазините са [аукционните](https://bg.wikipedia.org/wiki/%D0%90%D1%83%D0%BA%D1%86%D0%B8%D0%BE%D0%BD) сайтове. В тях всеки може да купува и продава каквото си пожелае, независимо дали е ново или употребявано. Характерна черта при този тип онлайн магазини е, че освен да се закупи дадена стока веднага, може да се наддава за нея, като печели този, дал най-много.

В онлайн магазините за поръчка на храна може да се намери най-близкия ресторант до адреса на посетителя и да се поръчва директно храна за вкъщи, разглеждайки менютата онлайн. Подобен тип магазини навлизат все повече и повече из Интернет пространството. В България вече има и онлайн супермаркети за хранителни стоки и стоки за бита, които осигуряват доставка на продуктите до дома на клиентите за изключително кратко време. Това е голямо удобство в двадесет и първи век, когато хората са толкова заети и стресирани.

Друга разновидност на онлайн магазините са тези, предлагащи пазаруване с отстъпки. Всичко се състои в това, даден търговец предлага своята стока или услуга с 50 или повече процента отстъпка, като потребителят може да закупи това, което се предлага на изключително евтина и изгодна цена.

Груповото пазаруване известно още като колективно пазаруване е такъв вид пазаруване, при което един търговец предлага за групово пазаруване продукт или услуга, която предлага, но най-често става дума за услуга, с намаление. Клиентите виждат офертата му, харесват я и я купуват веднага, като заплащането е предимно чрез банков превод, кредитна или дебитна карта. По този начин купувачите образуват група, като предварително е зададен минимален брой клиенти, за да може търговецът да направи отстъпката. След достигането на минималният брой клиенти, всички които са платили предварително за услугата или стоката получават в електронните си пощенски кутии ваучер, който е необходимо да се принтира и с него да се явят на определено място, където да получат услугата или стоката, която вече са заплатили.

## Предимства при пазаруване онлайн.

Пазаруването от Интернет улеснява намирането на стоки и услуги и сравняването им с предлаганите в другите магазини без да се налага физическо присъствие в магазина. Закупената стока се доставя до указано от клиента място, което спестява транспортни разходи. Обикновено в онлайн магазините стоките са по-евтини от тези, които се предлагат в обичайните магазините, защото в Интернет не се правят голяма част от обичайните разходи за наем, персонал и други съпътстващи обичайната дейност.

## Платформи за изграждане на интернет магазини

Платформа за изграждане на [онлайн магазини](https://bg.wikipedia.org/wiki/%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD_%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD) е автоматизирана система, която изгражда пълноценен независим [онлайн магазин](https://bg.wikipedia.org/wiki/%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD_%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD).

По отношение на [хостинга](https://bg.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82%D0%B8%D0%BD%D0%B3) съществуват два вида платформи:

- SAAS (Software as a service) – софтуер като услуга. В този случай платформата предоставя свой собствен хостинг.

- Self Hosted – В този случай, за да използва софтуера трябва да закупи хостинг и да се инсталира платформата.

Качествените платформи дават възможност потребителя да избере и закупи собствен [домейн](https://bg.wikipedia.org/wiki/%D0%94%D0%BE%D0%BC%D0%B5%D0%B9%D0%BD) който да бъде насочен към магазина, а така също и използване на [електронна поща](https://bg.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BF%D0%BE%D1%89%D0%B0) върху домейна на потребителя.

Съществуват различни форми на електронна търговия според това какви са участниците в нея, широко използвани са видовете:

* [**B2C**](https://bg.wikipedia.org/w/index.php?title=B2C&action=edit&redlink=1) – *business to customer*, или бизнес към клиент, представител на който са класическите онлайн магазини, базирани на каталожна търговия.
* [**B2B**](https://bg.wikipedia.org/wiki/B2B) – *business to business*, или бизнес към бизнес, например когато фирми интегрират информационните си системи, за да автоматизират обмена на стоки и информация помежду им.
* [**C2C**](https://bg.wikipedia.org/w/index.php?title=C2C&action=edit&redlink=1) – *customer to customer*, или клиент към клиент, схеми, където търговията се осъществява между потребители на платформа, а доставчикът на платформата само осигурява инфраструктурата, често срещу такса.
* [**B2G**](https://bg.wikipedia.org/w/index.php?title=B2G&action=edit&redlink=1) – *business to government*, или бизнес към правителство, където компании предлагат услуги на правителството с необходимите за целта специфики, напр. прозрачност.

# Преглед на вече съществуващи платформи

## **Amazon**

Гигант в интернет мрежата. Сайтът предлага милиони предложения за всеки един тип потребители. [Amazon](http://www.amazon.com/) започва като малка книжарница, достигайки в момента най-мащабното присъствие в интернет.

Именно Amazon е най-популярният сайт за пазаруване в Америка. В него могат да се намерят богато разнообразие от артикули, свързани с музика, филми, софтуер, книги, игри, електроника, козметика, мода, мебели и др. Може да се пазарува напълно безопасно от сайт като Amazon, тъй като той е един от сигурните онлайн магазини.

Amazon определено си е заслужил мястото на гиганта в онлайн пазаруването.

## eBay

Това е най-големият онлайн аукцион в света. Сайтът предлага страхотни възможности за пазаруване на абсолютно всякакви продукти, при това, по възможно най-лесен и изгоден за клиента начин. С [eBay](http://www.ebay.com/) може и потребителят да бъде самият търговец, продавайки стоки, които не са му нужни, но може би представляват интерес за някой друг, някъде по света. Това е един чудесен начин и да си осигурява допълнителни приходи. eBay е сайт, в който освен че можеш да намериш всякакви интересни продукти, но и е много лесно и бързо да селектираш, точно това, което търсиш. Онлайн шопинг платформата е изградена, така че да улеснява клиентите, предлагайки специални филтъри на търсачката, сред които бранд, цена, цвят, размер, рейтинг и други.

## **Etsy**

Този интересен шопинг сайт дава уникалната възможност да се пазарува директно от талантливи занаятчии, от различни точки на света. В [Etsy](https://www.etsy.com/) можеш да намериш всякакви оригинални и креативни стоки, включително ръчно изработени бижута, дърворезба и още изключително много забавни артикули, които могат да послужат за подарък на любим човек. Разбира се, ако потребителят имаш хоби да изработва арт неща, то задължително трябва да се възползва от възможностите, които сайтът предлага, а именно – да продава и популяризира изкуството си по света.

## **AliExpress**

[AliExpress](https://www.aliexpress.com/) е сайт за всичко – от дрехи до козметика, аксесоари и електроника. Освен това – сайтът предлага едни наистина конкурентни и ниски цени. Според сайта, AliExpress разполага с над 100 млн. продукта, чиито доставчици се намират главно в Китай.

Разбира се, сайтът има система за защита и е един от сигурните начини за пазаруване онлайн.

## Alibaba.com

Това е най-мащабният сайт за онлайн пазаруване в Китай и един от най-популярните в световен мащаб. В момента [Alibaba](https://www.alibaba.com/) разполага с 42 продуктови категории, включително мода, козметика, дом и градина, авточасти, електроника и грижа за здравето.

## OLX group (OnLine Exchange)

OLX Group е глобален онлайн пазар (със седалище в Амстердам и притежаван от Южна Африка медийна и технологична група Naspers), работещ в 45 страни и е най-голямата онлайн рекламна компания в Пакистан, Бразилия и Индия. България, Полша, Португалия и Украйна. Основана е през 2006 г.

Фабрис Гринда и Алек Оксенфорд стартираха компанията като алтернатива на Craigslist за света извън Съединените щати.

OLX е платформа за покупка и продажба на услуги и стоки, както и електроника, мебели, домакински стоки, автомобили и велосипеди. През 2014 г. платформата има 11 милиарда показвания на страници, 200 милиона месечни активни потребители, 25 милиона обяви и 8,5 милиона транзакции на месец.

Компанията инвестира в "Web 2.0" функции през 2008 г.:. Като джаджи на социални мрежи, подобрено търсене, Ajax-базирани редактори, интерактивни карти и мобилни версии.

През 2009 г. компанията си партнира с Hi5 - социална мрежа, която има 60 милиона потребители. Hi5 имплементира OLX функциите: Като показване на реклами и споделянето им с приятели, а OLX активиран видео, изображения и функции за мобилни устройства, в 39 езика и 90 страни.През 2010 г. мнозинството от компанията е закупено от южноафриканската медийна група Naspers, която изкупи съществуващите инвеститори.

Главен изпълнителен директор на Alec Oxenford казва в интервю 2014 направих OLX прие "Martian подход" към международна експанзия, стартира в Индия, най-големият големия наличен на пазара, отколкото в САЩ. [1] Naspers затвърди онлайн класифициран операции във Филипините, Тайланд, Полша, Унгария, България, Румъния, Украйна, Беларус, Казахстан и Индонезия и да ги ре-брандиран като OLX.

Компанията инвестира сериозно в телевизионната реклама. Oxenford заяви, че приемането на интернет от повече от три трилиона хора е направил телевизия по-ефективен като двигател на трафик към сайтове или приложения, отколкото това, което през ранния период дот-ком, когато телевизионната реклама не работи добре за уеб сайтове. Той каза, че OLX е действал като "създател на богатство" на нововъзникващите пазари, което позволява на хората лесно да си осигурят приходи от услугите и имуществото си

През 2014 г. глобалният трафик на OLX е 240 милиона уникални месечни посещения, от които 54% ​​са от мобилни устройства.

В края на 2016 г. тя стартира Tradus като сайт за тежки обяви за машини. Той изброява реклами за употребявани тежки машини и превозни средства в областта на строителството, транспорта и земеделието, а така също и резервни части за такива машини.

# 2. Преглед на използваните технологии

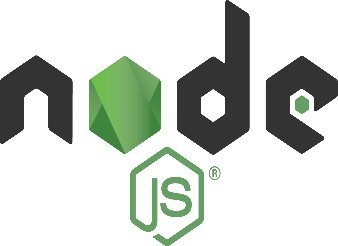
Приложението следва популярната MVC (Model View Controller) архитектура.

Приложението е разделено на 2 части : front end ( клиентска част ) и back end ( сървърна част ).

Клиентската част е написана на популярният open source framework Angular А сървърът на който работи приложението – Nodejs.

Сървърната част е написана на популярният програмен език Java. А сървърът на който работи приложението – Tomcat.

За база данни се използва Oracle.



Клиентската част прави request-и към backend-a, който от своя страна прави request-и към базата данни, базата данни връща съответния резултат в зависимост от request-a , след което backend-a връща получения резултат към клиентската част.

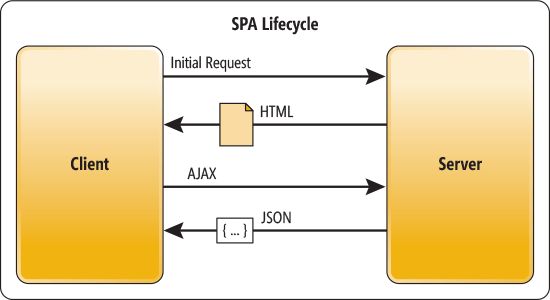
Клиентската част получава данните и ги репрезентира за user-а.

## Клиентска част

Клиентската част е написана на популярният framework Angular.

Използва се SPA (Signle page application) архитектура.

## SPA (Single page application) архитектура (Lifecycle)



Характерно за тази архитектура, е че браузърът само веднъж тегли всички необходими файлове - HTML, JavaScript и CSS , или съответните ресурси се зареждат динамично (асинхронно) използвайки AJAX, и се добавят към страницата, ако е необходимо, обикновено в отговор на действията на потребителя.

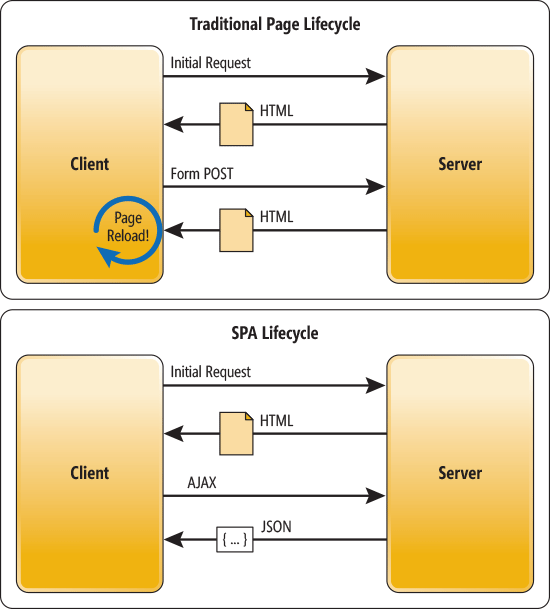
Страницата не се презарежда в нито един момент от сърфирането. Взаимодействието с приложението с една страница често включва динамична комуникация с уеб сървъра зад кулисите.

В SPA архитектурата, всяка страничка представлява отделен изглед, който най често е регистриран като компонент на приложението. Когато потребителят поиска дадена страничка, приложението скрива текущата отворена страница, и зарежда съответният компонент който отговаря за тази страничка.

Request-ите към сървъра обикновено водят получаването на данни в различен формат. Най-често използваните формати за получаване на данни са XML или JSON. Те се използват, защото са лесни за разбиране и четими от обикновеният човек.

Понякога от сървъра може да се получи и нов HTML код.

## Сравнение между SPA и традиционната архитектури



При традиционният подход, при всяко отваряне на нова страничка, клиентското приложение прави request към сървърът, който от своя страна връща HTML,CSS & JS кодовете на съответната страничка.

При странички с голямо съдържание, това може да направи приложението ни бавно и също така да натовари сървърите.

При SPA подходът, клиентското приложение прави request към сървърът, който от своя страна връща HTML,CSS & JS кодът на всички странички от приложението. От тук нататък браузърът си зарежда съответната страничка без да прави излишни извиквания към сървъра.

## Angular framework

Angular е TypeScript базирана платформа за уеб приложения от типа "open source", ръководена от Angular екипът в Google .

Сървърът на който работи приложението – Nodejs.

Работещ на повече от една платформа ( браузър ).

#### Архитектура на Angular приложение



Основните елементи са модули, компоненти, шаблони, метаданни, обвързващи данни, директиви, услуги и инжектиране на зависимости (Dependency Injection) .

Модули

Модулите в Angular са мястото, където могат да се групират компонентите, директивите, pipe’s и services , които са свързани с приложението.

Едно приложение може също да бъде изградено от множество отделни модули, като ще има един модул който обхваща всички тези отделни модули в едно.

## Компоненти

Голяма част от разработката с Angular се извършва в компонентите. Компонентите основно взаимодействат с .html файла на компонента, който се показва в браузъра.

В компонените се пише логиката, която трябва да се изпълнява при комуникация между потребителят и страницата.

Компонентите се използват като контролери, които обновяват модела(данните) и комуникират със сървърната част.

## Шаблони ( Templates )

Идеята зад шаблоните в Angular , е че всеки компонент може да дефинира шаблон ( таг име ) зад което да се зарежда друго HTML съдържание.

## Директиви

Директивите в Angular са javascript класове. Имаме 3 вида директиви в Angular.

## Директиви за компонентите

Те представят подробностите за това как компонентът трябва да бъде обработен, инстанцииран и използван по време на изпълнение.

## Структурни директиви

Проблем с манипулирането на елементите на домейна. Структурните директиви имат знак \* преди директивата. Например, \* ngIf и \* ngFor.

## Директиви за атрибутите

Директивите за атрибутите се използват за промяна на до поведението и изгледа на документа.

## Angular Services

Може да срещнем ситуация, в която има нужда от някакъв код навсякъде на страницата. Той може да се използва за връзка за данни, която трябва да бъде споделена между компонентите и т.н. Услугите ни помагат да постигнем това. С услугите можем да получим достъп до методи и свойства в други компоненти в целия проект.

## Инжектиране на зависимости (Dependency Injection)

В софтуерното инженерство, инжектирането на зависимост (Dependency Injection) е техника в един обект (или статичен метод) доставя зависимостите на друг обект. Зависимостта е обект, който може да бъде използван (услуга). Инжектирането е преминаването на зависимост към зависим обект (клиент). Услугата е част от състоянието на клиента. Предаването на услугата на клиента, вместо да позволи на клиента да изгради услугата, е основното изискване на модела.

Това основно изискване означава, че използването на стойности (услуги), произведени в този клас от нови или статични методи, е забранено. Клиентът трябва да приеме стойности, предавани отвън.

Инжектирането на зависимостта е една от формите на по-широката техника на инверсия на контрола. Както при другите форми на инверсия на контрола, инжектирането на зависимост поддържа принципа на инверсията на зависимостта. Клиентът делегира отговорността да предоставя зависимостите си от външен код (инжектора). Клиентът няма право да се обажда на кода на инжектора, това е инжекционният код, който изгражда услугите и се обажда на клиента, за да ги инжектира. Което означава, че клиентският код не трябва да знае за инжектиращия код; клиентът трябва да знае само за вътрешните интерфейси на услугите. Това разделя отговорностите за използване и конструиране.

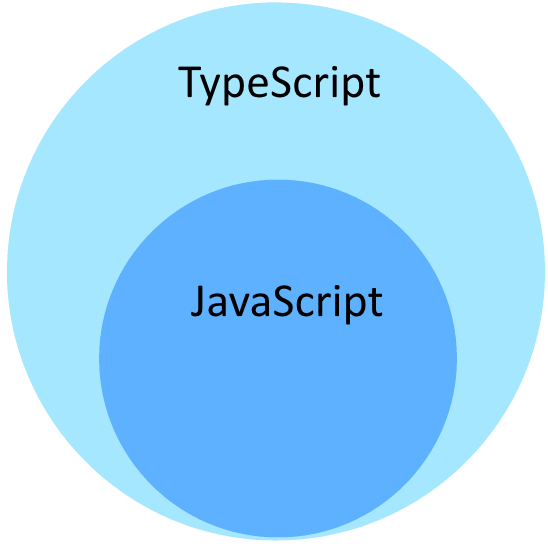
Има три общи начина, по които даден клиент може да приеме инжектиране на зависимост: настройка, интерфейс и инжектиране на конструктор. Инжектирането на сетер и конструктор се използва главно от тях. Инжектирането на интерфейса се различава, тъй като на зависимостта се дава възможност да контролира собствената си инжекция. Всеки от тях изисква отделен строителен код (инжекторът) да поеме отговорност за въвеждане на клиент и неговите зависимости един от друг.

## Typescript

TypeScript е език за програмиране с отворен код, разработен и поддържан от Microsoft.

TypeScript е предназначен за разработване на големи приложения тъй като той се преобразува в чист JavaScript .

TypeScript е надмножество на JavaScript, т.е Javascript + екстра фийчъри.



TypeScript може да се използва за разработване на JavaScript приложения както за изпълнение от страна на клиента, така и от страна на сървъра (Node.js).

TS е предназначен за по обектно ориентирани програмисти, тъй като използва стандартните принципи на ООП ( Наследяване, капсулация, абстракция и полиморфизъм ).

Typescript е лицензиран под лиценза Apache 2.

TypeScript произлиза от недостатъците на JavaScript за разработването на мащабни приложения както в Microsoft, така и сред външните им клиенти.

Тип разработчиците търсят решение, което не би попречило на съвместимостта със стандарта и неговата поддръжка на различни платформи. Знаейки, че настоящото предложение за стандарт ECMAScript обеща бъдеща подкрепа за програмиране на базата на класове, TypeScript се основава на това предложение. JavaScript е компилатор с набор от синтактични разширения на езика, надмножество, основано на предложението, което се превръща в JavaScript. В този смисъл TypeScript е предварителен преглед на това, което да очакваме от ECMAScript 2015. Това е опция за статично въвеждане на текст, който позволява статичен език, който е инструмент и поддръжка на IDE.

## Node.js

Node.js е open source среда за изпълнение на JavaScript, която изпълнява JavaScript код извън браузъра. Исторически, JavaScript е бил използван предимно за скриптове от страна на клиента, в които скриптовете, написани в JavaScript, са вградени в HTML на уеб страницата и се изпълняват от клиент-страна от JavaScript в уеб браузъра на потребителя. Node.js позволява на разработчиците да използват JavaScript, за да записват инструменти на командния ред и скриптове, работещи на сървъра от страна на сървъра, за да създават съдържание от динамично уеб страница, преди страницата да бъде изпратена в уеб браузъра на потребителя. Следователно, Node.js представлява парадигма "JavaScript навсякъде" , обединяваща разработката на уеб приложения около един програмен език, вместо различни езици за скриптове от страна на сървъра и клиента.

Въпреки че .js е конвенционалното разширение на файловото име за кода на JavaScript, името "Node.js" не се отнася за конкретен файл в този контекст и е само името на продукта. Node.js има архитектура, управлявана от събития, която е в състояние да използва асинхронни I / O (input/output) операции. Тези избор на дизайна имат за цел да оптимизират производителността и мащабируемостта в уеб приложения с много входно / изходни операции, както и за уеб приложения в реално време.

Корпоративните потребители на софтуера Node.js включват IBM, LinkedIn, Microsoft, PayPal, Yahoo и други.

## Java

Java е универсален език за компютърно програмиране, базиран на класове, обектно-ориентиран и е разработен специално, за да има възможно най-малко зависимости при внедряването. Той е предназначена да позволи на разработчиците на приложения да " "write once, run anywhere"" (WORA), което означава, че компилиран Java код може да се изпълнява на всички платформи, които поддържат Java.

Java приложения обикновено се компилират до байткод, който може да се изпълнява на всяка Java виртуална машина (JVM), независимо от компютърната архитектура. От 2016 г. Java е един от най-популярните езици за програмиране, които се използват, особено за уеб приложения за клиент-сървър, като 9 милиона разработчици са докладвали. Java първоначално е разработена от Джеймс Гослинг в Sun Microsystems (която оттогава е придобита от Oracle Corporation) и е пусната през 1995 г. като основен компонент на Java платформата на Sun Microsystems. Езикът извлича голяма част от синтаксиса си от C и C ++, но има по-малко съоръжения на ниско ниво от всеки един от тях.

Java е мултипарадигмен език. Използва следните парадигми : Многонишкова, обектно-ориентирана (основана на класове), структурирана, императивна и др. парадигми.

Оригиналните и референтните изпълнения на Java компилатори, виртуални машини и класни библиотеки първоначално са пуснати от Sun под собствени лицензи.

Най-новата версия е Java 10, пусната на 20 март 2018 г. , която следва Java 9 само след шест месеца в съответствие с новия график за пускане. Java 8 все още се поддържа, но няма да има повече актуализации за сигурност за Java 9. Версии, по-ранни от Java 8, се поддържат от компании на търговска основа.

## SHA-256

Хеширане на паролите става чрез SHA-256 хеширащ алгоритъм

SHA ( Secure Hash Algorithms ) са семейство криптографски хеш функции, публикувани от Националния институт по стандарти и технологии (NIST) като САЩ. Федерален стандарт за обработка на информация (FIPS).

SHA-256 и SHA-512. Те се различават по размер на думата; SHA-256 използва 32-битови думи, където SHA-512 използва 64-битови думи. Съществуват също скъсени версии на всеки стандарт, известни като SHA-224, SHA-384, SHA-512/224 и SHA-512/256.

## Spring framework

Pivotal Java Spring Logo.png

Spring е рамка за приложение и инверсия на контролния контейнер за платформата на Java. Основните функции на рамката могат да бъдат намерени на платформата Java EE (Enterprise Edition). Въпреки че рамката не налага специфичен модел за програмиране, тя стана популярна в Java общност като допълнение към или дори за замяна на модела Enterprise JavaBeans (EJB).

Spring Boot е решение на Spring за конфигурация за създаване на самостоятелни производствени приложения, базирани на Spring. Предварително е конфигуриран с "Изглеждания изглед" на Spring на Springfield и библиотеки от third-party system’s, така че да можете да започнете с минимални условия. Повечето приложения за зареждане на Spring изискват много малка конфигурация.

## Apache Maven



Maven е инструмент за build-ване на приложения, използван основно за проекти на Java.

Maven адресира два аспекта на изграждането на софтуер: на първо място, той описва как е построен софтуерът, има нужда от изясняване, и второ, описва зависимостите му. За разлика от по-ранните инструменти като Apache Ant, той използва конвенции за процедурата за създаване и трябва само да бъде записан. XML файлът описва процеса на изграждане, зависимостите му от други компоненти и компоненти, реда на създаване, директории и задължителни приставки. Той идва с предварително определени цели за изпълнение на определени добре дефинирани задачи, като компилиране на код и неговата опаковка.

Maven динамично изтегля Java библиотеки и Maven плъгини от един или повече хранилища(repositories) като Maven 2 Central Repository и ги съхранява в локален кеш.

Този кеш от изтеглените артефакти може да бъде обновен с артефакти, създадени от проекти. Обществените хранилища могат да бъдат актуализирани.

По този начин Maven може да се използва за изграждане и управление на проекти, написани в C#, Ruby, Scala и други езици. Проектът Maven се хоства от фондация "Софтуерна фондация" Apache, където преди това е била част от проекта "Джакарта".

Алтернативни технологии като Maven бяха въведени. С Apache Ivy, специализиран мениджър на зависимости, който се развива, както и подкрепя хранилища на Maven.

## Жизнен цикъл на Maven

Изграждането на жизнен цикъл е процес, който може да се използва за даване или изпълнение на фази. Един от стандартните жизнени цикли на Maven е стандартният жизнен цикъл, който включва следните фази, в този ред.

1 validate

2 generate-sources

3 process-sources

4 generate-resources

5 process-resources

6 compile

7 process-test-sources

8 process-test-resources

9 test-compile

10 test

11 package

12 install

13 deploy

## POM.xml

Project Object Model или POM е основната единица за работа в Maven. Това е XML файл, който съдържа информация за проекта и конфигурацията, използвани от Maven за изграждането на проекта. Когато изпълнява задача или цел, Maven търси POM в текущата директория. Той чете POM, получава необходимата информация за конфигурацията, след което изпълнява целта.

Някои от конфигурациите могат да бъдат зададени в POM са зависимите от проекта, плъгините или целите, които могат да бъдат изпълнени, профилите на изграждане и т.н. Друга информация може да се използва и като версия на проекта, описание, разработчици, пощенски списъци и търсене.

## Apache Tomcat

Apache Tomcat, често наричан Tomcat Server, е отворен Java Servlet Container, разработен от Apache Software Foundation (ASF). Tomcat изпълнява няколко Java EE спецификации, включително Java Servlet, JavaServer Pages (JSP), Java EL и WebSocket, и осигурява "чиста Java" HTTP уеб среда, в която Java код може да работи.

Tomcat е разработен и поддържан от отворена общност от разработчици “Apache Software Foundation” , издадена под лиценза на Apache License 2.0, и е софтуер с отворен код.

## WAR файл

WAR файл (Web Application Resource) е файл, използван за разпространение на JAR файлове, JavaServer Pages, Java Servlets, Java класове, XML файлове, библиотеки за тагове, статични уеб страници и свързани с тях файлове) и други ресурси.

## Предимства на WAR файловете

\* Лесно тестване и внедряване на уеб приложения

\* Лесна идентификация на версията на внедреното приложение

\* Всички контейнери на Java EE поддържат WAR файлове

\* Структурата на MVC поддържа WAR файлове.

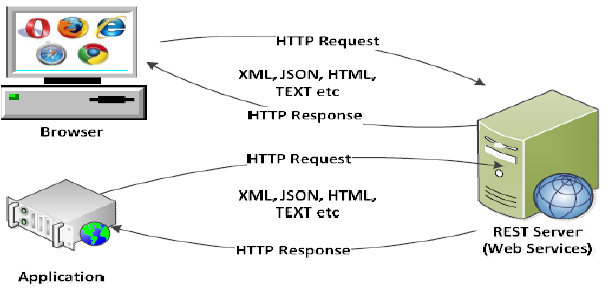
Недостатъци на WAR файловете

Някои смятат, че програмирането чрез използване на WAR файлове за недостатък . Всяка промяна на изходния код трябва да бъде преопакована и развита. Това не изисква уеб сървърът, ако е конфигуриран за разгръщане по време на изпълнение.

## Representational State Transfer (REST) архитектура

Back-end часта е реализирана чрез REST архитектура

Representational State Transfer (REST) е архитектурен стил за създаване на стандарти между компютърните системи в мрежата, което улеснява системите да комуникират помежду си. Системите, съвместими с REST, често наричани RESTful системи, се характеризират с това как са безпартийни и отделни притеснения на клиента и сървъра.



Браузърът прави request към Rest server , който от своя страна връща съобщение, което може да е чист текст или форматирано в някои от следните формати XML, JSON, HTML.

## База данни

**База данни** (БД, още база от данни) представлява колекция от логически свързани данни в конкретна предметна област, които са структурирани по определен начин. В първоначалния смисъл на понятието, използван в компютърната индустрия, базата от данни се състои от записи, подредени систематично, така че компютърна програма да може да извлича информация по зададени критерии. Например БД може да се използват в моделирането на хотелските системи, за да се проверява дали има налични свободни стаи в даден хотел

Поддръжката на база от данни се осъществява от т.нар. Система за управление на база данни (СУБД).

Система за управление на бази данни е компютърно приложение (софтуер) създадено за комуникация между потребителя, други приложения, както и други БД, с цел да се сравнят и анализират данни. Общото специфично предназначение на СУБД е да позволи определянето, създаването, заявки, актуализацията и администрирането на бази данни. Добре известни СУБД включват MySQL, PostgreSQL, Microsoft SQL Server, Oracle, Sybase, SAP HANA, и IBM DB2. Бази данни не са съвместими с различните СУБД, за това различните СУБД работят със стандартни като SQL и ODBC или JDBC, за да позволи на всяко приложение да работи с различни СУБД, а така и с различни БД. Управлението на БД често се избира от модела им, които те подкрепят. Най-използвани системи от бази данни от 1980 г. насам са всички поддържани релационния модели на езика SQL. Често срещано е СУБД да се нарича само „база данни“.

В текущото приложение се използват релационни бази данни.

Релационни бази от данни

През [1970](https://bg.wikipedia.org/wiki/1970) г., когато системите базирани на йерархичния и мрежовия модел са били в разгара на развитието си, [Едгард Франк Код](https://bg.wikipedia.org/w/index.php?title=%D0%95%D0%B4%D0%B3%D0%B0%D1%80%D0%B4_%D0%A4%D1%80%D0%B0%D0%BD%D0%BA_%D0%9A%D0%BE%D0%B4&action=edit&redlink=1) публикува статия, в която предлага разнородните данни да се съхраняват в таблици, което ще позволи да се установят връзки между тях. В днешно време този модел е масово разпространен, но през 1970 г. тази идея е смятана за интелектуален куриоз. Смятало се, че тези таблици не биха могли да бъдат ефективно управлявани от компютър. Този скептицизъм не успял да спре проучванията на Е. Ф. Код. Един от първите прототипи на система за управление на релационни бази от данни (СУРБД) е бил създаден в лабораториите на [IBM](https://bg.wikipedia.org/wiki/IBM).

От 80-те години насам, тази идея се развила и била приета в индустрията. През [1987](https://bg.wikipedia.org/wiki/1987) г. езикът за заявки към БД, [SQL](https://bg.wikipedia.org/wiki/SQL), е стандартизиран. В днешно време, релационният модел е най-масово използвания в [системите за управление на бази от данни](https://bg.wikipedia.org/wiki/%D0%A1%D0%A3%D0%91%D0%94).

## Oracle database

Това е база данни, която обикновено се използва за обработка на обработка на онлайн транзакции (OLTP), съхранение на данни (DW) и смесени (OLTP & DW) бази данни. Най-новото поколение, Oracle Database 18c, е налично в облака, в облака или в хибридна обкръжаваща среда. 18c може също така да бъде внедрена на Oracle Engineered Systems (например Exadata) на Oracle (обществен) Cloud или (частен) Cloud на клиента (например Exadata Cloud at Customer).

Oracle Database 12c се поддържа на следните OS и архитектурни комбинации:

* [Linux](https://en.wikipedia.org/wiki/Linux) on [x86-64](https://en.wikipedia.org/wiki/X86-64) (only [Red Hat Enterprise Linux](https://en.wikipedia.org/wiki/Red_Hat_Enterprise_Linux), [Oracle Linux](https://en.wikipedia.org/wiki/Oracle_Linux) and [SUSE](https://en.wikipedia.org/wiki/SUSE) distributions are supported[[13]](https://en.wikipedia.org/wiki/Oracle_Database#cite_note-13))
* [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) on x86-64
* [Oracle Solaris](https://en.wikipedia.org/wiki/Solaris_(operating_system)) on [SPARC](https://en.wikipedia.org/wiki/SPARC) and x86-64
* [IBM AIX](https://en.wikipedia.org/wiki/IBM_AIX) on [POWER Systems](https://en.wikipedia.org/wiki/IBM_Power_Systems)
* IBM [Linux on z Systems](https://en.wikipedia.org/wiki/Linux_on_z_Systems)
* [HP-UX](https://en.wikipedia.org/wiki/HP-UX) on [Itanium](https://en.wikipedia.org/wiki/Itanium)

През 2008 г. Oracle Corporation обяви наличието на Oracle Exadata Database Machine (V1), първото поколение инженерни системи, специално проектирано за натоварване на Oracle Database.

През 2011 г. Oracle Corporation обяви наличието на Oracle Database Appliance, предварително изграден, предварително настроен, високодостъпен клъстериран сървър на база данни, базиран на два Sunfire X86 сървъра и директно прикачено хранилище.

Някои бази данни на Oracle Enterprise Edition, които се изпълняват от определени хардуерни устройства, предлагани от Oracle, могат да използват хибридна Columnar Compression за по-ефективно съхранение.

# MVC (Model-View-Controller) архитектура



**Модел-Изглед-Контролер** (Model-View-Controller или MVC) е архитектурен шаблон за дизайн в програмирането, основан на разделянето на бизнес логиката от графичния интерфейс и данните в дадено приложение.

* **Модел** - ядрото на приложението, предопределено от областта, за която се разработва; обикновено това са данните от реалния свят, които се моделира и над който се работи - въвеждане, промяна, показване и т.н. Трябва да се прави разлика между реалния обкръжаващ свят и въображаемият абстрактен моделен свят, който е продукт на разума, който се възприема като твърдения, формули, математическа символика, схеми и други помощни средства. Например в банково приложение това са класовете, описващи клиентите, техните сметки, транзакциите, които са осъществили и т.н., както и класовете за извършване на операции над тези обекти (engines) - например клас Transfer с методи като createInterBankTransfer(), createInnerBankTransfer(), getCash() и т.н.
* **Изглед** (View) - тази част от изходния код на приложението, отговорна за показването на данните от модела. Например изгледът може да се състои от [PHP](https://bg.wikipedia.org/wiki/PHP) шаблонни класове, [JSP](https://bg.wikipedia.org/w/index.php?title=JSP&action=edit&redlink=1) страници, [ASP](https://bg.wikipedia.org/wiki/ASP) страници, [JFrame](https://bg.wikipedia.org/w/index.php?title=JFrame&action=edit&redlink=1) наследници в [Swing](https://bg.wikipedia.org/w/index.php?title=Swing&action=edit&redlink=1) приложение. Зависи от това какъв [графичен интерфейс](https://bg.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D0%BD_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) се прави и каква платформа се използва;
* **Контролер** - тази част от сорс кода (клас или библиотека), която взима данните от модела или извиква допълнителни методи върху модела, предварително обработва данните, и чак след това ги дава на изгледа.

## Предимства на MVC

* Моделът е независим от контролера и изгледа.
* Моделът може да бъде планиран и осъществен независимо от другите части на системата.
* За един и същи модел могат да бъдат осъществени различни изгледи (интерфейси) – например уеб интерфейс и нативен интерфейс към Facebook.
* Контролерът и изгледът могат да бъдат променени, без да се налага промяна в модела.

## Недостатък на MVC

* Софтуерната система достига ново ниво на сложност, което я прави по-трудно разбираема за софтуерните разработчици, работещи по нея.

# 3. Проектиране на базата от данни и програмна реализация

Описания на таблиците

\*таблица потребители

\*таблица обяви

\*таблица групи на обяви

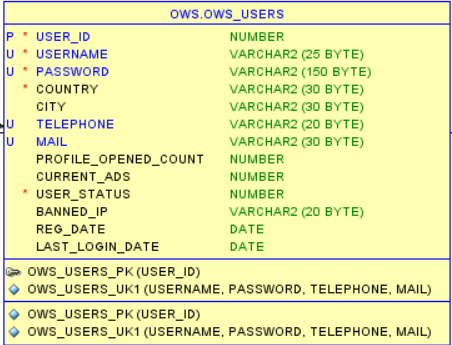
\*таблица снимки

\*таблица харесвания

\*таблица харесвания от потребители

таблица „Потребители“

В таблица потребители, се пазят данните на регистрирани потребители.



**User\_ID** е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя от така наречените sequences, които представляват броячи, дефинирани за всяка таблица.

За всяка първичен ключ от базата е създаден отделен sequence.

**Username** е стринг, и може да бъде най много 25 символа.

**Password** позволява 150 символа, тъй като оригиналната парола въведена от потребителя, се хешира в 60-150 хеш символа.

**Country** е стринг, и може да бъде най много 30 символа. Тя се взема от локацията на която се намира потребителя.

**Telephone** представлява номерът на потребителят. Може да се записват до 20 цифри.

**MAIL** – електронна поща.

**PROFILE\_OPENED\_COUNT** – колко пъти този профил е бил гледан.

**CURRENT\_ADS** – колко обяви има този потребител.

**USER\_STATUS** – определя правата на потребителя(обикновен/администратор)

**REG\_DATE** - дата на регистрация

**LAST\_LOGIN\_DATE** – кога за последно този потребител се е логвал.

Sequence’s : OWS\_USERS\_SEQ

Constraints ( ограничения на таблицата )

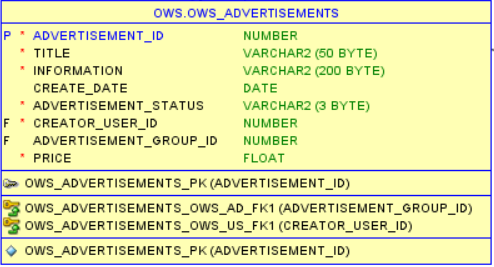
Колони първичен ключ( primary key ) : **User\_ID.**

Колони с уникални стойности : **User\_ID, USERNAME, TELEPHONE,MAIL**

Foreign keys ( външни ключове ):

Таблицата няма външни ключове.

Таблица „Обяви“



Advertisement\_id е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя sequence , които който е дефиниран за тази таблица.

Title – стринг от 50 символа, който пази заглавието на обявата.

Information – стринг от 200 символа, който пази информация за обявата.

Create\_date – дата на създаване.

Advertisement\_status – статус на обявата (създадена,одобрена,изтрита).

Creator\_user\_id – ид на потребителят, създал обявата.

Advertisement\_group\_id – ид на групата към която принадлежи обявата.

Price – цена на обявата.

Sequence’s : OWS\_ADVERTISEMENT\_SEQ

Constraints ( ограничения на таблицата )

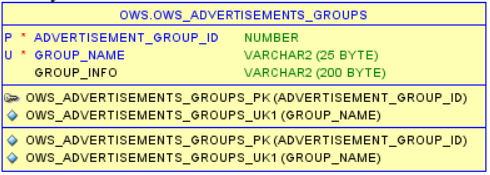
Колони първичен ключ( primary key ) : Advertisement\_id**.**

Foreign keys ( външни ключове ):

Advertisement\_group\_id който референцира ид в таблицата Advertisement\_Groups.

Creator\_user\_id който референцира ид в таблицата USERS.

таблица „групи на обявите“



Advertisement\_group\_id е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя sequence , които който е дефиниран за тази таблица.

Group\_name – стринг от 25 символа, който пази името на групата.

Information – стринг от 200 символа, който пази информация за групата.

Sequence’s : OWS\_ADVERTISEMENT\_GROUPS\_SEQ

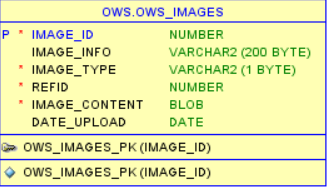
Constraints ( ограничения на таблицата )

Колони първичен ключ( primary key ) : Advertisement\_group\_id**.**

Foreign keys ( външни ключове ):

Таблицата няма външни ключове.

таблица „ снимки“



Image\_id е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя sequence , които който е дефиниран за тази таблица.

Image\_Info – стринг от 20 символа, който пази информация за снимката.

Image\_type – по този тип се разпознава, дали снимката се отнася за потребител или обява.

Refid – записва ид на потребителя, или ид на обявата – и се разпознава в зависимост от колоната Image\_type.

Image\_content – тип BLOB ( binary large object ) – голям масив от байтове, където се записва снимката. При изваждане на снимката от базата към фронт-енда, байтовете се декодират до снимка.

Date-upload – дата на качване на снимката.

Sequence’s : OWS\_Images\_SEQ

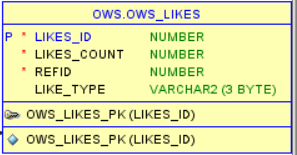
Constraints ( ограничения на таблицата )

Колони първичен ключ( primary key ) : Image\_id**.**

Foreign keys ( външни ключове ):

Таблицата няма външни ключове.

таблица „ харесвания„



Таблицата Likes се използва за да събира харесвания върху обяви или потребители.

Likes\_id е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя sequence , които който е дефиниран за тази таблица.

Likes\_count – съдържа броя на харесвания за този обект( снимка,обява,потребител) .

Refid – записва ид на потребителя, обявата или снимката – и се разпознава в зависимост от колоната Like\_type.

Like\_type – по този тип се определя за какво се отнася записът – снимка, обява, потребител.

Sequence’s : OWS\_Likes\_SEQ

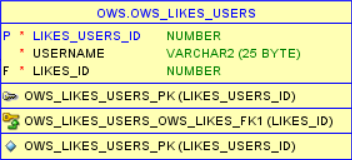
Constraints ( ограничения на таблицата )

Колони първичен ключ( primary key ) : Likes\_id**.**

Foreign keys ( външни ключове ):

Таблицата няма външни ключове.

таблица „ харесвания от потребители „



Таблицата Likes\_users се използва за да вижда за дадена обява,потребител – кои точно потребители са харесали.

Likes\_users\_id е първичен ключ, който се генерира при всяко добавяне на нов ред в таблицата.

Неговата стойност се определя sequence , които който е дефиниран за тази таблица.

Username – името на потребителя харесал .

Likes\_id – референцира запис в таблица LIKES.

Sequence’s : OWS\_Likes\_users\_SEQ

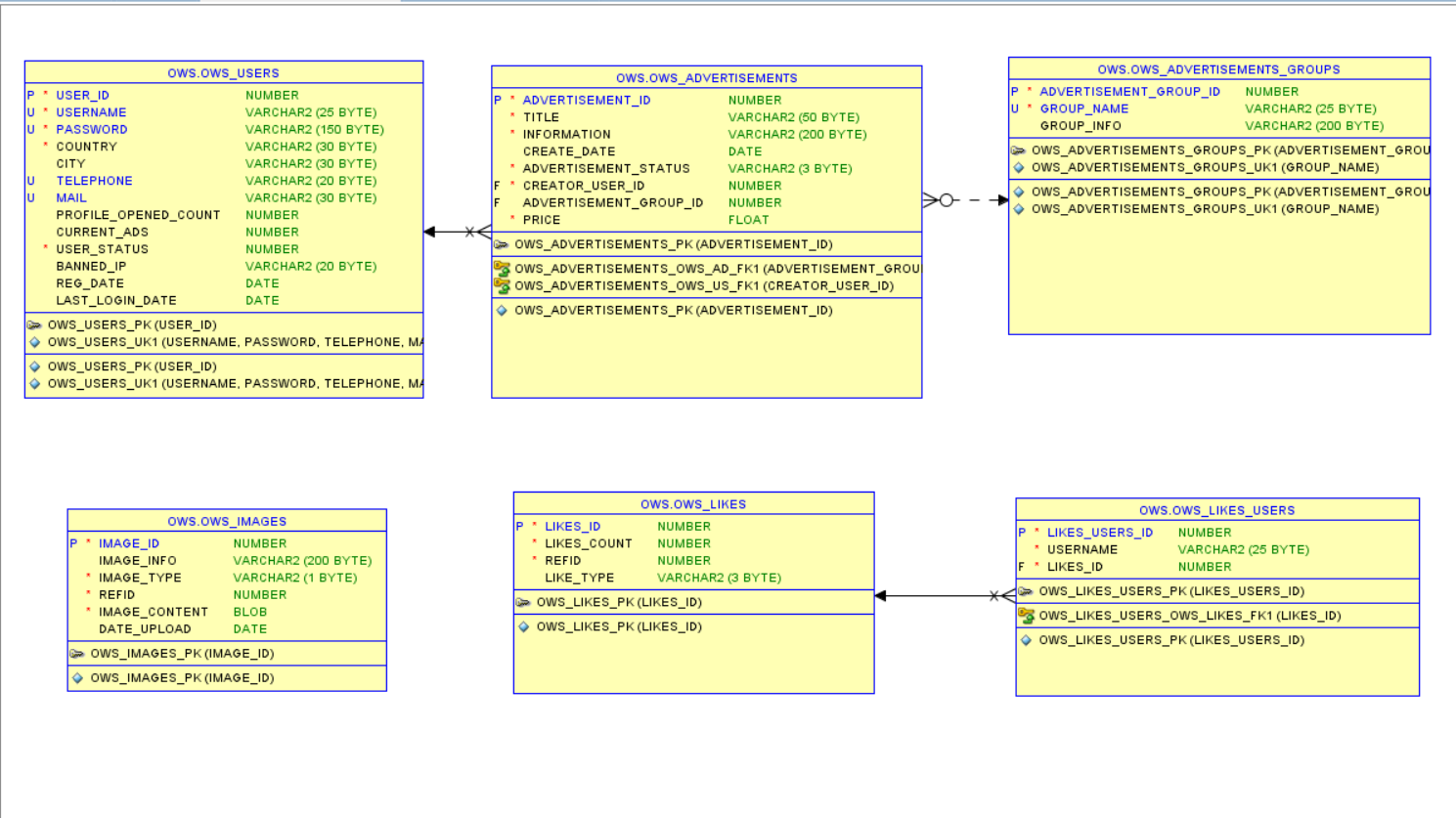
Constraints ( ограничения на таблицата )

Колони първичен ключ( primary key ) : Likes\_users\_id**.**

Foreign keys ( външни ключове ):

LIKES\_ID референцира запис в таблица LIKES, по която може да се определи точно кои потребители са харесали тази обява/снимка/потребител.

ER диаграма на базата данни



Програмна реализация

Проектът е реализиран от два отделни проекта, които си комуникират по-между си:

- клиентска част (фронт-енд) – online-web-shop-AngularClient

- сървърна част (бак-енд) - online-web-shop-be-server

Клиентска част (front-end) – имплеметация

За фронт-енд частта е използвана популярният framework – Angular.

Чрез Node.js 8.0 се инсталира angular/cli – client line interface ,чрез който се създават angular проекти.

При създаването на нов проект, angular/cli създава всички модули и нужни файлове, които са необходими за започването на нов проект.

Имплементация на приложението :

Приложението имплементира няколко компонента, като всеки компонент съдържа в себе си html съдържание ( view ) , .ts файл ( controller ) модели които се използват в контролера и .css файлове за стиловете на страничката.

Компоненти

RegisterLogin компонент

RegisterLogin компонент съдържа html съдържанието на страницата за логване и регистриране на потребители в зависимост от това, какво потребителят е избрал да направи, за валидация на входящите данни, стиловете на страничката и http call към бак-енд частта за запазване на новата обява.. При успешна регистрация, приложението логва автоматично.

Add advertisement компонент

Компонент за добавяне на нова обява. Видимо е само за обикновеният потребител. Администраторът не може да достъпва този компонент(екран), тъй като той е предназначен само за одобрение на обяви.

Компонентът съдържа html съдържанието на страницата добавяне на нова обява,валидация на полетата и http call към бак-енд частта за запазване на новата обява.

home компонент

Компонент, който е видим от всички потребители, показващ одобрените от администратори обяви и тяхната информация. Потребилят има право да дава харесване, или да маха такова. Като всичко става чрез Ajax заявки и се обновява динамично.

my-advertisement компонент

Компонент, който показва обявите на текущият логнат потребител. Потребителят има правото да изтрие обява през този компонент.

profile компонент

Компонент, който показва информация за потребител, данните му и негови обяви.

Admin компонент

Компонент, който е видим само за потребители със статус „2“ – админ.

Този тип потребител може да одобрява или да отхвърля нови обяви.

Като ако одобрява обява, трябва да избере групата към която тази обява ще принадлежи. Ако няма подходяща група, то администратора може да създаде такава.

create-new-advertisement-group компонент

Компонент, който е видим само за потребители със статус „2“ – админ.

Този тип потребител може да създава нови групи при необходимост. Когато няма подходяща група за нова обява.

Advertisement-full-information компонент

Компонент, който показва пълната информация за дадена обява.

Search-by-advertisement-group компонент

Компонент, който показва обяви, филтрирани по избрана група с цел по бързо откриване на желаното.

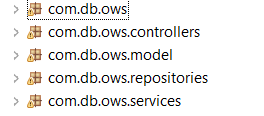
Всичките заявки към бак-енд сървъра са асинхронни, и данните се обновявав динамично.

Сървърна част

Сървъртната част е реализирана с Java 8 + spring boot framework, използвайки REST архитектура. Всеки request от front-end-a идва под формата на JSON формат, като от това съобщение spring boot си намапва полетата на съотведния java дефиниран модел.

Ако всичко с request-а е валидно, сървърът прави необходимата поискана операция и взема/добавя/обновява/изтрива нещо от базата данни.

Дефинират че 5 отделни пакета със сорс-код за приложението.



com.db.ows пакета, държи стартиращата логика на приложението. Main функция, както и допълнителна конфигурация за spring boot приложение.

Back-end REST контролери

Com.db.ows.contrllers са пакетирани сорвовете, класифицирани като rest контролери.

Всички контролери са регистрирани като @RestController spring bean.

Дефинирани са 3 REST контролера :

Login контролер

Той имплементира логиката, по която потребителят се логва в приложението, регистрира или изпраща забравена парола ( ако е необходимо ).

Admin контролер

Той имплементира логиката, при която администратора може да одобрява обяви, да ги отхвърля , както и да създава нова група в която да добавя нови обяви.

Edit контролер

Имплементира логиките, които се изпълняват от логнатият потребител – четене на обяви, търсене на обяви по група, създаване на обява, изтриване на обява, четене на обявите на логнатият потребител , добавяне/махане на харесване и др.

Всички контролери съдържат в себе си service-ите от com.db.ows.services като spring beans.

За всеки валиден request, контролерът вика необходимият му сървис за извършване на операцията.

Back-end модели

Дефинирани са необходимите модели, които приложението изисква : User , Advertisement, AdvertisementGroup , Image, Like и др.

Декларирани Enum’s : AdvertisementStatus, DatabaseSequences , LikeTypes, ImageType, RegistrationState и др.

Back-end services

Всички service’s са регистрирани като @Services spring bean.

Сървисите, услугите извикват необходимото repository, където се правят и изпълняват заявки към базата данни.

Дефинирани са няколко сървиса : AdminService, AdvertisementService, ImageService, LikeService и LoginService.

В тези сървиси, е дефиниран съответният им repository bean.

Back-end repositories

Всички repositories са регистрирани като @Repository spring bean.

Дефинирани са няколко repository : AdminRepository, AdvertisementRepository, ImageRepository, LikeRepository и LoginRepository както и SequenceRepository.

SequenceRepository е дефинирано с цел по добра четимост на сорс-кода. Тъй като в базата трябва да се взема за всяка таблица, следващата стойност от sequence-a. Вместо да се прави това във всяко repository отделно, е създадено отделно SequenceRepository което да отговаря само за тази работа.

Във всяко repository е дефиниран така нареченият NamedParameterJdbcTemplate като spring bean.

Този интерфейс отговаря за връзката с базата, и през него се изпращат заявките към нея. По време на създаването му, spring boot търси неговите конфигурации, които се дефинират в application.properties на приложението.

Нещата които трябват за да се конфигурира връзката с базата са :

spring.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver

spring.datasource.url=jdbc:oracle:thin:@localhost:1521:orcl

spring.datasource.username=username

spring.datasource.password=password

Във всеки от горе изброените Repository файлове, има методи които комуникират с базата.

SQL заявките се конструират чрез StringBuilder-и, в зависимост от входящите параметри.

Всички сорс файлове в проекта следват висока кохезия(strong cohesion), за да може софтуера да е по четим, по надежден и по лесен за поддръжка.

Кохезия

**Кохезия** (вътрешна съгласуваност) в [програмирането](https://bg.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B8%D1%80%D0%B0%D0%BD%D0%B5) представлява измерител за степента, в която отделните редове програмен [код](https://bg.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4) в даден [модул](https://bg.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB_(%D1%81%D0%BE%D1%84%D1%82%D1%83%D0%B5%D1%80)) се сработват за да предоставят специфична функционалност. Кохезията е [ординален](https://bg.wikipedia.org/w/index.php?title=%D0%9E%D1%80%D0%B4%D0%B8%D0%BD%D0%B0%D0%BB%D0%BD%D0%BE%D1%81%D1%82&action=edit&redlink=1) [измерител](https://bg.wikipedia.org/w/index.php?title=%D0%98%D0%B7%D0%BC%D0%B5%D1%80%D0%B8%D1%82%D0%B5%D0%BB&action=edit&redlink=1) и обикновено се характеризира като „висока“ и „ниска“. Тенденцията е да се предпочитат модули с висока кохезия, защото тя е свързана с желани черти на софтуера като устойчивост, сигурност, [надеждност](https://bg.wikipedia.org/wiki/%D0%9D%D0%B0%D0%B4%D0%B5%D0%B6%D0%B4%D0%BD%D0%BE%D1%81%D1%82_%D0%BD%D0%B0_%D1%81%D0%BE%D1%84%D1%82%D1%83%D0%B5%D1%80%D0%B0), лесна разбираемост. И обратно — модул с ниска кохезия е труден за поддръжка и разширение, труден за повторно използване и разбиране.

AdminController – са имплементирани само функционалностите, които само админът може да изпълнява.

LoginController - са имплементирани само функционалностите за логване и регистрация.

EditController – са имплементирани функционалностите, които всеки обикновен потребител може да изпълнява.

AdvertisementRepository – само SQL заявки за управление на таблицата на обявите : добавяне/изтриване/промяна.

ImageRepository – само SQL заявки за управление на таблицата на снимките : добавяне/изтриване/промяна.

SequnceRepository – само SQL заявки за вземане на следващия номер от sequence на таблица.

AdminRepository – само SQL заявки, свързани с администратора : одобрение на обява/ отхвърляне на обява/създаване на нова група.

LikeRepository – само SQL заявки за управление на таблицата на харесванията : добавяне/премахване на харесване.

LoginRepository – само SQL заявки за управление на таблицата на потребители : регистриране/логване/ забравена парола.

При записване на нова парола за потребител в базата, се използва хеширащ алгоритъм **SHA-256.**

Този алгоритъм идва чрез Maven dependency guava:

<dependency>

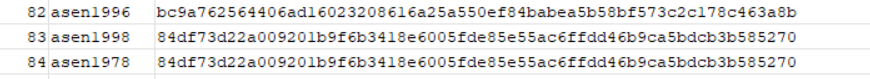
<groupId>com.google.guava</groupId>

<artifactId>guava</artifactId>

<version>20.0</version>

</dependency>

Хешът получен от този алгоритъм се записва в базата. Излежда по този начин



При всеки опит за логин, паролата въведена от потребителя се хешира със същият алгоритъм и се сравнява с хеша от базата. Ако те съвпадат , потребителят се логва успешно.

# 4.Ръководство на потребителя

## 

## Роли на потребителя

Потребителят може да бъде регистриран, не регистриран и администратор.

Не регистрираният потребител, може само да разглежда обяви, но не и да дава харесване, нито да създава нови обяви.

Регистрираният потребител, може да гледа обяви, да създава свои, да им следи статуса и да ги трие при извършена поръчка.

Администратора може да разглежда обяви, да одобрява обяви от други хора, да ги отхвърля. Администратора не може да създава обяви.

Администратор може да се стане, само чрез свързване със собственика на уеб сайта.

## Екрани и техните функционалности

В home екрана се показват всички одобрени от администратор обяви. Показват се в съкратен вид – снимка, заглавие, цена и др.

При кликване върху обявата се отваря пълното и съдържание.

В create advertisement екрана, потребителят трябва да попълни необходимата информация за новата обява , и да качи снимка(по – избор).

Advanced Search екрана, може да се показват обявите от дадена група и да се филтрира по най-ниска/най-висока цена, както и най-стари/най-нови обяви.

My-advertisement екрана е видим за логнат потребител. В него се показват обяви, публикувани от логнатият потребител – ако има такива, както и техният статус в текущият момент.

Approval екрана е видим само за администратори – за одобряване и отхвърляне на обяви.

New group екрана е видим само за администратори – за създаване на нови групи.

Обявите могат да се харесват само по веднъж от даден потребител – както и да се маха харесването на по-късен етап.

Ако даден потребител хареса дадена обява, то той може да види потребителят който я е публикувал и да го потърси на номерът му, или да му пише мейл.

# 5. Приложения

Регистриране/логин

Регистриране



При регистрация, потребителят трябва да въведе потребителско име, парола, снимка(по желание), страна в която е роден, град, телефон и мейл.

Когато всички тези данни са налице и са валидни, потребителят се регистрира успешно.

Ако има невалидни полета обаче, под всяко невалидно поле излиза текст „ подсказка“, че даденото поле не е валидно. Потребителят поправя невалидните данни и опитва да се регистрира. Angular изпраща ajax call към сървърната част и ако потребител с такова име вече съществува, излиза съобщение под бутона регистер като данните попълнени до този момент се пазят, и потребителят трябва да въведе единствено ново потребителско име и да опита отново. Ако регистрирането е успешно, приложението логва новорегистриралият се потребител автоматично и му отваря home екрана с обявите.

Паролата се пази в базата данни криптирана. Алгоритъмът за криптиране е SHA-256.

При натискане на бутона “Already register”се отваря логин формата.

Логин



При този случай, потребителят въвежда името и паролата си. Java сървърът криптира паролата чрез SHA-256 и сравнява с криптираната парола в базата. Ако такива съществуват в базата, потребителят се логва успешно.

При натискане на бутона “not registered” се отваря формата за регистрация.

При натискане на бутона „Forgotten password”, пред потребителят се показва форма, в която той трябва да въведе username, telephone & mail с които се е регистрирал. Ако такива съществуват в базата, то той ще получи на съответният mail новата парола.

## Забравена парола



При натискане на Reset password, ако всичко е въведено правилно излиза съобщение „ new password has been send to the mail , please check ”.

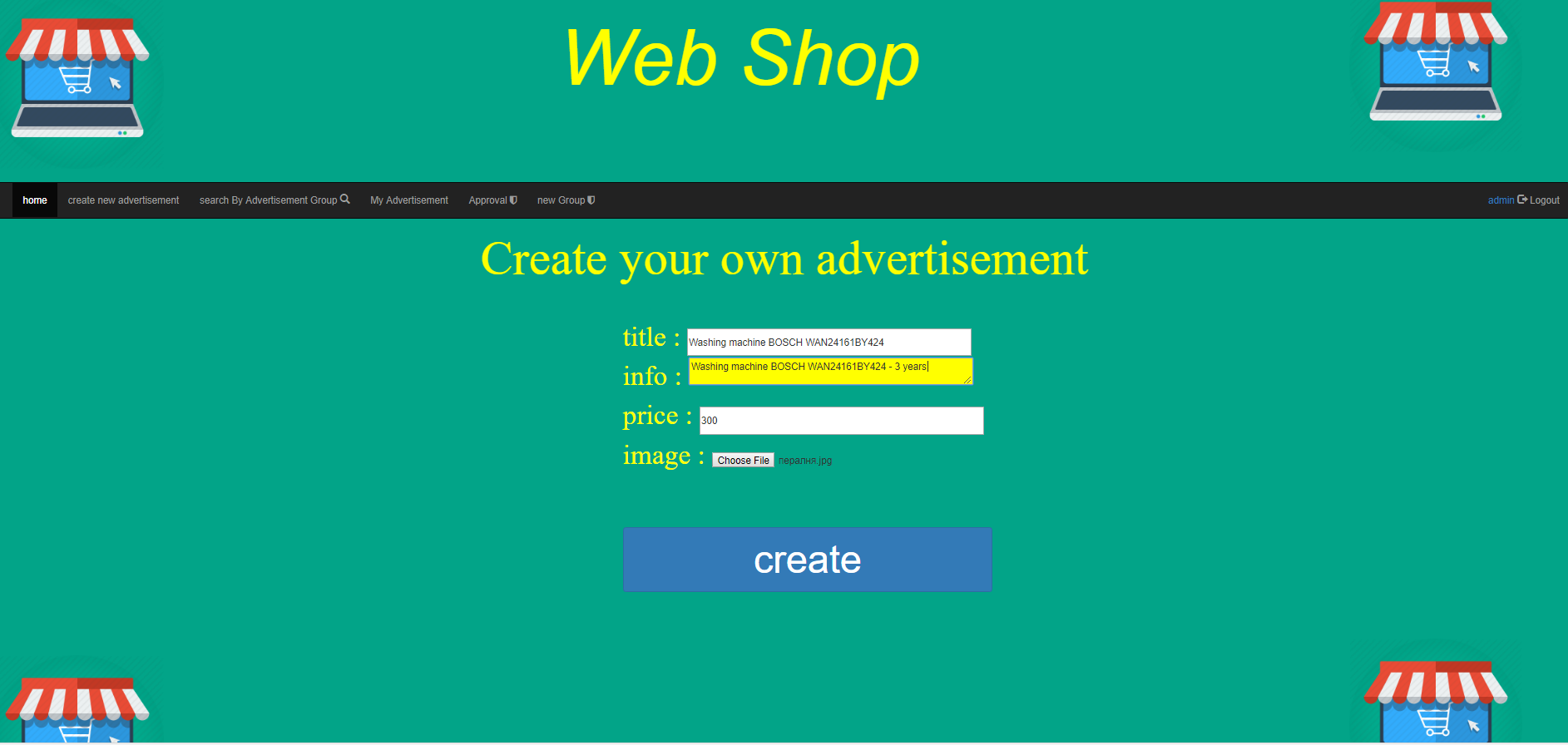
При натискане на Back бутона, потребителя се връща обратно в логин формата.

Създаване на обява

Приложението представлява платформа за предлагане на различни продукти и услуги.

Всяка валидна обява има – заглавие, информация, дата на създаване, статус, цена , снимки, харесвания и информация за човека създал обявата.

За всеки продукт/услуга, потребителя попълва следната информация :



\*Заглавие-задължително.

\*Допълнителна информация – описание на продукта/услугата.

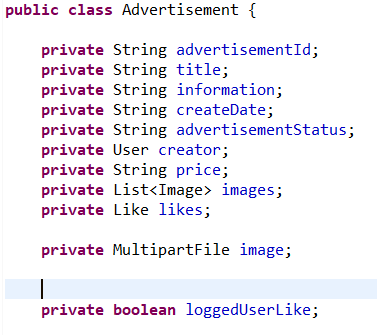
\*Снимка(1 или повече)- не е задължителна.

\*Цена(лв)- задължително.

Бутонът CREATE е заключен, и се отключва когато са валидни горе-посочените изисквания.

Front-end програмата изпраща събраните данни форматирани в JSON формат към съответният REST контролер на Back-end частта.

Back-end сървъра получава JSON формата и си го намапва в свой модел Advertisement.java



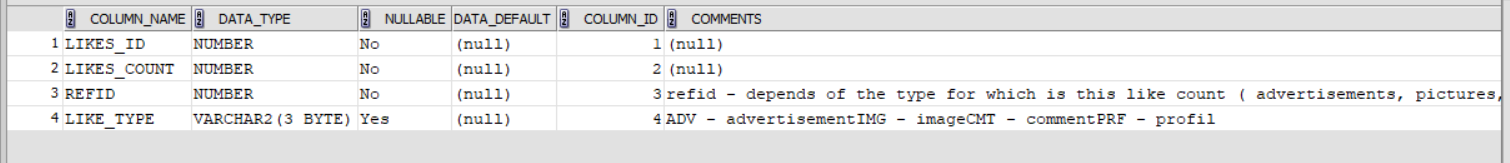
Намапването става чрез Spring boot framework – който използва съответните генери и сетери на полетата от класа.

От тук нататък Java сървърът се обръща към базата с INSERT заявка и подава информацията за обявата която трябва да бъде записана. Тази информация се записва в таблица OWS\_ADVERTISEMENT

След като бъде запазена тази информация, Java сървърът прави още една INSERT заявка за снимките които са добавени към обявата. Снимките се запазват в отделна таблица **OWS\_IMAGES:**

След като запази снимките – java сървърът прави още една INSERT заявка за харесванията на новата обява.

Харесванията се запазват в отделна таблица **OWS\_LIKES:**

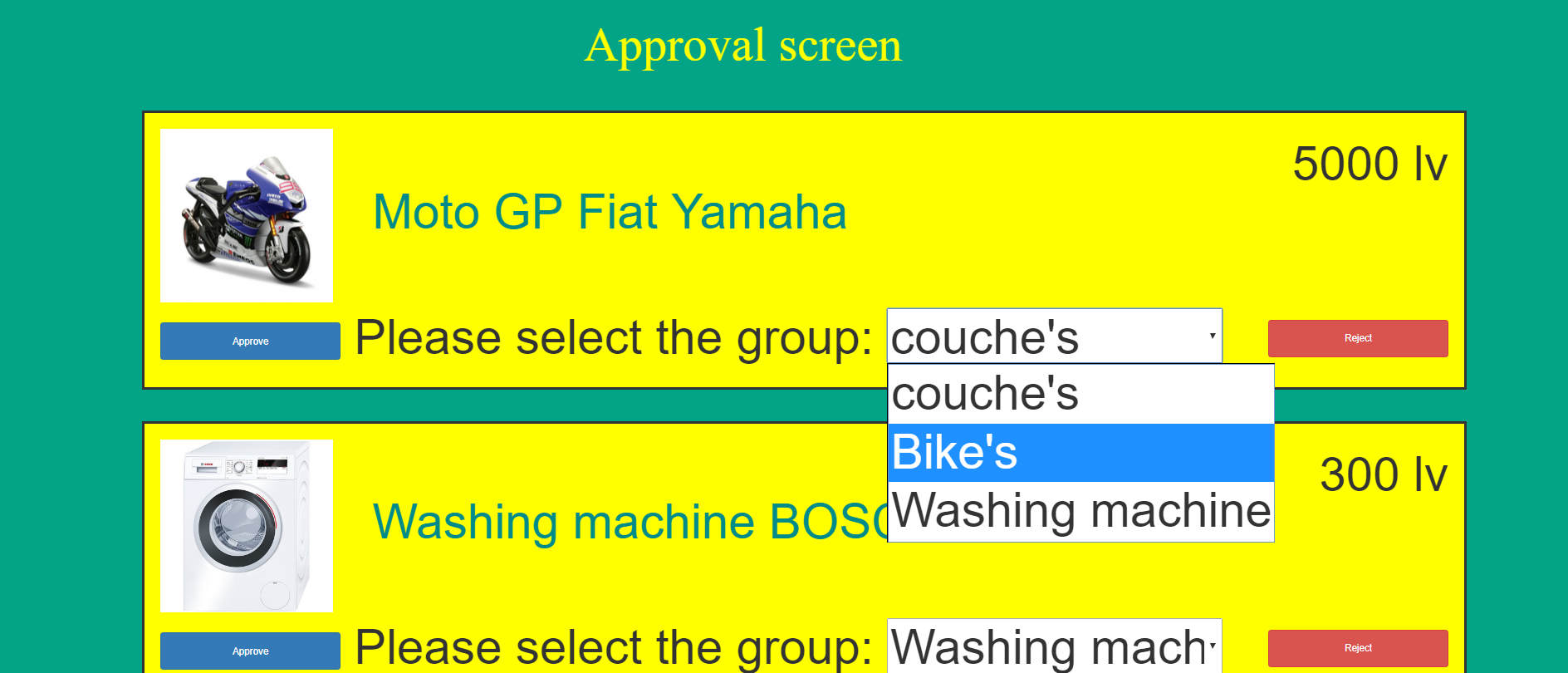


След въвеждането на цялата тази информация, обявата става видима в административния екран – в който административно лице решава дали тази обява е валидна и ще стане видима за аудиторията.

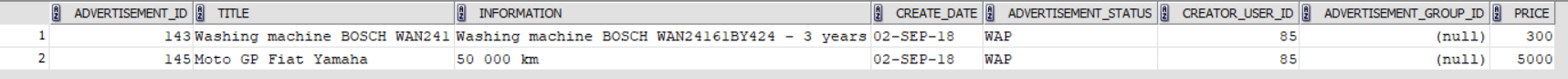
Ако бъде одобрена от административно лице, обявата става видима. Ако бъде отхвърлена обаче, тя се появява в екранчето My advertisements със статус – rejected.

Административна част

Даден потребител може да бъде административно лице. Като административно лице, той има право да одобрява и отхвърля обяви, да избира група към която принадлежи конкретната обява и да създава нови групи ако се налага. Потребител може да получи администраторски права само по договорка.



В базата данни:



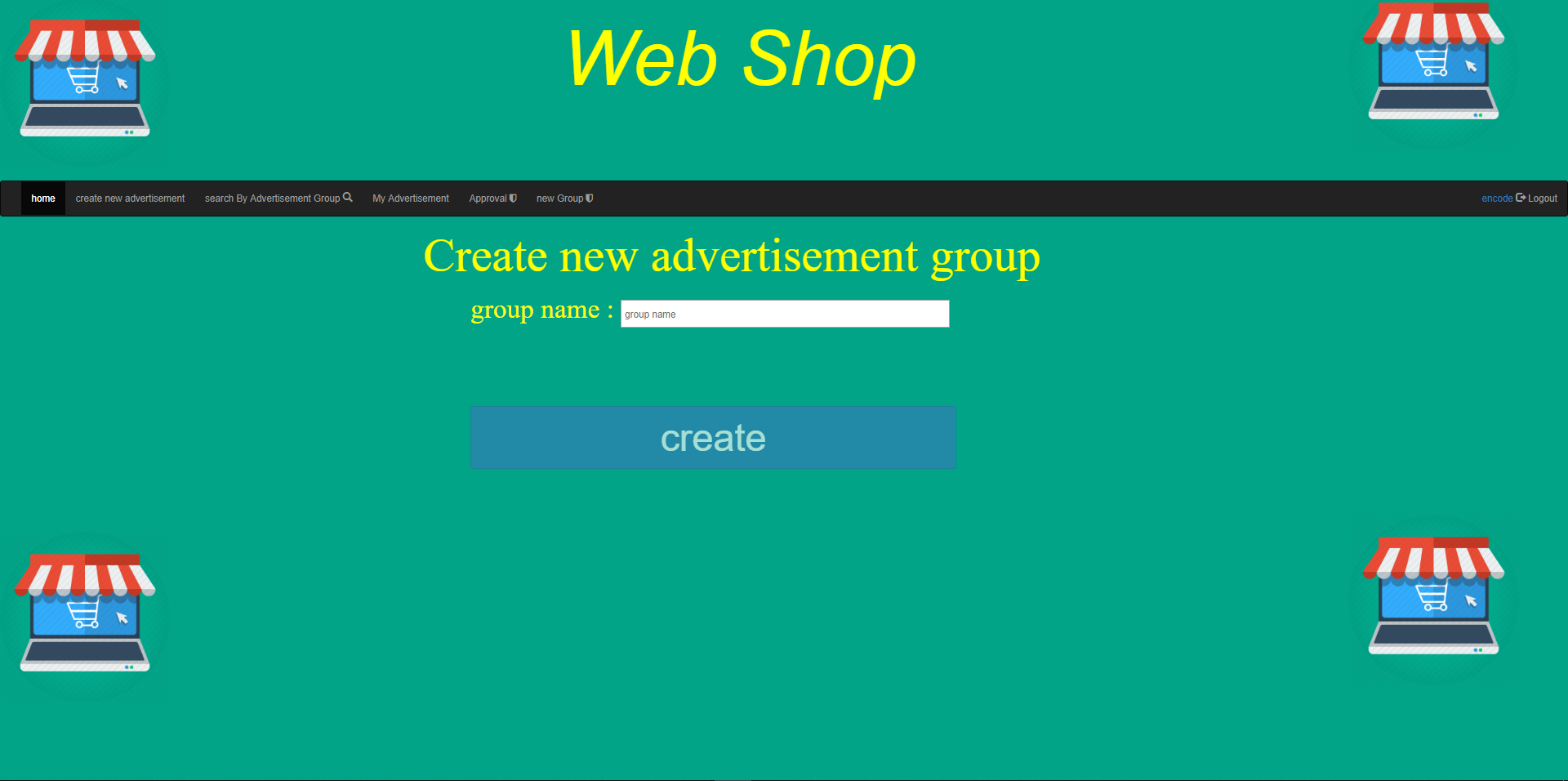
Обявите със статус „ чакащи одобрение – WAP „ се показват в този екран със съкратена информация. От тук администраторското лице може да отвори дадена обява, като кликне на заглавието и, и да прочете пълната информация на обявата, да одобри обявата като избере съответната група в която да я помести, или да я отхвърли.

Ако натисне бутона Approve без да е избрал групата, селект менюто ще светне.

След като approve-не, обявата става видима за всички в home екрана.



Ако няма подходяща създадена група в която обявата трябва да бъде поместена, администраторското лице може да създаде нова група от менюто New Group, което е видимо само за администраторски лица.



Бутонът create е заключен докато не се въведе валидно име за група.

След като се натисне create , групата е създадена и администраторското лице може да се върне при обявите за одобрение. Новосъздадената група вече ще присъства.

Като допълнителна анимация, когато цъкнем в дадено поле, то се оцветява в жълто.

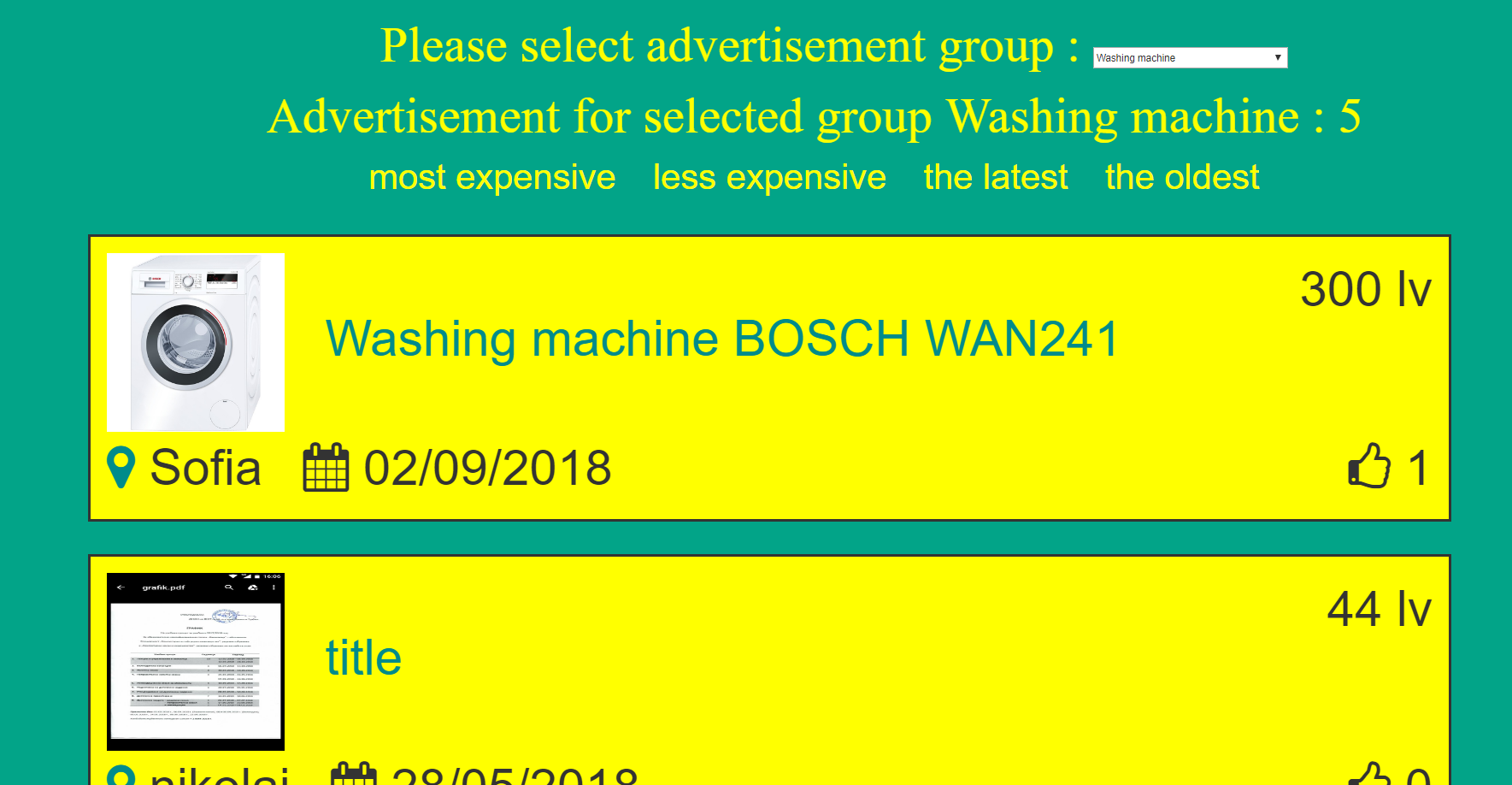


Търсене по група

Понякога потребителите биха искали да търсят конкретен вид продукти/услуги. Тук на помощ идва функционалност да се търси по група.



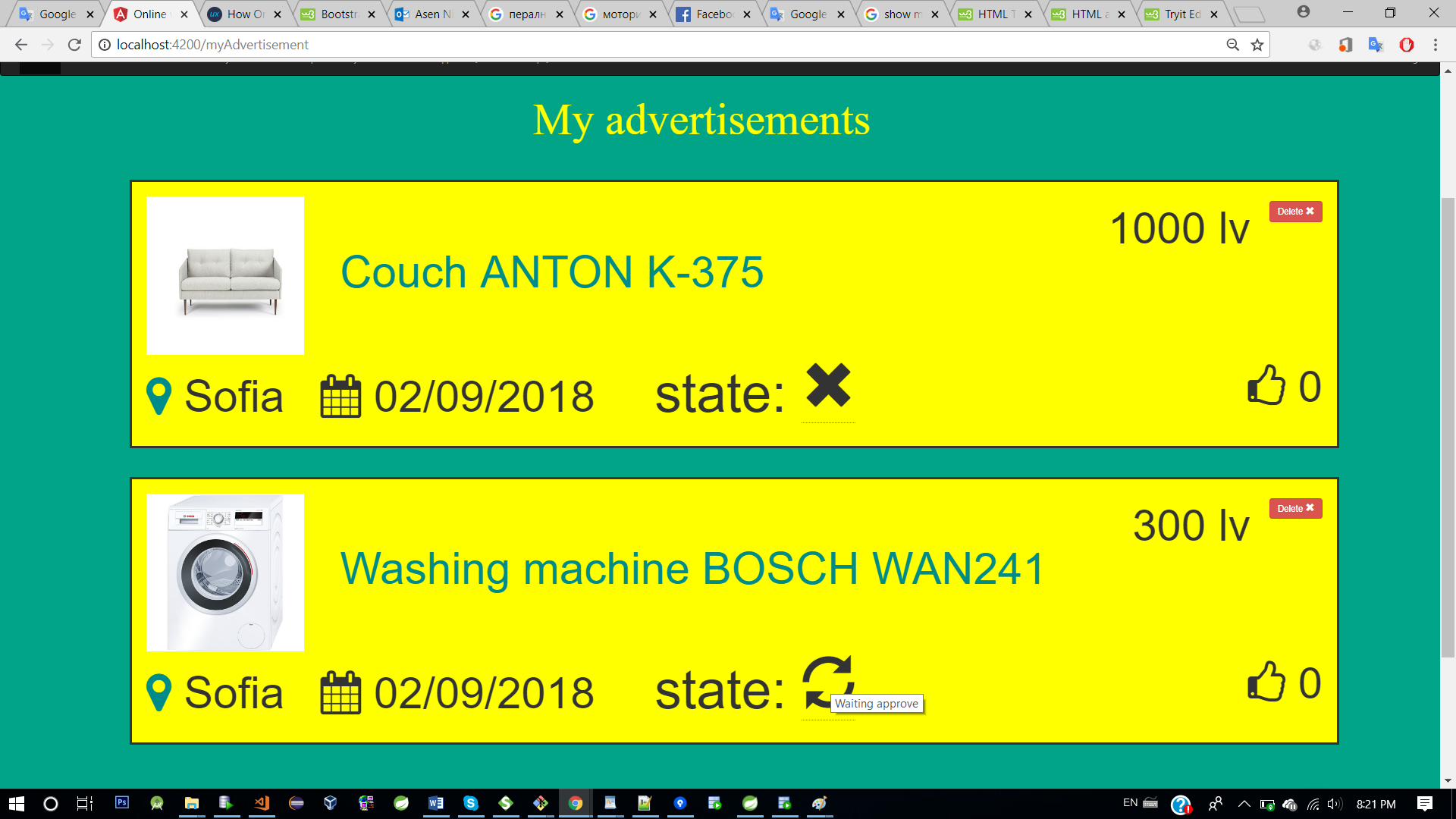
По този начин търсенето е много по опростено, тъй като потребителят се съсредоточава върху един клас продукти/услуги.



При търсенето има и 4 филтъра за по качествено търсене. За най ниска цена, най висока цена, най стари и най нови обяви.с

Моите обяви

Всеки потребител има екранче, което му показва текущия статус на неговите обяви (чакащи/одобрени/отхвърлени).



Показана е иконка, която е в замисимост от статуса на обявата в момента. При посочване на иконката се показва подсказващ текст.

Чрез бутонът DELETE, потребителят може да изтрие своите обяви.