**ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

Тема: Търговски софтуер за създаване и обработка на поръчки

Дипломант: Научен ръководител:

*Иван Драганов Юлиан Линев*

СОФИЯ

2019

**Увод**

Връзката ни с дигиталните технологии се вплита все по-дълбоко в живота ни от гласовите помощници на Гугъл и Майкриософт до разпознаването на снимки във Фейсбук и VK.com. В забързаното ни ежедневие потребителите очакват всичко да се случва с един клик и минимално усилие.

Въпреки това все още използваме брошури, периодични издания и рекламни материали на физически носители, остарeли идеи, парадоксално запазили се в едни от най-динамичните отрасли – търговия и логистика. Негативите са сериозни както за бизнеса, така и за околната среда - тромава структура за вмъкване на нови продукти, както и екологичното замърсяване и нуждата от голям брой хора за поддръжка на услугите.

От друга страна, много хора не се престрашават да встъпят в света на дигиталните технологии, несигурност, която лесно ще бъде преодоляна с ергономичен дизайн и бързодействие на софтуера. Такъв тип програма ще изиграе ролята на междинно стъпало между онлайн търговията и физическата такава – често искаме да поръчаме или запазим артикул през интернет, но да го закупим по „класическия“ начин – посещавайки магазина и преглеждайки артикула на място, всичко това в удобно за клиента място и време. Такъв „хибриден“ подход помага и за разрешаване на проблеми, както свързани със самия продукт, така и в случай, че се получават чрез куриер. Не на последно място ползването на програма ще позволи бързото създаване и проследяване на доставки, направени от клиентите на даден магазин.

Проектът е алтернативен вариант за интернет пазаруване, вдъхновен, но неограничаващ се с иновативните идеи за онлайн търговия на Amazon, чрез “Alexa” и плеядата от устройства, работещи с него. Тъй като Югоизточна Европа и България в частност не са развит пазар, системите на Amazon, Google и Microsoft не са подходящи, тъй като обвързват крайния клиент със серия продукти, които или не се предлагат на нашия пазар, или са комбинирани със скъпи устройства. Нашият продукт предлага гъвкавостта на приложениеза смарт устройство (Android), които така или иначе са широко разпространени и използвани и изключват първоначалната инвестиция за физическо устройство.

Потребителите имат възможност с едно натискане на бутон да сканират предоставен QR код и да поръчат или запазят в обект дадените продукти. За търговците и производителите това е предпоставка за свобода при създаването и разпространението на QR кодове, било то онлайн или на хартиен носител. Този по-нестандартен начин на пазаруване го прави интересен ,което спомага за по-добра връзка клиент – магазин.

# ПЪРВА ГЛАВА

# Обзор за съществуващите решения

## **1.1 Подобни съществуващи решения**

Много от магазините и търговските обекти в днешно време имат своя онлайн платформа, предлагаща различни услуги. В основата си всички те целят широка приложимост и бързодействие. Благодарение на иновативен дизайн,голяма функционалност и предоставената информация, работата с клиенти прогресивно се подобрява, което допринася за добро развитие на бизнеса. Подобни търговски софтуери са Microinvest Склад Pro и Mistral.

Microinvest Склад Pro е гъвкава система за управление на стоковите наличности и парични потоци на фирмите. Програмата позволява да бъдат контролирани всички процеси, свързани с управление на производствената и търговската дейност. С нея можете да оцените постигнатото от Вашия бизнес чрез подробни справки и статистически отчети. Едновременно с това с програмата се извършва пълен мониторинг върху дейността на персонала чрез регистрирането на всяко действие и операция, извършени от него. Системата позволява да се проследи във всеки един момент количеството използвани запаси и да се планират необходимите разходи.

От друга страна, няколко от софтуерните решения на Mistral предлагат мобилни справки чрез смартфон (на устройството се визуализира информация за стоковата наличност,оборот,**зети** маси и доставки и др. ). Също така кухненски дисплей – поръчките се появяват мигновено на дисплей в кухнята след подаването им от сервитьор или от посещаващите онлайн платформата за поръчки. Готвачът има възможност интерактивно да променя рецептата при липса на продукт и наличие на заместващ.

Има много подобни и интересни софтуерни решения на пазара, но това, с което се различава представеното, е неговият начин на работа. Потребителят се нуждае единствено от смарт телефон, който вече е нещо,което всеки притежава. Чрез сканиране на QR код с камерата на устройството автоматично се добавя продукт в количката на клиента и той получава информация за него, а благодарение на иновативния си интерфейс с един клик може да се изпрати поръчка към конкретен магазин. От друга страна, частта на продукта, предназначена за търговеца, предоставя лесен начин за създаване на QR кодове, добавяне на артикули към база данни и получаването и обработването на поръчки – всичко това на един екран!

## **1.2 Развойни средства и среди. Проучване.**

Чрез проучване и разговори с хора, свързани пряко с търговията и търговски магазини, достигнах до идеята за направата на интересна и нестандартна система за поръчки. Със съвместна работа с програмисти, разработващи търговски софтуери на професионално ниво, получих необходимите знания, за да започна работа, а именно- какви параметри да съдържа един продукт, поръчка, клиент и правилен начин на функциониране и структуриране на програмата и базата данни.

Този софтуер може да се развие много. Има добра основа,в която могат да се вграждат нови функционалности, свързани с търговията както за търговеца, така и за купувача.

Системата се състои от три софтуерни части разпределени по така наречената „3-Tier Architecture“, където програмите се групират в три групи – презентативна, логическа , информативна. В първата влизат технологийте, които вижда крайният потребител, като интерфейси и визуална част. Във втората се съхранява логиката и командите на цялата система и служи като връзка между останалите две. Информацията се съхранява и взима от информативната група.

Първият и най-важен фрагмент от продукта, това е уеб услугата, в която са заложени всички взаимодействия с базата, под формата на HTTP заявки. Всяко действие от другите две програми, например добавяне на нов артикул, потребител, изтриване на поръчка или нещо друго, изпраща заявка към уеб услугата, чрез която се извършва конкретна работа, свързана с базата данни.

Втората част е настолно приложение, което е предназначено за търговеца или търговския обект. Там служителите въвеждат данни за продукти,предлагани от магазина и генерират QR кодове за тях. В друг прозорец в същото време могат да следят новите поръчки и да ги обработват.

Третият сегмент представлява андройд приложение, определено за клиенти на дадения обект. Функционалността му позволява разчитане на QR кодове чрез камерата на мобилно смарт устройство. След сканиране артикул, отговарящ на информацията от QR кода, автоматично се добавя в количката, където може да се коригира количеството му. Също така на екрана се появява повече информация за него. Може да се изпрати поръчка по всяко време и да следи нейният статус.

Aндроид и десктоп приложенията комуникират с уеб услугата, но не и помежду си. От своя страна тя работи с базата данни. Цялата връзка се осъществява посредством интернет.

Базата данни е MSSQL, изградена с помощта на code-first метод, наличен в “Entity Framework Core”.

Entity Framework представя Code-First метода с Entity Framework 4.1. Чрез Code-First, програмистът може първо да се фокусира върху основите на неговото приложение и да започне да създава необходимите за него базови класове, вместо първо да създаде база данни и след това по нея да направи необходимите такива. В общия случай уеб частта ще създаде базата данни спрямо моделите на приложението и конфигурацията между тях. Това означава че първо трябва да се напише кода на C# или VB.NET и след това Entity Framework ще създаде необходимата база от него. За разлика от Database-first подхода, Code-first ни дава пълен конрол върху кода, вместо работа в базата. С него можем да изпълняваме всички операции в базата чрез код и най-вече при необходимост от промяна на базовите класове, автоматично ни се проустройва и структурата на базата данни.

# ВТОРА ГЛАВА

# ПРОЕКТИРАНЕ НА СТРУКТУРАТА

## **2.1. Функционални изисквания**

Системита за клиент и тази за търговец са независими една от друга и комуникират единствено с уебуслугата, която от своя страна извършва определени операции с базата.

Андроид приложението дава възможност на потребителите да се впишат в свой профил или да си създадът такъв, ако нямат. След успешно вписване в програмата клиентът може да веднага да започне да сканира съответните QR кодове, което автоматично ще добавя продуктите в кошницата. Там е възможно да се коригира количеството на дадения артикул, а над нея се изписва информация за него. По всяко едно време може да се приключи пазаруването и да се изпрати поръчка към дадения магазин. Приложението съдържа стандартни настройки, като смяна на език и размер на буквите.

Програмният продукт за Андроид платформа се дели на пет основни части:

* Екран за вписване в програмата с име и парола. След попълване на полетата, информацията се изпраща към уеб частта , която проверява за такъв потребител и връща нужният резултат. На този панел се намира и бутон, навигиращ към интерактивна форма за регистрация. Също така настройките на приложението могат да се достъпят от бутон, намиращ се от лявата страна на хедъра.
* Екран с форма за регистрация. След попълване на всички полета, информацията се изпраща към уебуслугата, която от своя страна регистрира новия потребител.
* Основен екран. От него се достъпват функционалностите за сканиране,количката и полето за информация. При натискане на бутона „Scan“ устройството отваря своята камера, която влиза в режим за четене на кодове. След успешно сканиране даденият продукт автоматично се добавя в количката. Там може да се увеличава или намалява бройката му , а също така и да се премахне при нужда. След прилючване на пазаруването се изпраща заявка към уебуслугата и поръчката се записва в базата.
* Екран за следене на състоянието на поръчките на клиента. На него са визуализирани всички активни такива и информацията за тях.
* Екран за настройки на приложението. При необходимост за смяна на език или размер на шрифта.

Десктоп приложението е предназначено за търговеца или веригата магазини. То дава лесна възможност за добавяне на продукти в база данни,също така и генерирането на QR код за тях. Другата му функционалност е следеното и приемането на поръчки от клиенти.

Десктоп решението се състои от четири основни части:

* Екран за влизане в системата. След попълване на полетата съдържанието им се изпраща към уеб частта ,която проверява за такъв потребител и връща нужната информация.
* Екран за генериране на кодове. След въвеждане на необходимите данни във всички полета, на екрана се визуализира QR код. Това действие автоматично изпраща информацията към уеб модела, който записва продукт в системата с предоставените характеристики.
* Екран за поръчки. На него се показват всички поръчки. Там те могат да бъдат обработвани.
* Екран за общи настройки. На него потребителят може да зададе желаните от него настройки, с които да работи приложението.

От своя страна уеб услугата държи базата данни и служи за връзка между двете приложения и нея. В него са HTTP заявките, които могат да се ползват и от двете програми. Тези заявки извършват необходимите манипулации с базата или връщат нужната информация.

## **2.2. Съображения за избор на програмни средства и развойната среда**

Трите части на системата са ясно разделени и от страна на езиците за програмиране.

Тъй като Android приложението трябва да е лесно за използване от всеки потребител на различна възраст, е направено много просто от към front-end. Поради тази причина избрах да използвам един многофункционален език за създаване на Android приложения, а именно Basic4Android. Езикът е алтернатива на програмирането с Java и Android SDK. Платформата компилира до Java и след това отново го връща като Basic, използва смесен (Cross) компилатор.

Резултатът е изпълним код, предназначен за платформа, различна от тази, на която е извършена компилацията. Използва се например при създаване на код за [вградени системи](https://bg.wikipedia.org/wiki/%D0%92%D0%B3%D1%80%D0%B0%D0%B4%D0%B5%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) (embedded systems), както и в Basic4Android, където няма възможност или е по-трудно да се създават програми.

B4A включва визуален дизайнер, който опростява процеса на изграждане на потребителски интерфейси, насочени към телефони и таблети с различни размери на екрана. Компилираните програми могат да бъдат тествани в емулатори на AVD Manager или на истински Android устройства, използващи Android Debug Bridge и B4A Bridge.  
Самият език е подобен на Visual Basic и Visual Basic .NET, но е адаптиран към родната Android среда. Има разлики и прилики с Java, най-големия плюс е, че се пише много бързо и е много функционално, не е гъвкаво колкото Java, но по отношение на приложението, което е нужно за системата и ще се използва, е нужно просто бързо действие и лесно разгадаване на програмата.

За да може да работи приложението, освен компилатора, който се предоставя с B4A платформата, е нужно да се свърже към нея Java JDK, като възможност да се установи компилация, Android SDK- за инсталиране на платформите и инструментите за версиите на Andoid-а, които ще се използват. Също така най-новата версия на B4A е платена и е нужен лиценз.

По желание може да се използва B4A-Bridge, което компилира на устройството безконтактно чрез Wireless връзка, ако двете устройства са в една и съща мрежа. Може да се изтегли от Google Play безплатно.

Другият малко по-гъвкав, но непрактичен вариант, поради усложняване на работата за създаването на Android проложението, е писането на Java на Android Studio. Но използването на Android Studio ще създаде повече работа по програмирането на това не толкова сложно приложение.

От друга страна, уебуслугата и десктоп приложението са написани на C#,Visual Studio(IDE).

Microsoft Visual Studio е мощна [интегрирана среда за разработка](https://bg.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%B0%D0%BD%D0%B0_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D0%B7%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0) на софтуерни приложения за [Windows](https://bg.wikipedia.org/wiki/Windows) и за платформата [.NET Framework](https://bg.wikipedia.org/wiki/.NET_Framework). Използва се за разработка на конзолни и [графични потребителски интерфейс](https://bg.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D0%BD_%D0%BF%D0%BE%D1%82%D1%80%D0%B5%D0%B1%D0%B8%D1%82%D0%B5%D0%BB%D1%81%D0%BA%D0%B8_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) приложения, както и Windows Forms или WPF приложения, [уебсайтове](https://bg.wikipedia.org/wiki/%D0%A3%D0%B5%D0%B1%D1%81%D0%B0%D0%B9%D1%82), [уебприложения](https://bg.wikipedia.org/wiki/%D0%A3%D0%B5%D0%B1_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) и [уебуслуги](https://bg.wikipedia.org/wiki/%D0%A3%D0%B5%D0%B1_%D1%83%D1%81%D0%BB%D1%83%D0%B3%D0%B0) на всички поддържани платформи от Microsoft Windows, [Windows Mobile](https://bg.wikipedia.org/wiki/Windows_Mobile), Windows CE, .NET Framework, .NET Compact Framework и Microsoft Silverlight.

Visual Studio предоставя мощна интегрирана среда за писане на код, компилиране, изпълнение, дебъгване (както за високо ,така и за машинно ниво), тестване на приложения, дизайн на потребителски интерфейс (форми, диалози, уебстраници, визуални контроли и други), моделиране на данни, моделиране на класове, изпълнение на тестове, пакетиране на приложения и стотици други функции. Могат да се добавят и плъгини, които повишават функционалността на почти всяко ниво – включително добавянето на поддръжка за [source-control системи](https://bg.wikipedia.org/w/index.php?title=Source-control_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8&action=edit&redlink=1) (като [Subversion](https://bg.wikipedia.org/wiki/Subversion_(%D1%81%D0%BE%D1%84%D1%82%D1%83%D0%B5%D1%80)) и Visual SourceSafe), добавяне на нови инструменти като редактори и визуални дизайнери за [domain-specific languages](https://bg.wikipedia.org/w/index.php?title=Domain-specific_languages&action=edit&redlink=1) или инструменти за други аспекти (като например: [Team Foundation Server](https://bg.wikipedia.org/wiki/TFS), Team Explorer).

Visual Studio поддържа различни [езици за програмиране](https://bg.wikipedia.org/wiki/%D0%95%D0%B7%D0%B8%D1%86%D0%B8_%D0%B7%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B8%D1%80%D0%B0%D0%BD%D0%B5) (например: [C#](https://bg.wikipedia.org/wiki/C_Sharp), VB.NET, [C](https://bg.wikipedia.org/wiki/C_(%D0%B5%D0%B7%D0%B8%D0%BA_%D0%B7%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B8%D1%80%D0%B0%D0%BD%D0%B5))/[C++](https://bg.wikipedia.org/wiki/C%2B%2B), [F#](https://bg.wikipedia.org/w/index.php?title=F_Sharp&action=edit&redlink=1), [XML](https://bg.wikipedia.org/wiki/XML)/[XSLT](https://bg.wikipedia.org/wiki/XSLT), [HTML](https://bg.wikipedia.org/wiki/HTML)/[XHTML](https://bg.wikipedia.org/wiki/XHTML), [JavaScript](https://bg.wikipedia.org/wiki/JavaScript), [CSS](https://bg.wikipedia.org/wiki/CSS) и други) и различни технологии за разработка на софтуер (Win32, COM, [ASP.NET](https://bg.wikipedia.org/wiki/ASP.NET), ADO.NET Entity Framework, Windows Forms, WPF, Silverlight и още десетки други Windows и .NET технологии). Съществуват и отделни езикови версии на Visual Studio, които обаче предоставят по-ограничени услуги за потребителя: [Microsoft Visual Basic](https://bg.wikipedia.org/wiki/Microsoft_Visual_Basic), Visual J#, Visual C#, and Visual C++.

## **2.3. Принцип на софтуерен модел**

След стартиране на Android приложението се отваря екран за идентификация на потребителя. Въвеждат се име и парола и информацията се изпраща към уебуслугата за валидация. След успешен вход в системата се преминава в следващия екран на програмата , където хората имат възможността да сканират кодове, да управляват количката си и да следят своите поръчки.

Всяка операция изпраща заявка и очаква отговор, затова още при стартиране на приложението се проверява връзката на устройството с интернет.

'Ping Google DNS - if you can't reach this you dont have internet on the device!

Sub **Connected** As Boolean

Dim p As Phone

Dim Response, Error As StringBuilder

Response.Initialize

Error.Initialize

p.Shell("ping -c 1 8.8.8.8",Null,Response,Error)

Log("======= Response ========")

Log(Response)

Log("======= Error ===========")

Log(Error)

Log("======================")

If Error.ToString="" Then

Return True

Else

Return False

End If

End Sub

Tози метод се ползва като параметър при инициализацията на HTTP модула.

'Initializes the object. You can add parameters to this method if needed.

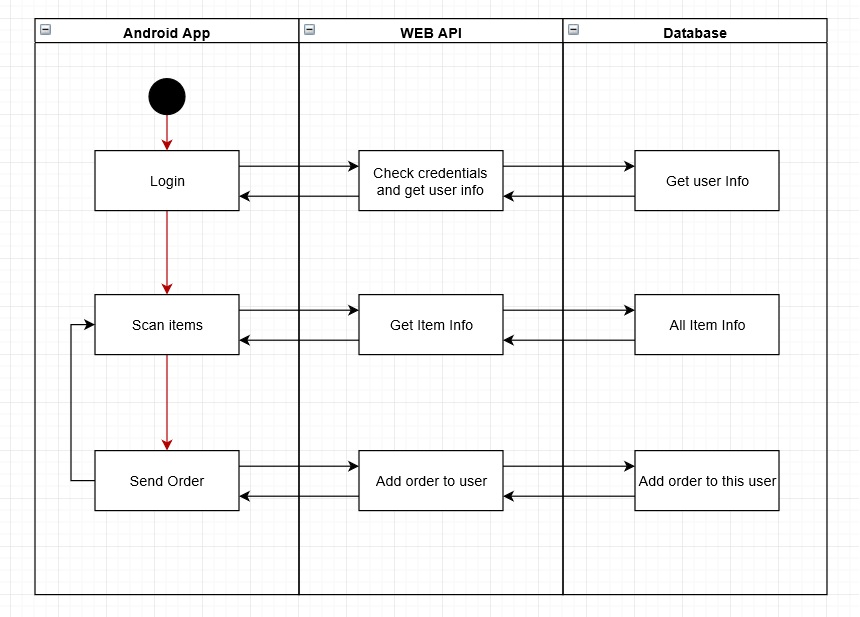
Public Sub **Initialize**(Connection As Boolean)

IsConnected = Connection

End Sub

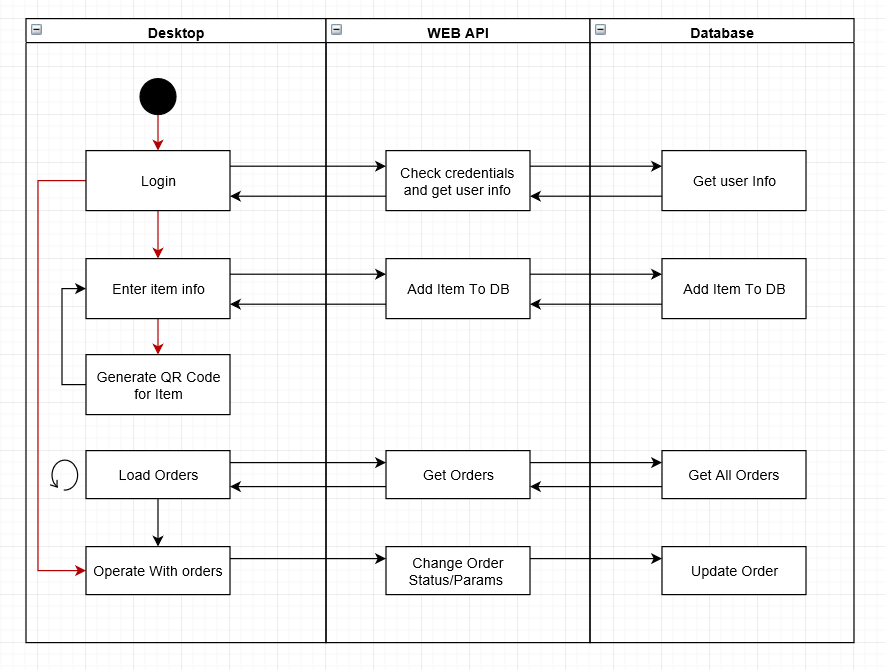
Така при неуспешна връзка с интернет не може да се ползва нито една заявка и потребителят не може да използва приложението.

След като влезе, той може да сканира желаните от него продукти, за всеки продукт се взима нужната информация от базата, а при изпращане на поръчка,тя се финализира и изпраща отново към базата.

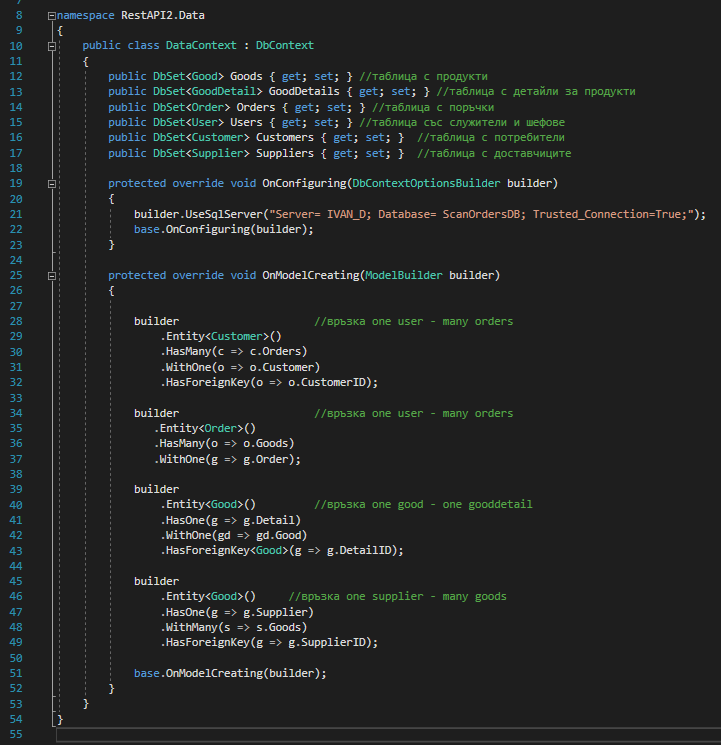


Настолното приложение работи на същия принцип, с разликата ,че то е предназначено за персонал и има различна функционалност. Създава и добавя продукти в базата, като в същото време им създава QR код. Също така получава и обработва поръчки , дошли от клиенти чрез мобилното приложение.

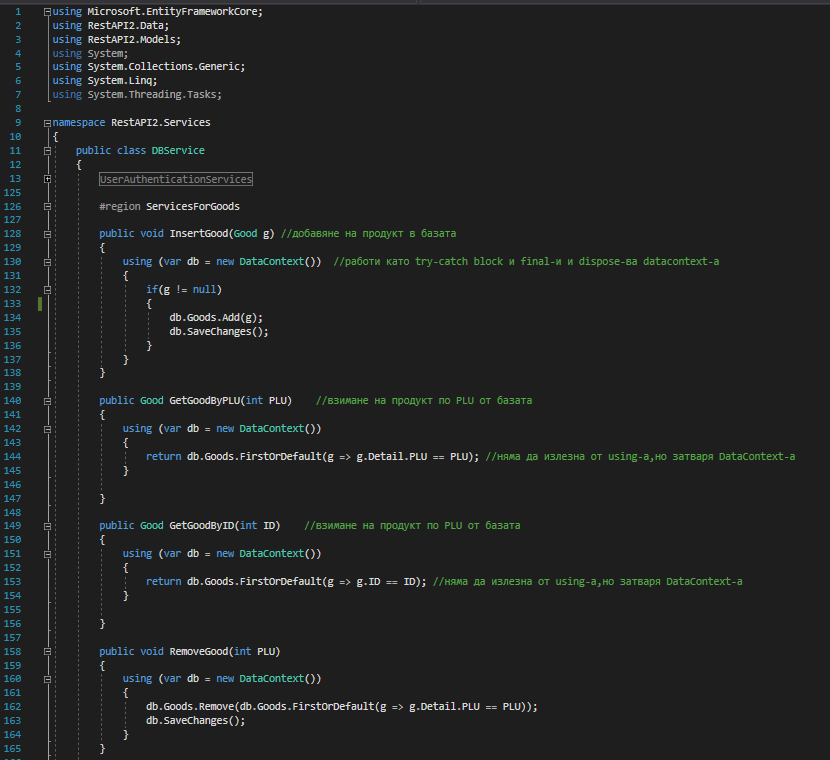
Потребителите за настолното приложение и това за андроид са различни и не могат да се вписват в чуждите платформи поради различното предназначение на двете програми.



Уебуслугата е предназначена за комуникация и различни операции с базата. Двете приложения комуникират чрез заявки с нея, а тя изпълнява нужните операции с базата данни. Самата база е писана с подход Code First , на Entity Framework Core. Това ще рече,че моделите на приложението са създадени първи и спрямо тях Visual Studio е направило базата и необходимите връзки. Обработката на заявките се извършва в класа Controller, a връзките с базата са описани в класа DataContext.



Всички операции, свързани с базата, са записани под формата на методи в класа DBServices. След всяка манипулация на информацията, записана в базата, се извиква методът SaveChanges() , чрез който се запазва сегашното състояние на базата.



## **2.6. Преглед на библютеки, използвани за приложенията**

* Андроид

JSON

* Taзи библиотека помага с сериализацията и десериализацията на JSON обекти и създаването на JSON string.

B4aZXing

* трансформира камерата на устройството в скенер и предоставя съответните методи за работа със сканирани данни.

ОkHttpUtils2

* Тази библиотека създава и изпраща необходимите HTTP заявки

Runtime Permissions

* Ако версията на андоид SDK е по-малка от 23, тогава стандартната система за позволения ще бъде използвана на всички устройства- включително Android 6+, но скоро всички Google Play апликации ще трябва да използват версия на андоид SDK 26+. Чрез runtime permissions потребителят не е питан за позволения, когато инсталира приложение от Google Play.Вместо това искането за позволение е по време на изпълнение.

Camera

* Дава достъп до камерата на дадено усройство и съдържа проверки за намирането на налична такава.

Reflection

* Тази библиотека съдържа обект Reflection, който позволява достъп до методи и полета от обектите на Android, които не са изложени на езика Basic4Android. Това прави чрез съоръжение, наречено "Reflection", което използва метаданни за обекти, които са включени в пакета за приложения и позволява динамичен достъп до полета и методите по време на изпълнение.
* Настолно приложение и уебуслуга

Мicrosoft EntityFrameworkCore

* Entity Framework (EF) Core e лека cross-platform версия на популярната Entity Framework data access technology. EF Core може да служи като object-relational mapper (O/RM), позволяващо на .NET разработчиците да работят с база данни, използвайки .NET обекти и елиминирайки нуждата на по-голямата част the data-access кода , който по принцип те трябва да напишат.

Microsoft AspNetCore Mvc

* ASP.NET Core MVC е framework за разработката на web приложения и APIs, използвайки Model-View-Controller модел на разработка, който разделя приложението на три главни групи от компоненти : Models, Views, и Controllers. По този начин потребителските заявки са насочени към Controller, който е отговорен за работата с Model, за да изпълни потребителски действия или да вземе резултат от queries.

## **2.7. Получени резултати**

Полученият резултат е едно интерактивно и нестандартно за пазара приложение, което улеснява търговията. Системата дава свобода на рекламна изява от страна на продавача, тъй като той може да разпространява, където поиска своите QR кодове – брошури,списания,билборди и други реклами. Потребителят може по всяко едно време да сканира и поръча конкретните продукти.

a. Мнението на хората

Приложението е все още в тестов период и не е достъпно публично.

b. Плюсове:

- интересен и различен начин на работа;

- улеснен интерфейс;

- поръчването на продукти става по-лесно;

- оползотворява използването на брошури и хартиени рекламни материали.

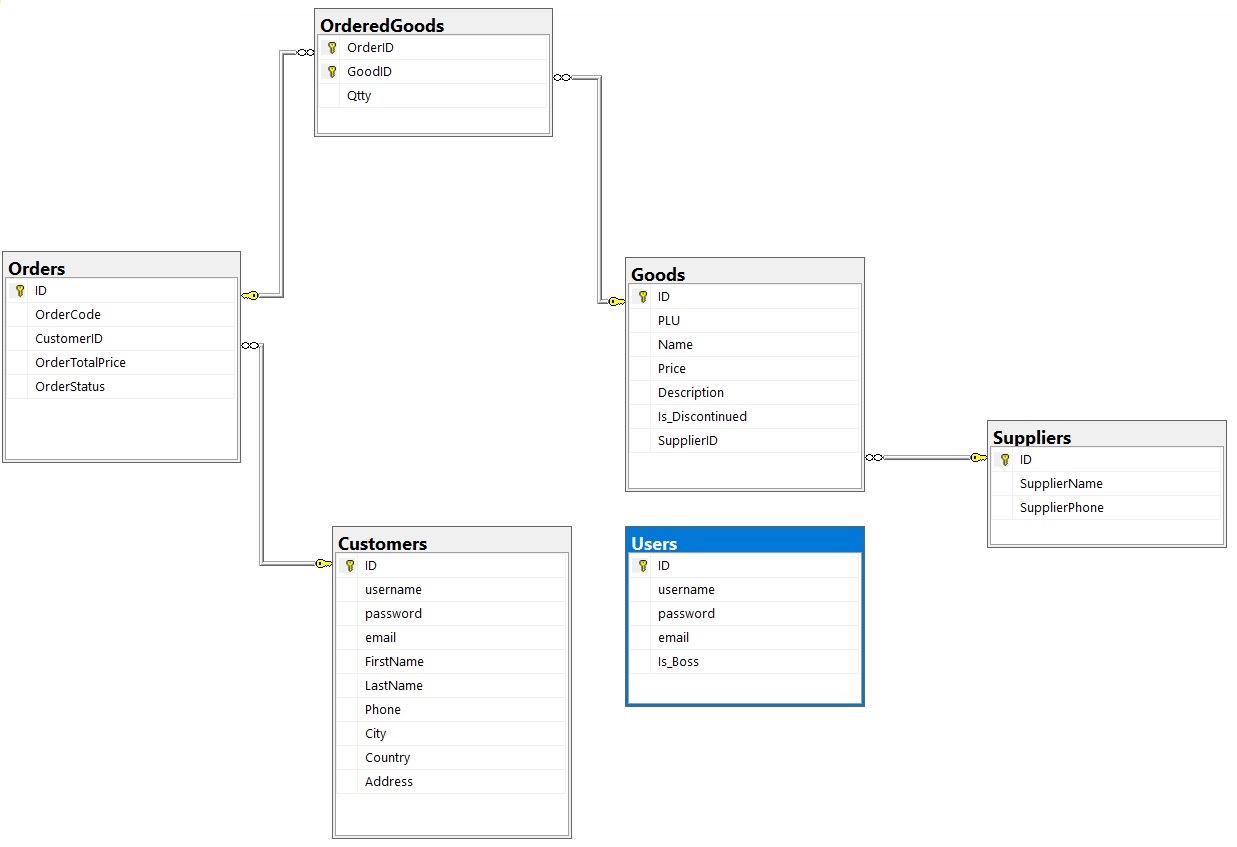
c. Минуси

За сега минуси не са установени, тъй като не е пуснато в употреба в реална среда.

## **2.8. Проектиране на структурата на базата от данни**

Базата данни се състои от няколко таблици – User,Customer,Order,Good,Supplier.

Tаблицата User съхранява информацията за потербителите на конкретния магазин.Тя се използва за тези от тях, които се вписват в настолното приложение. В таблицата Customers се съхраняват регистрациите на потребителите на андроид приложението. Всеки Customer има много Order(Поръчки). Връзката е one-to-many.Всяка поръчка съдържа информация за сеге си и списък от продукти(Good).Tук връзката е пак one-to-many. Eдна поръчка може да има много продукти. От своя страна всеки Good има по един GoodDetail и един Supplier. В GoodDetail се съхранява информацията за даден продукт, а в Supplier – информацията за доставчика. Всеки Supplier обаче има много Goods,които той доставя.



Таблицата User е независима и не е обвързана с останалите таблици.

# ТРЕТА ГЛАВА

# ПРОГРАМНА РЕАЛИЗАЦИЯ

Реализацията на продукта започва с направата на уебуслугата, която е най-основната част от системата. Приложението се дели на четири основни части:

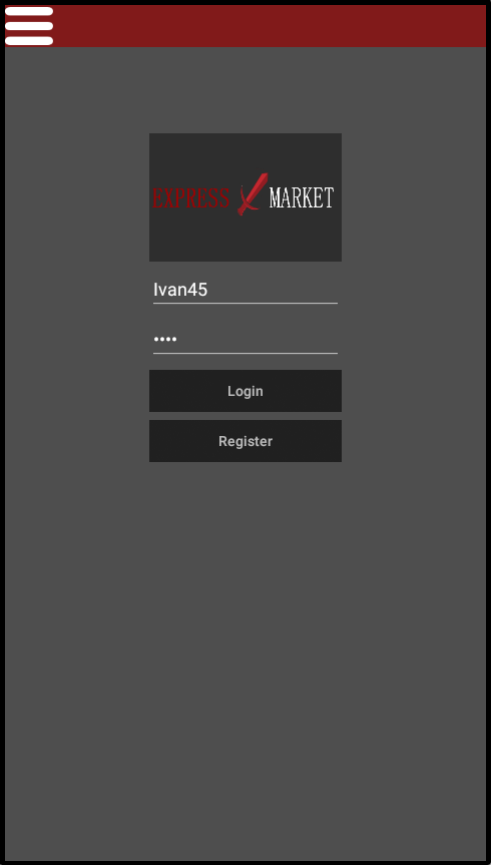
* Модели на обектите,които ще съхраняваме в базата данни.
* DataContext клас, в който създаваме таблиците, връзките между тях и връзката със сървър.
* DBServices клас, в който са описани различните методи, свързани с работа с базата.
* Controller клас, който съдържа всички HTTP заявки, нужни за работа с приложенията.

При стартиране на Уебуслугата се създават таблиците на базата данни,описани в DataContext класа, и връзките между тях. Самата база данни служи за запазване на иформацията на нови продукти,поръчки и потребители.

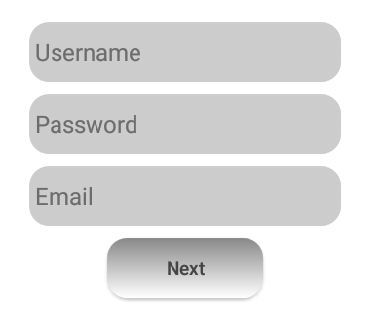
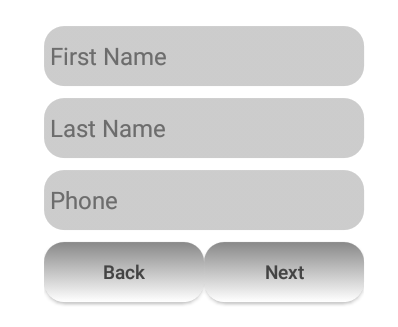
Системата приема HTTP заявки от останалите приложения, които се обработват в Controller класа. Te може да са POST,GET,PUT,DELETE и др.. В зависимост от подадената заявка се извикват методи от DBServices класа, които са отговорни за директна работа с базата данни. Всичко, свързано със SELECT,INSERT,UPDATE и т.н., е в този клас.

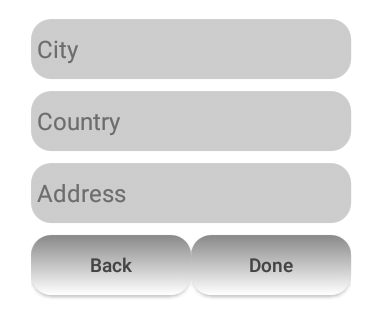
От своя страна Android приложението се дели на пет основни части:

* Панел, върху който се построява екран за влизане в системата с име и парола, след попълване на полетата, информацията се изпраща чрез заявка към уебчастта за валидация.

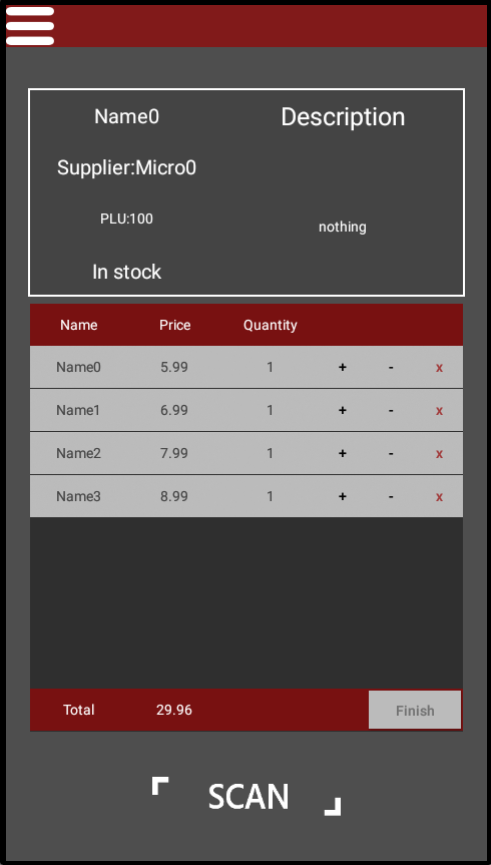


* Екран за регистрация. При натискане на бутона за регистрация се построява друг екран. На него потребителят следва необходимите стъпки и се регистрира.След натискане на бутона “Done” се изпраща заявка,съдържаща необходимата информация за регистрация.

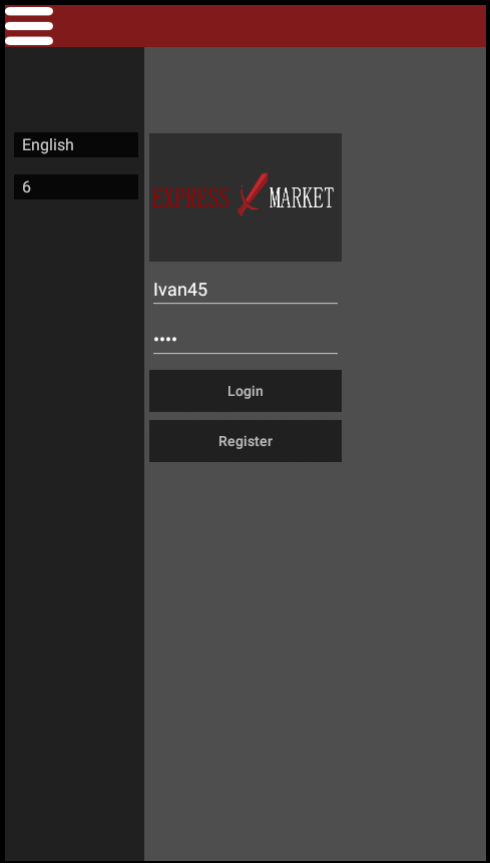


* Главен екран на приложението. Визуализира се след успешен вход в системата. На него се намират бутонът за сканиране,количката с продукти и бутон за настройки.



При селектиране на продукт от количката, допълнителна информация за него се изписва на екрана. Всеки продукт в количката има бутони за манипулация на количеството му и бутон за премахване. На количката е бутонът за приключване на поръчки,който при натискане отвежда потребителя на друг екран,където да финалицира своята поръчка.

Бутонът за настройките е един и същ в екрана за вход и в главния екран.Различното обаче е , че преди потребител да се е вписал в системата, от този бутон се визуализират общите настройки на приложението като език и размер на шрифта. След успешно влизане в системата функционалността на бутонът се променя и започва да служи за преминаване в различните екрани.Тази функционалност не е налична в началния екран на приложението по обясними причини.



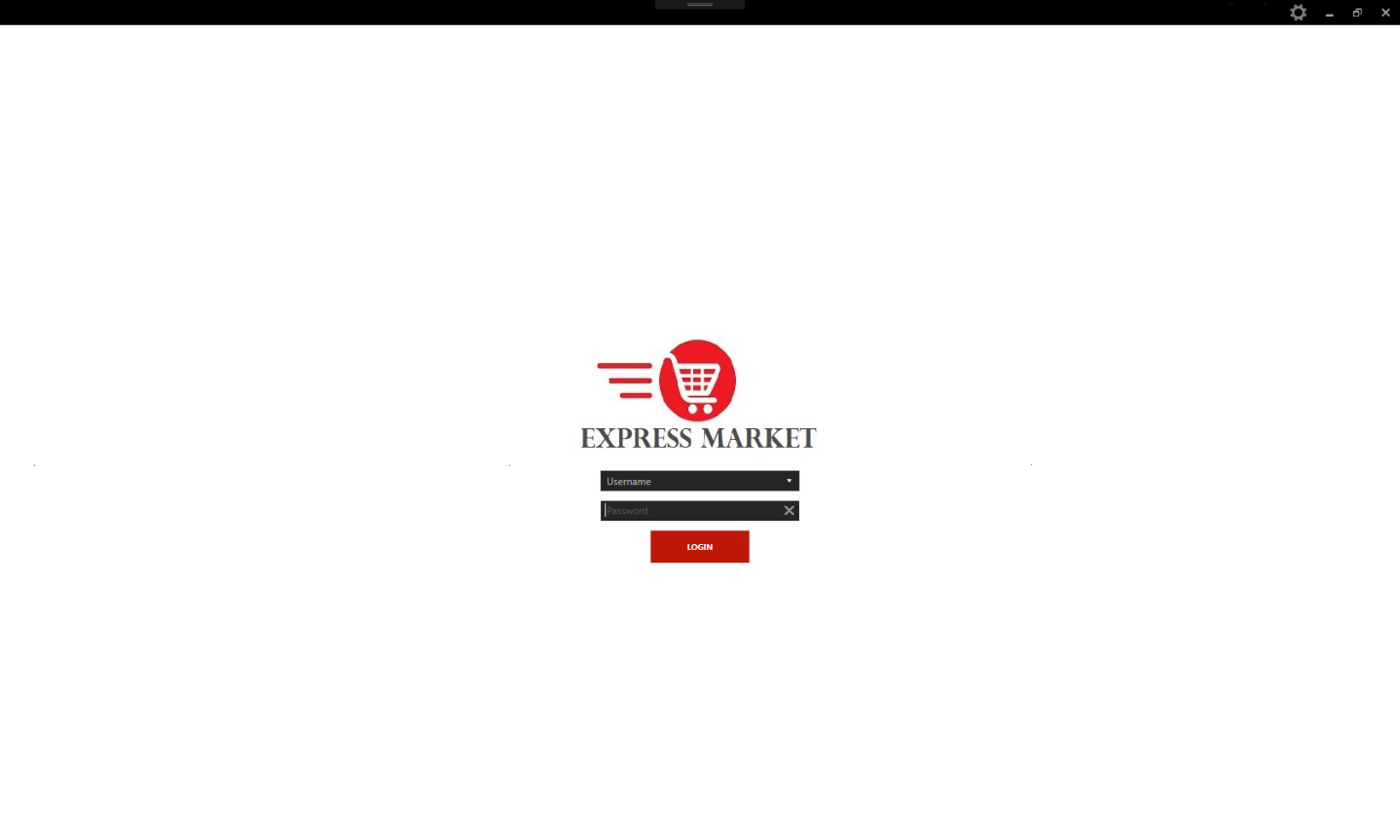
* Екранът,в който се визуализират поръчките, прилича на главния екран. В него се построява Scroll View , което съдържа поръчките на клиента. При селектиране на дадена поръчка на екрана се показва повече информация за нея.



След натискане на бутона за връщане назад на устройството или след смяната на екран от бутона за настройки,въведените данни се запазват.Тези, които са за приложението, се запазват локално,а тези, свързани с локацията, се изпращат със заявка.

Настолното приложение е съставено от три части,подобни на мобилния софтуер,но с различно практическо предназначение.

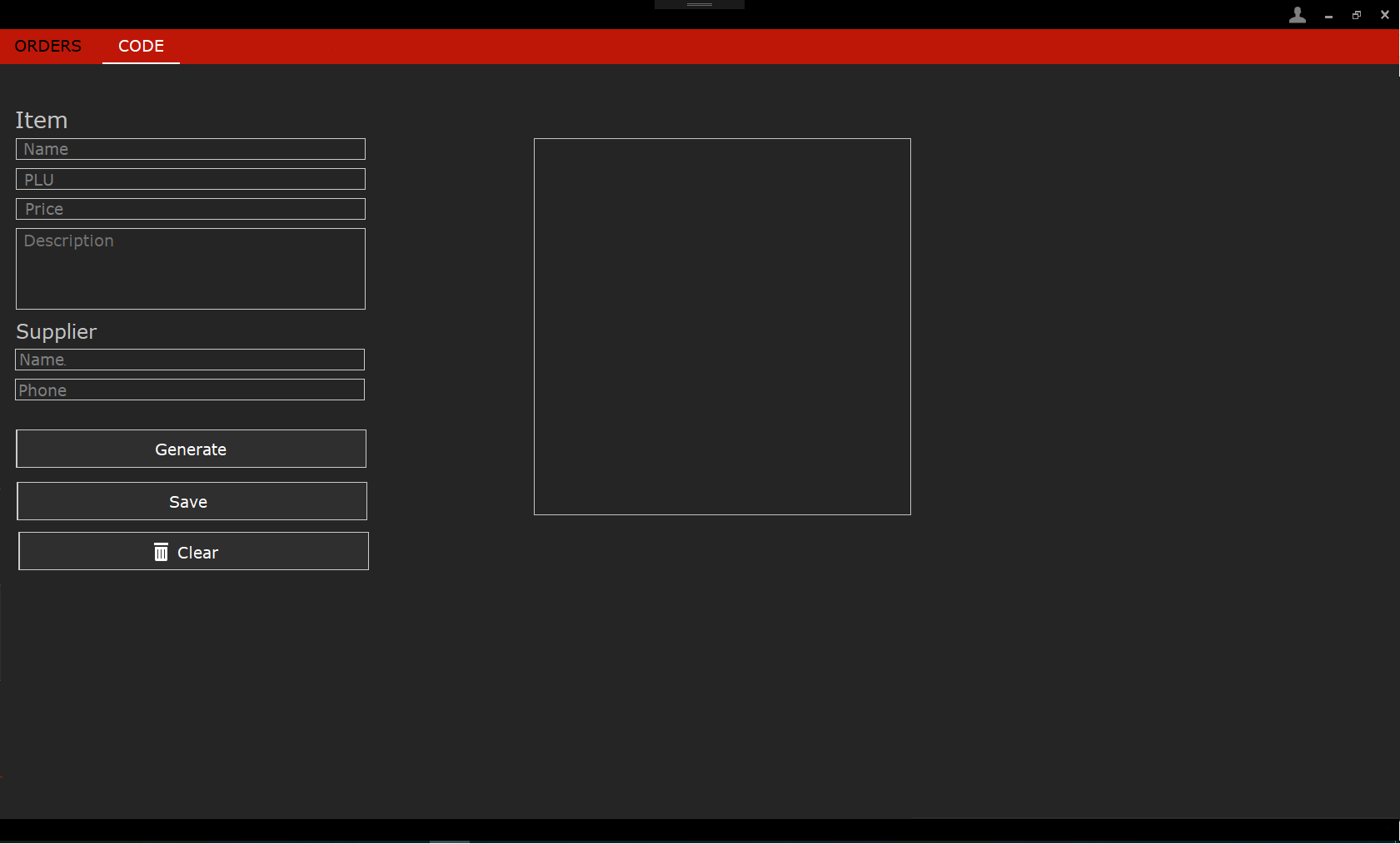
* Началният екран се състои от форма за въвеждане на име и парола,както и бутон за общи настройки на приложението. Не съдържа бутон за регистрация,тъй като това е служебен софтуер и се раздават готови потребители на служителите. След въвеждане на правилна информация се изпраща заявка към уебуслугата.



След успешно влизане служителите попадат в основния прозорец на програмата,където имат възможност да сменят между два изгледа. На единия могат да видят всички налични поръчки и да ги обработят, а на другия да въведат необходима информация за генерирането на QR код.



* В изгледа за поръчки новите такива се визуализират от лявата страна и с тях е представена основна информация за клиента. При селектиране на поръчка в центъра на екрана се визуализира нейното съдържание.



* В прозореца за създаване на QR код потребителите могат да въвеждат данни в полетата и след натискане на бутона „Generate” се визуализира код с нужната в него информация. При нужда могат да изчистят всяко поле с бутона „Clear” . След натискане на бутона „Save” , генерираният QR код се запазва в дадена папка. При всяко генериране се изпраща заявка към уебуслугата и се добавя нов продукт към базата със съответните данни.

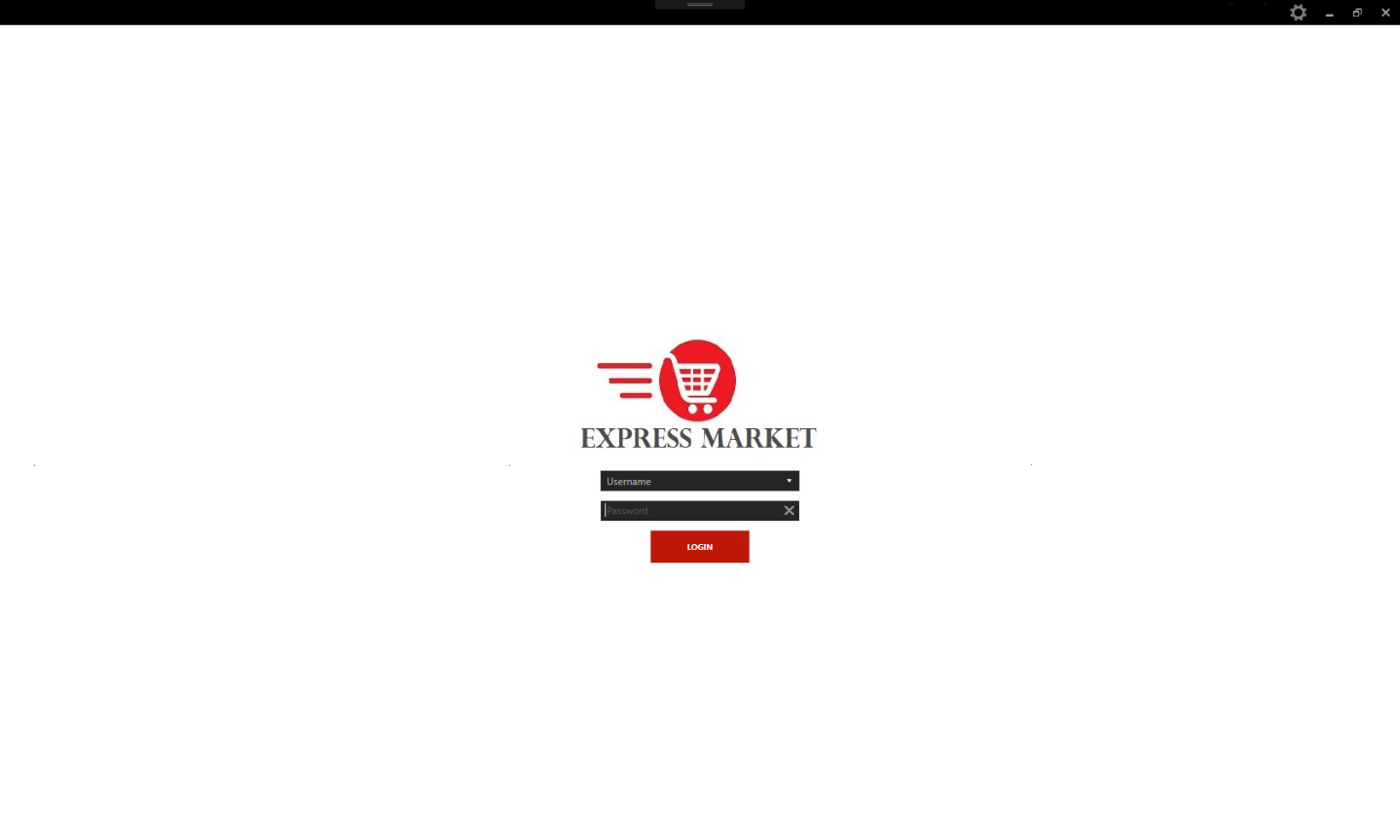
# ЧЕТВЪРТА ГЛАВА

# РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ

Системата ‚‚Търговски софтуер за създаване и обработка на поръчки'' ще се ползва от даден магазин и неговите клиенти. Състои се от две части, предназначени за директна работа – една за клиентите на магазина и една за работещите в него и една сървърна част.

За да ползват клиентите предоставения софтуер, те трябва да инсталират даденото приложение на мобилните си устройства и да се регистрират. От друга страна, на служителите на даден магазин се предоставят готови регистрации за работа с настолното приложение. Сървърът се инсталира и се стартира върху лаптоп или компютър.

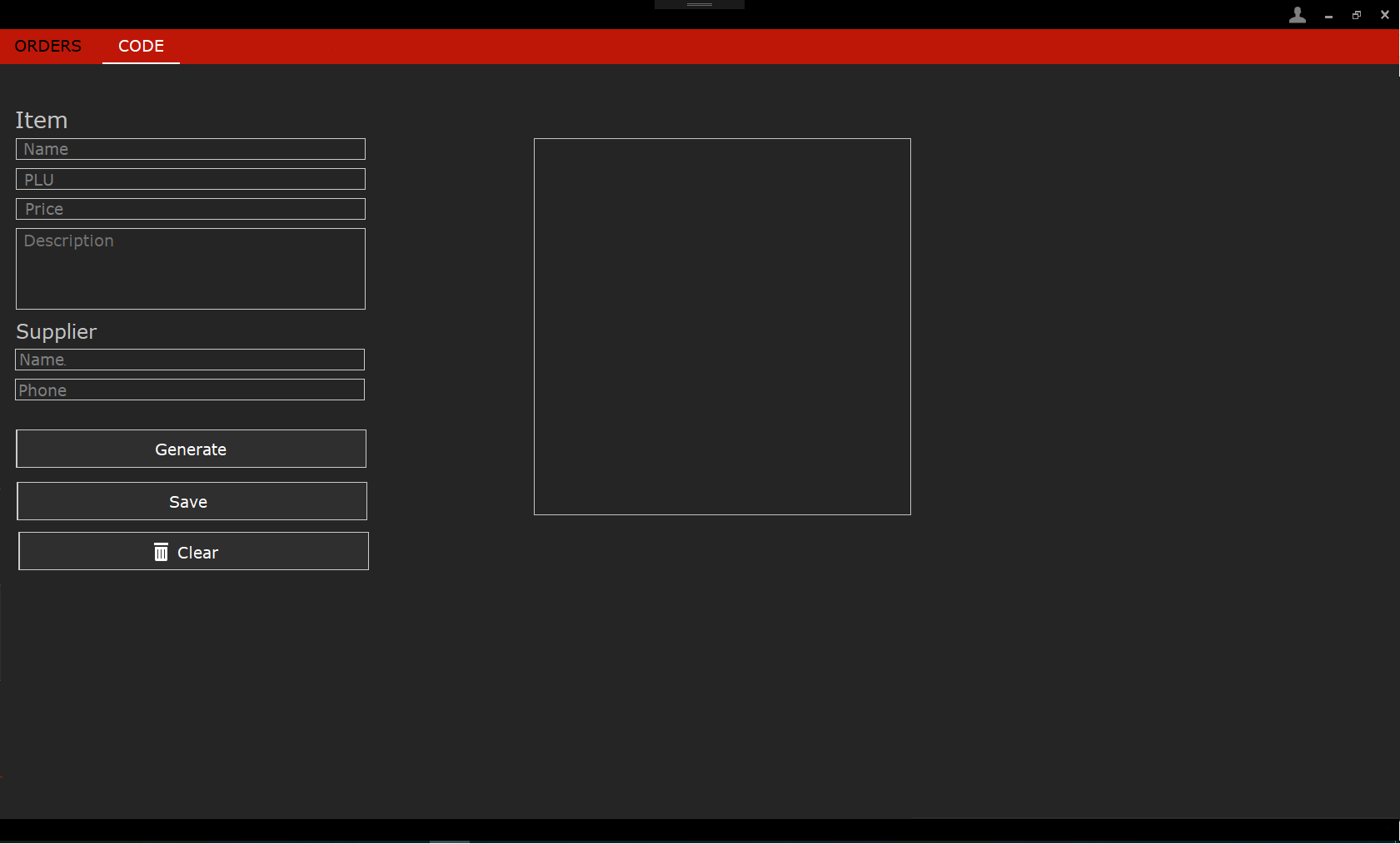
След като е инсталирано и стартирано настолното приложение, служителите трябва да въведат своето потребителско име и парола.



При правилно въведена информация се показва базовият екран на приложението.

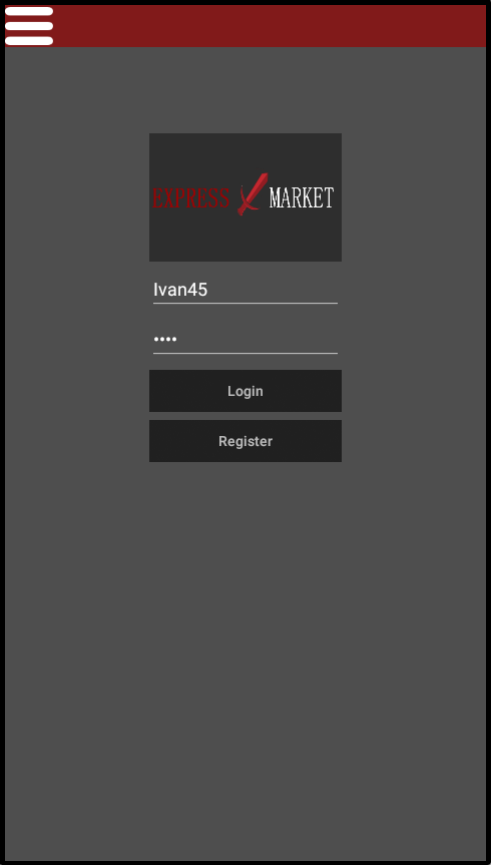


Тук се визуализират всички налични поръчки. Представена и информация за всяка поръчка. Работниците могат да селектират дадена поръчка и да я потврърдят или откажат при необходимост. Също така могат да сменят изгледите и да отидат в екрана за създаване на QR код.

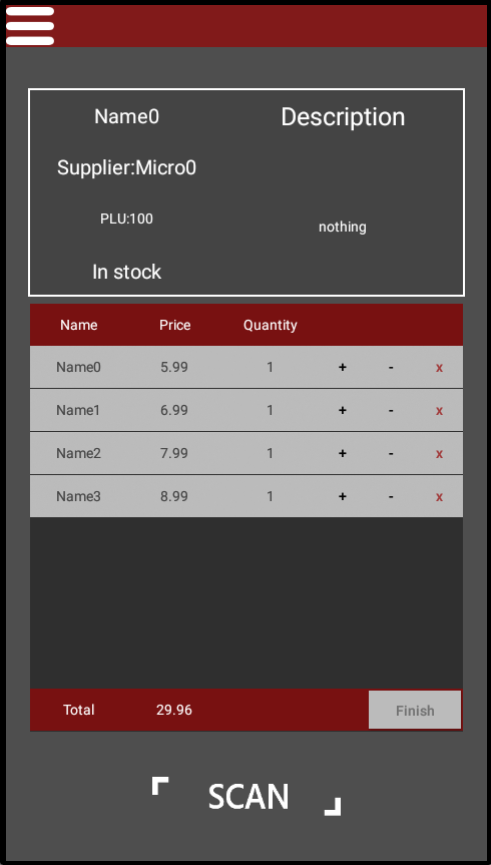


Тук те могат да въведат нужната информация за даден продукт и да създадат QR код за него. Този код е необходим за работата на мобилното приложение, предназначено за клиентите. Всеки магазин има пълната свобода да разпространява, където желае своите кодове.

От своя страна след инсталация и стартиране на мобилния софтуер, клиентите също трябва да въведат своите потребителски данни или да се регистрират при липсата на такива.



След успешен вход в системата се визуализира началният екран на приложението.



Чрез натискане на бутона за сканиране се отваря камерата на устройството, която се преобразува на скенер. Клиентите могат да сканират предоставените QR кодове и по този начин добавят дадените продукти в своята количка. В нея те могат да следят сканираните от тях продукти и информацията за тях. По всяко едно време могат да приключат и изпратят своята поръчка към магазина.

След изпратена поръчка може да се следи нейния статус на екрана за поръчки в приложението.



На този екран те могат да селектират своите поръчки и да виждат конкретна информация за тях,докато следят статуса им

# 

# Заключение

В заключение можем да кажем, че ‚‚Търговски софтуер за създаване и обработка на поръчки'' е един по-интересен начин на електронно пазаруване , предоставящ бързина и удобство. За момента е само тест версия, но развитието на продукта продължава и все повече се подобрява.

Разработката продължава. Също така постоянно се измислят нови допълнения към двете платформи , целящи по-добра работа със софтуера както от страна на служителите на даден магазин, така и за техните клиенти.

# Източници

<https://en.wikipedia.org/wiki/Basic4android>

<https://www.b4x.com/>

<https://en.wikipedia.org/wiki/Microsoft_Visual_Studio>

Съдържание

[ПЪРВА ГЛАВА 4](#_Toc255525)

[Обзор за съществуващите решения 4](#_Toc255526)

[**1.1 Подобни съществуващи решения** 4](#_Toc255527)

[**1.2 Развойни средства и среди. Проучване.** 6](#_Toc255528)

[ВТОРА ГЛАВА 8](#_Toc255529)

[ПРОЕКТИРАНЕ НА СТРУКТУРАТА 8](#_Toc255530)

[**2.1. Функционални изисквания** 8](#_Toc255531)

[**2.2. Съображения за избор на програмни средства и развойната среда** 12](#_Toc255532)

[**2.3. Принцип на софтуерен модел** 15](#_Toc255533)

[**2.6. Преглед на библютеки, използвани за приложенията** 21](#_Toc255534)

[**2.7. Получени резултати** 24](#_Toc255535)

[**2.8. Проектиране на структурата на базата от данни** 25](#_Toc255536)

[ТРЕТА ГЛАВА 27](#_Toc255537)

[ПРОГРАМНА РЕАЛИЗАЦИЯ 27](#_Toc255538)

[ЧЕТВЪРТА ГЛАВА 39](#_Toc255539)

[РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ 39](#_Toc255540)

[Заключение 47](#_Toc255541)

[Източници 47](#_Toc255542)