LuncherBox - Интерактивно Меню

Симо Георгиев Александров и Любо Ивайлов Любчев

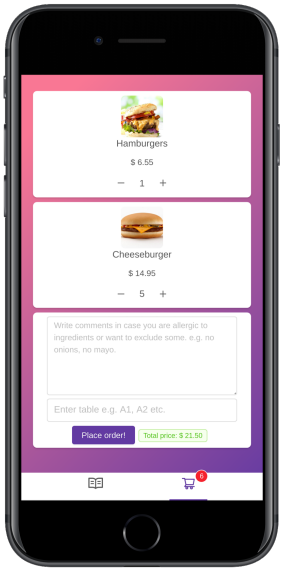
Февруари 2019

Абстракт

Luncher Box - Интерактивно Меню е уеб приложение, което цели поръчването на храна от клиенти в ресторанти да става по-бързо. Освен доволни клиенти, собствениците на заведението ще спестяват пари, чрез свеждане на нуждата от сервитьори до минимум.

Проектът е с приложен характер, все още е в процес на разработка и е от сферата по информатика и информационни технологии. Идеята е измислена от Любо Любчев, а е реализирана от двамата автори.

Клиентите на ресторанта ще осъществяват поръчките си, чрез "Luncher Box", който ще изпрати заявката към кухнята, където тя ще бъде приета и обработена отново през приложението.



Abstract

Luncher Box - Interactive Menu is a web app which aims making placing orders by clients in restaurants a much faster task.

Other than the satisfied customers, the restaurant owners will save money by lowering the required amount of waiters to a minimum.

The project has applicational nature, it is still under development and belongs to the IT field. The idea was conceived by Lyubo Lyubchev and was realised by both of the authors.

The clients of the restaurant will place orders through "Luncher Box", which will send the request to the kitchen, where it will be processed through the app.

Съдържание

Увод

Много често клиентите на ресторантите чакат прекалено дълго само за да направят поръчка - изискват се доста сервитьори, за да обслужат всички, от това следва спад на общата ефективност на заведението.

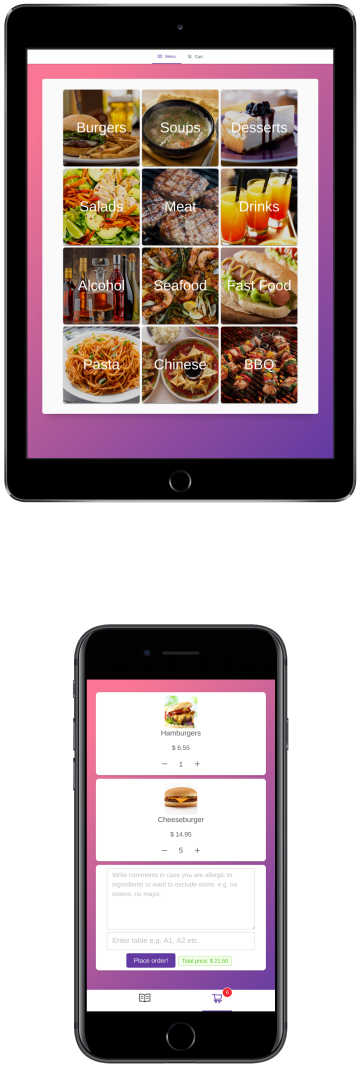
С тази разработка целим намаляването или премахването на времето, нужно за чакане преди да се приемат и изпълняват поръчки в заведения. Това ще допринесе до по-висока ефективност на ресторантите, тъй като се намаля нужния брой сервитьори, а така ще се намалят и разходите на ресторанта. Целим се в създаването на решение, което не изисква покупката на отделно устройство за всяка маса.

За да постигнем горепосочените цели, сме си поставили малко, но за нас предизвикателни задачи, а именно следните:

- Планиране на архитектура и подбор на правилните технологии;
- Разработка на backend сървър;
- Оформяне на красив и интуитивен графичен интерфейс;
- Създаване на цялостно функциониращо уеб приложение, което ще работи само в локалната мрежа на ресторанта;
- Рекламиране на продукта;
- Разпространяване на продукта.

В следващите глави ще задълбочим как приложението ще работи и как ще решим тези задачи.

Галерия



Фигура 1: Преглед на различни устройства

Функции

Приложението предоставя следните функции:

- Лесен, красив, бърз и интуитивен интерфейс за всички резолюции;
- Административен панел;
- Потребителски панел;
- Директен достъп до менюто на ресторанта;
- Масово импорт-ване и експорт-ване на продукти в различни формати като - JSON, YAML, CSV или XML;*
- Правене и отказ на поръчка към кухнята без нужда от регистрация;
- Статус на поръчката;
- Плащане чрез кредитна/дебитна карта, в брой или криптовалута;*
- Автоматично запазване на продуктите в количката.

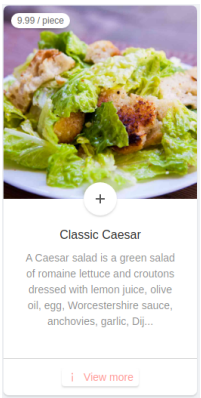
*в процес на разработка

Как работи?

На всяко заведение ще му бъде предоставено устройство (например Raspberry Pi), на което приложението ще се хоства на локалната мрежа. По този начин, приложението ще може да се използва само в обсега на ресторанта. При поставянето на поръчка, клиентът ще бъде помолен да извърши плащане или чрез кредитна/дебитна карта, в брой или с криптовалута, предотвратявайки експлоатация от външни клиенти.

Потребителски панел

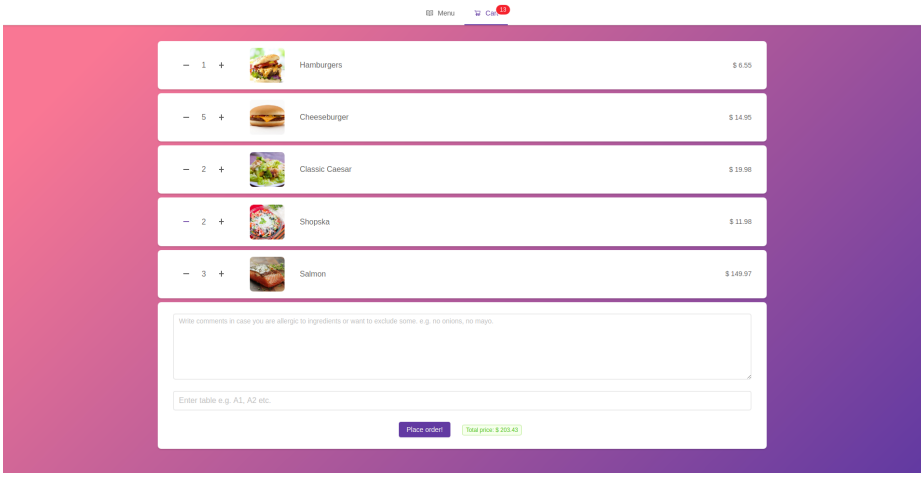
Използването на приложението ще бъде лесна задача. Клиентът ще трябва да се свърже за интернет мрежата на ресторанта, използвайки собственото си мобилно устройство. След осъществената връзка, потребителят ще достъпи менюто през browser-а си. Вследствие на това клиентът може да избере продукти, които може да добави чрез “+” бутона. (фиг. 2)



Фигура 2: Продукт

След добавянето, продуктът се добавя в количка,
която държи предишно добавените продукти.

Когато клиентът счита, че е готов за поръчка, той може да я осъществи в таб-а на количката. (фиг. 3)

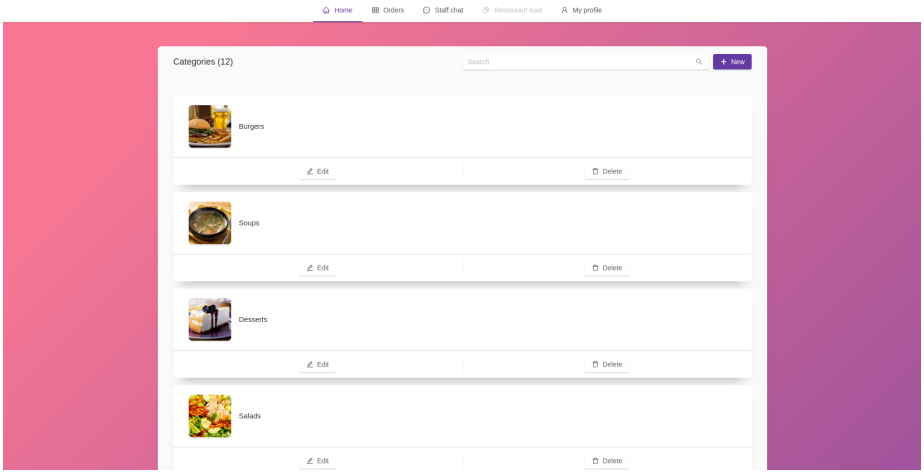


Фигура 3: Количка

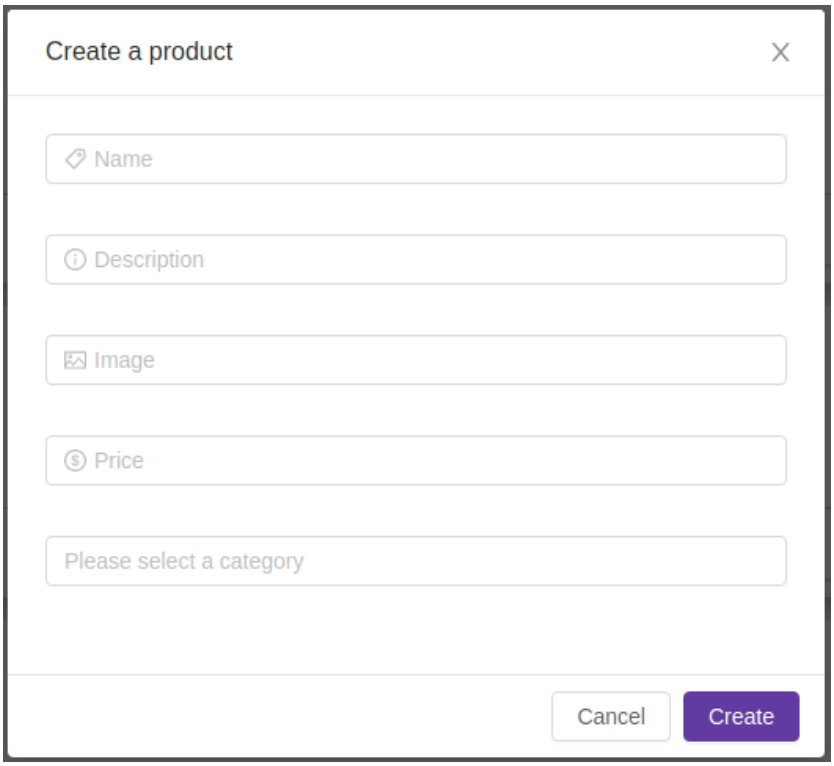
Тук потребителя може да завърши неговата поръчка с натискането на зеления бутон. Вследствие на това, клиента трябва да избере метод на плащане и да заплати.

Административен панел

При успешно изпратена поръчка от клиента, тя ще се появи в административния панел. Там готвачите мигновено ще виждат новите поръчки и ще могат да уведомят клиентите кога ще е готова директно през приложението. Всеки готвач и сервитьор ще има собствени потребителски профили, чрез които ще влизат в административният панел. (фиг. 4)
Друга функция е лесното добавяне, обновяване и премахване на продукти и категории. (фиг. 5)



Фигура 4: Административен панел



Фигура 5: Добавяне на продукт

Технологии

Като език на програмиране използвахме TypeScript, тъй като, може да се използва за frontend и за backend, е cross-platform, е statically typed и синтаксиса не е сложен за разбиране.

Backend

node.js

Избрахме node.js защото е висококоростен, високопроизводителен, щадящ ресурси, ефикасно комуникиращ в реално време, гъвкав и лесен за разработване environment на TypeScript / JavaScript.

express.js

За HTTP сървър подходихме с express.js, тъй като тя ни предоставя абстракцията за създаване REST API, вместо да задълбаваме подробно в прости неща като CRUD операциите.

passport.js

Passport.js избрахме за създаване на профили както и за управлението на автентикация.

nodemailer

Nodemailer бе използван за изпращане на e-mail-и за потвърждаване на потребители.

TypeORM и MySQL

Решихме да имплементираме TypeORM за ORM, а MySQL за база от данни, тъй като е скалираща база

от данни и ще е устойчива дори и в най-натоварените ресторанти.

ioredis и Redis

Ползвахме ioredis клиента за redis за session store, кеширане на MySQL query-та и държене на поръчки.

socket.io

За realtime връзката между backend и frontend използвахме socket.io, поради леснотата му на използване.

Frontend

react.js

За да създадем хубав и гъвкав интерфейс, заедно с reusable компоненти използвахме react.js - библиотека, създадена от Facebook.

next.js

За да създадем SPA и PWA използвахме next.js - библиотека, която предоставя SSR.

node.js

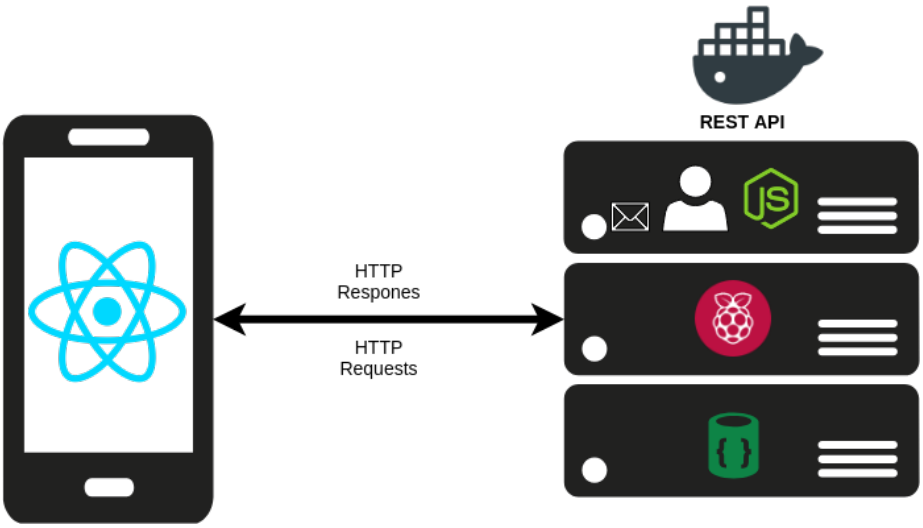
Тъй като библиотеката react.js е зависима от node.js, се наложи да го използваме и за frontend-а.

socket.io

За приемането и изпращане на поръчки използвахме socket.io.

Архитектура

Визуализирайки архитектурата на “Luncher Box”, тя ще изглежда по следния начин (фиг. 6):



Фигура 6: Архитектура

Етапи на развитие

Избор на тема

Двама от авторите сме забелязвали дългото чакане по заведенията само за да бъде взета поръчката. В продължение на 30 мин. размишляване по време на чакането в едно от заведенията, започнахме да се чудим как да оптимизираме целия процес със знанията, които имаме. Хрумна ни, какво можем да направим по въпроса - да създадем интерактивно меню. Чрез него клиентите ще могат директно да изпращат поръчки към кухнята или бара на заведението.

Проучване

Бяха проучени няколко подобни решения, но се оказа, че имат следните изисквания:

- Регистрация за правене на поръчки;
- Отделно хардуерно устройство на всяка маса.

При изискването на отделно хардуерно устройство за всеки клиент или маса, се изисква висок начален капитал.
Убедихме се, че има неща, които могат да се подобрят в другите решения, спестявайки доста ресурси и време както на клиентите - липсата от нужда за регистрация, а на собствениците нуждата от купуването на устройства.

Избиране на технологии и архитектура

През този етап ни минаха доста идеи относно подхода ни с технологиите, като се спряхме на вече гореспоменатите. (Виж сек. 5 - Технологии)

Изработване

По време на различни хакатони, работихме върху приложението.
Създадохме прост REST API, чрез който извършвахме прости CRUD операции.
Създадохме базов прототип на дизайна в Figma, по който си структурирахме разработката на frontend–а.
Свързахме frontend-а с backend-а чрез HTTP заявки и отговори.

Конкуренция

В процеса на създаване на приложението, бяха прегледани няколко решения, които имаха недостатъци, които ние разрешихме.
Например голяма част от тях изискват покупката на отделни устройства на всяка маса - таблет, специален екран вграден в масата или други, също така почти всички имаха ужасен интерфейс.
За разлика от конкуренцията, ние се целим да решим тези проблеми. Основния от тях е нуждата от покупката на отделни устройства за всяка маса. Ние разрешаваме този проблем като създадохме уеб приложение, което може да използва от всякакъв вид устройства, стига да могат да отварят уеб страници.
Самият сървър ще бъде хостван на само едно устройство, а като алтернатива може да бъде използван компютъра на заведението, ако има такъв.

Описание на приложението

Сорс кода на проекта може да бъде намерен в GitHub (https://github.com/DeliriumProducts/luncher-box/). След като е изтеглен проекта, той може да бъде конфигуриран, следвайки инструкциите в документацията в GitHub (https://github.com/deliriumproducts/luncher-box/wiki) След като бъде настроен, ще може да бъде достъпен на предварително нагласения порт (по default - 3000). За поддръжка на уеб-приложението, то ще бъде обновявано след всеки update, така и в GitHub, така и в host-а (https://luncherbox.deliriumproducts.me)

Заключение

Luncher Box - Интерактивно Меню ще спести много време на клиентите на заведения, като им предостави онлайн интерактивно меню с лесен за използване интерфейс. Те ще могат да го отворят през browser-ите си и да направят поръчка без да чакат, поради липсата на нужда от сервитьор. А собствениците на ресторанта ще могат да спестят пари от намаляването на служители.
Потребителски панел: luncherbox.deliriumproducts.me,
Админ панел: luncherbox.deliriumproducts.me/admin.
Въпреки предизвикателността на поставените от нас задачи, ние успяхме да преодолеем почти всички.
Подбрахме правилната технология - react.js, тъй като тя предоставя лесен начин за създаване на SPA и PWA, както и reusable компоненти, чрез които си създадохме графичния интерфейс.
За backend-а използвахме съвкупността от node.js, express.js, TypeORM, passport.js, nodemailer, socket.io поради гъвкавостта им и улесняването на целия процес на създаването на REST API.
Оформихме нашия интерфейс, базирано на най-новите тенденции през 2018. Целяхме се да е възможно най-интуитивен и максимално "User-friendly".
Приложението е все още в процес на разработка и се очаква до края на декември месец да бъде завършено.
Двамата автори сме допринесли за общото развитие на проекта.

Бъдеще и развитие

Както вече споменахме, проектът е все още в процес на разработка и доразвитие, съответно това означава, че има много потенциал за нови реализации. Някой от тях, които трябва да осъществим са:

- Мигриране на backend към Serverless технологии; *
- Масово импорт-ване и експорт-ване на продукти в различни формати като - JSON, YAML, CSV или XML;
- Плащане чрез кредитна/дебитна карта, в брой или криптовалута;
- Достигане до потенциални клиенти.

С две думи можем да кажем, че според нас проектът има огромен потенциал да се развие.

*не е сигурно