SPA single page application

**SQL** ali **strukturirani povpraševalni jezik za delo s podatkovnimi bazami** (

Povzetek

Slovenija velja za državo z veliko število restavracij, vendar le malo iz med njih uporablja napredne sisteme naročanja kot npr. ena izmed večjih verig s hitro prehrano McDonalds. Obstajali so projekti, ko so naredili aplikacije, ki bi/so vpeljale nekaj tehnologije v restavracije vendar v Sloveniji niso uspeli. Eden ključnih razlogov zakaj niso uspeli je bilo sabotiranje s strani natakarjev, saj so misli da bo tehnologija zamenjala njegove službe. V zavedanju njihovih problemov sem se odločil narediti diplomski nalogo na to temo. Rezultat je aplikacija, ki jo je mogoče uporabiti samo kot pregled hrane in pijače, ki jo ponuja restavracija ali pa kot sistem za naročanje. V diplomski nalogi najprej opišem kako sem prišel do ideje, sam razvoj aplikacije in njeno končno delovanje. Predstavim vse težave na katere sem naletel med samim razvojem. Na koncu sem pripravil tudi primerjavo s konkurenco na trenutnem trgu oziroma z aplikacijo, ki vem da ni uspela. Seveda sem pripravil tudi seznam izboljšav, ki bi mojo aplikacijo lahko pripeljalo med eno izmed boljših.

Uvod

* 1. Problem

Tehnologije v sedanjem času predstavljajo velik napredek na vseh področij. Na strani gostinstva ni drastičnih napredkov, saj sem to ugotovil kar na osnovi lastnih izkušen. Kot študent fakultete za računalništvo in informatiko sem opravljal delo v eni izmed restavracij, saj biti študent z dodatnim zaslužkom ni dilema. Opravljal sem delo dostavljavca hrane v eni izmed bližnji restavraciji, hkrati pa opazoval potek dela v kuhinji in strežbi. Kasneje sem napredoval v strežbo in kot programer hitro opazil stvari, ki bi se jih dalo izboljšati. Najbolj me je motilo nezadovoljstvo strank ob veliki zasedenosti restavracije. Preprosto en človek ne uspe postreči več kot eno mizo na enkrat. Zato sem si zamislil sistem za naročanje hrane, ki ne bi bil namenjen zamenjavi ljudi v strežbi, vendar samo kot pregledovalnik ( angl. Menu ) ali pa naročanju jedi in pijače. Stranka bi bila tista, ki bi se odločila ali želi pri naročanju uporabiti stik z osebo v strežbi ali bi preprosto naročila z uporabo aplikacije na tablici, ki bi bila vedno dosegljiva na vsaki mizi.

1. Uporabljene tehnologije in programske opreme

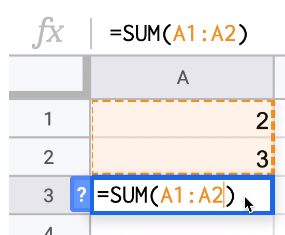
V diplomski nalogi sem se srečeval predvsem z naslednjimi programskimi jeziki: JavaScript, Python, SQL. Vse podrobnosti pa so opisane v spodnjih poglavjih.

* 1. Vue.js

JavaScript je programski jezik, zaradi katerega so spletne strani postale dinamične in bolj zmogljive. Z njem smo programsko kodo, ki je bila na strežniškem delu, preselili v brskalnik. Tako smo na začetku dobili veliko JavaScript programske kode povezane z različnimi HTML in CSS datotekami brez kakršne koli formalne organizacije (poznano kot «script»). Zaradi tega smo razvijalci začeli uporabljati JavaScript Frameworks, saj poenostavijo izdelavo aplikacij.

Vue je eden izmed mnogih kot npr. Angular, Ember, React,… poznan pa je predvsem zaradi enostavnosti za upravljanje in izvajanje testov. Vsem pa je skupna točka reaktivnost, vendar v drugačnem pomenu besede. Gre za to, da se aplikacija postopno prilagaja glede na vrednosti podatkov. Vue.js je namenjen izdelavi SPA projektov, saj vsebuje samo eno .HTML datoteko.

1. Kaj je reaktivnost?

Reaktivnost je programska paradigma, ki nam omogoča, da se na deklarativni način prilagodimo spremembam. Dober primer reaktivnosti je npr. funkcija SUM, ki jo uporabljamo v Excelu. Slika~\ref{Excel-SUM} prikazuje primer v Excelu.

Če vstavimo številko 2 v prvo celico in številko 3 v drugo celico ter izberemo funkcijo SUM teh dveh celic. Kot rezultat dobimo vsoto obeh številk skupaj, kar ni nič posebnega. Vendar če bomo spremenili vrednost prve celice, bo funkcija SUM avtomatsko posodobila skupno vrednost. Tako deluje tudi reaktivnost v aplikacijah za razliko, da je na nek podatek lahko vezanih več funkciji oziroma delov programske kode, ki se ob spremembi njegove vrednosti posodobijo.

# https://v3.vuejs.org/guide/reactivity.html#what-is-reactivity

1. Delovanje reaktivnost v Vue

Vue se torej v primerjavi z navadnim JavaScriptom sprehodi skozi podatke in njihove lastnosti (angl. Properties) pretvori v fukciji Getter in Setter, ki sta nevidni uporabniku. Poglejte si sliko ~\ref{ VueReacitivity} za lažjo predstavo.

Torej funkcija Getter pokliče instanco Watcher z namenom odvisnosti do drugih komponent. To pomeni, če je podatek označen kot odvisen (angl. Dependency) bodo nekateri deli programske kode oziroma funkcije poklicane vsakič, ko se spremeni vrednost podatka. Funkcija Setter pa obvesti instanco Watcher, vsakič ko se podatku spremeni lastnost. Ta poskrbi, da se pokliče funkcijo upodabljanja (angl. Render) tiste komponente, ki potem prikaže spremembe v samem pogledu aplikacije.



# https://vuejs.org/v2/guide/reactivity.html#How-Changes-Are-Tracked

1. Dodatki

Tako kot vsak programski jezik ima tudi Vue svoje dodatke, ki pomagajo pri razvoju aplikaciji. Spodnje sem uporabljal sam in so vredni omembe.

\item[Vue CLI] velja za standardno orodje za ekosistem Vue. Zagotavlja, da že pri gradnji novega projekta povezuje različne dodatke med seboj. To omogoča razviljacu, da se bolj osredotoči na programiranje in ne na povezovanje njih v celoto. Zadeva izgleda nekako tako, da preko CLI vmesnika izbereš kakšen projekt želiš. Imaš seveda že privzete nastavitve, vendar omogoča tudi nastavljanje po meri. Sam sem uporabil Vuex, Vue-Router, ESLint in Vuetify.

\item[Vuex] je knjižnica za shranjevanje vrednosti v aplikacijah Vue.js. Služi kot centralizirana baza podatkov za vse komponente v aplikaciji.

\item[Vue-Router] je uradni usmerjevalnik za Vue.js. Integrira se globoko z jedrom Vue.js, tako da poenostavi izdelavo SPA aplikacij. Usmerjevalki je mišljen v smislu usmerjanja na druge komponente (angl. Component), ki v Vue.js predstavljajo druge poglede, lahko bi rekli podobno kot podstrani.

\item[ESLint] je orodje za prepoznavanje in priporočanje o popravkih v programski kodi. Cilje je narediti kodo bolj usklajeno in urejeno, kar pripomore k izogibanju napak.

\item[Vuetify] je eden izmed mnogih uporabniških vmesnikov zgrajen na vrhu Vue.js. Za razliko od drugih vmesnikov je Vuetiy enostaven za učenje z več stotimi komponentami izdelanih po specifikacijah Material Design.

\item[Vue-devtools] je zgolj dodatek v brskalniku, ki omogoča lažje sledenje delovanja aplikacije in odpravljanje napak.

* 1. Python/Flask

Flask je eno izmed najbolj popularnih spletno aplikacijskih vmesnikov (angl. Freamwork). Zasnovan je tako, da omogoča hiter in enostaven začetek z možnostjo razširitve na zapletene aplikacije. V primerjavi z Django spletnim vmesnikov je za enak primer veliko bolj ekspliciten. Flask je prvotno zasnoval in razvil Armin Ronacher kot prvoaprilsko šalo leta 2010. Kljub taki predstavitvi je Flask postal izjemno priljubljen kot alternativa projektom narejenih v Django. Jaz sem ga uporabil za komunikacijo med aplikacijo in podatkovno bazo. Je enostaven za uporabo z veliko podpore na spletu.

* 1. MySQL/Toad DataModler

MySQL je eden od odprtokodnih sistemov za upravljanje s podatkovni bazami, ki za delo s podatki uporablja jezik SQL. Napisan je v programskem jeziku C in C++ in deluje v vseh modernih sistemih npr. Windows, Linux, OS X,… Prva verzija je bil razvita leta 1995 s strani Michael Widenius in David Axmark. Kratica My izhaja iz imena prve hčerke očeta Michaela.

Toad DataModler je orodje za izdelavo visokokakovostnih podatkovnih modelov. Omogoča izdelavo logičnih in fizičnih podatkovnih modelov, kar pripore k lažjem razumevanju in razvijanju podatkovne baze. Njegova najboljša funkcionalnost je, da lahko generiramo SQL kodo v različne podatkovne sisteme (npr. MySQL, Ingres, Microsoft Azurem, Microsoft Access, Mircrosoft SQL Server,…) . Program sem uporabil prav zato, da sem najprej naredil fizični podatkovni model ter nato generiral MySQL kodo, ki sem jo potem samo še uvozil v XAMPP.

* 1. XAMPP/Phpmyadmin

XAMPP je odprtokodni strežnik za razvoj spletnih aplikacij, ki se uporablja lokalno za testne projekte predno postanejo aktivni vsem preko spleta. Ta strežnik za razvoj spletnih aplikacij vsebuje tudi druge pred namestitvene aplikacije, kot so Apache spletni strežnik, MySQL podatkovno bazo, PHP in Perl. Deluje na vseh platformah Linux, Windows in Mac OS. Vse aplikacije se vklaplja preko XAMPP nadzorne plošče, ki jo lahko vidite na sliki …

PhpMyAdmin je ime za spletni vmesnik, ki omogoča upravljane z MySQL podatkovnimi bazami. Program je vključen v XAMPP in sicer znotraj aplikacije MySQL. Je zelo močno orodje in predvsem enostavno za uporabo. Sam sem ga uporabil pri testiranju poizvedb in dodajanju podatkov v podatkovno bazo.

Git Hub

Git Hub je spletna platforma za distribuirano upravljanje s programsko kodo, ki je bila ustanovljena februarja leta 2008 \cite{GitHub}. Njen poudarek je predvsem na hitrosti, integriteti podatkov in vzporednem toku dela. Namenjen je tako za samostojni kot tudi za kolaborantovi razvoj. Začneš tako, da ustvariš repozitoriji, ki predstavlja glavno programsko kodo. Iz tega se lahko naredi več podružnic (angl. Branch), ki predstavljajo kopijo glavne kode na kateri se lahko izvajajo testi oziroma odpravljajo težav. Podružnice potem združuješ in tako glavna programska koda ne vsebuje napak oziroma predstavlja vedno delujoč izdelek.

Git Hub Desktop je aplikacija za operacijske sisteme, ki je zelo enostavna. Ponuja vse funkcionalnosti kot na spletu, vendar zaradi uporabe lokalnih orodij veliko bolj priročna. Ko zaključiš delo narediš Commit in Push ter vse spremembe shraniš v repozitoriji na spletu. Tako nikoli ne izgubiš projekta oziroma imaš sinhronizacijo z ostalimi ljudmi na projektu.

Viusal Studio Code

Visual Studi Code je odprtokodno razvojno orodje, ki ga je razvil Microsoft leta 2015. Namenjeno je razvoju programov v operacijskem sistemu Windows. Omogoča razvoj aplikacij v jeziku C, C++, C#, Visual Basic .NET, JavaScript,…

Orodje v osnovi omogoča veliko funkcionalnosti, npr. razhroščevanje, popravljanje sintaktičnih napak, avtomatično dopolnjevanje programske kode, izvajanje programske kode v realnem času, dostop do CLI vmesnika, povezavo z GIT,… Kar ni vključeno v orodje je mogoče dodati z razširitvami, ki se jih brezplačno namesti znotraj orodja. Sam sem si namestil razširitev za Vue.js s katerimi sem pridobil avtomatsko dopolnjevanje kode in ESLint, da se je nepravilno urejena koda sproti označevala.

1. Struktura in razvoj aplikacije

3.1 Problematika

Z izdelavo spletne aplikacije sem želel zagotoviti lažje delovanje restavracij predvsem olajšati delo natakarjem, ki bi se potem lažje osredotočili na kakovost postrežbe pijač in jedi. Izboljšati sem želel čas postrežbe gostov, kar bi prizaneslo tudi k večjemu število postreženih gostov v nekem časovnem obdobju. Naredil sem analizo, ki si jo lahko pogledate v članku 3.2. Vsekakor ni moja aplikacija namenjena zamenjavi delovnih mest, vendar samo pohitritvi strežbe. To bi vplivalo tudi na zmanjšanje časa, ki ga gost porabi v primeru, da je natakar zelo zaseden. Obstajajo tablice, katere uporabljajo natakarji pri sprejemanju naročil, ki predvsem pomagajo, da si ne rabijo vsega zapomniti in da lahko pobirajo zaporedna naročila. Iz lastnih izkušenj vem, da z uporabo spomina pride hitreje do napak, saj si preprosto ne moreš vsega zapomniti. Težava je tudi v tem, da moraš za vsako pobrano naročilo do blagajne, da naročilo ne čaka in čimprej pride do kuharja oziroma natakarja, ki pripravlja pijače. Po navadi največ pri pobiranju naročil vzamejo gosti, ki se radi premišljujejo zadnje minute, kar ni nič narobe. Ko sem opravljal delo natakarja sem se spraševal zakaj ne bi ta čas raje posvetil strežbi oziroma pripravi naročil, kot pa pobiranju naročil strank, ki ne vedo kaj bi. Ta aplikacija bi še dodatno zmanjšala čas pobiranja naročil, saj natakar včasih pride prehitro ali pa prepozno, vendar z uporabo aplikacije bi bilo naročanje vedno na voljo.

Ideja

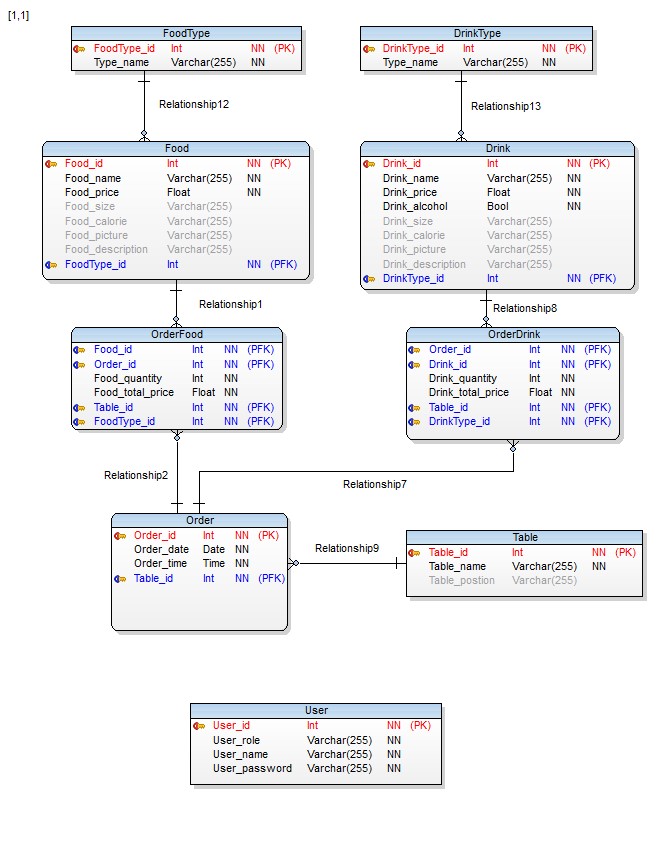
Aplikacijo sem si zamislil v treh pogledih, in sicer uporabnik oziroma gost, natakar in kuhar. Gost bi imel na voljo pregled vseh pijač in jedi, ki jih restavracija ponuja. S tem bi pripomogel k ekologiji, saj restavracije ne bi več potrebovale papirnatih menijev. Natakar in kuhar bi imela pregled nad naročili. Naročilo bi razdelil v dva dela, da ima tako natakar kot kuhar nadzor samo nad svojimi naročili. S tem bi omogočil, da se lahko kreira statistika za vsakega posebej. Kako sem to vse izvedel si lahko pogledate v poglavju 5. Delovanje aplikacije.

Analiza

* 1. Podatkovna baza

Najprej sem se lotil izdelavo podatkovne baze s pomočjo programa Toad DataModler, katero fizično obliko lahko vidite na sliki …

obliko lahko vidite na sliki



Sama podatkovna baza je sestavljena iz osmih tabel oziroma šestih, vendar zaradi dveh mnogo-proti-mnogo razmerij (angl. many-to-many) med tabelo Food in Order ter Drink in Order, sta nastali še dve vmesni tabeli OrderFood in OrderDrink.

FoodType table je namenjan vrsti jedi t.i. predjedi, glavne jedi, sladice,… Sestavljena je iz atributov: ID, Name.

DrinkType tabela je namejna vrsti pijač, npr. sokovi, koktejli, piva, topli napitki,… Sestavljena je enakih atirbutov kot FoodType.

Food tabela je namenjena samemu opisu hrane in je sestavljena iz atributov: ID, Name, Price, Size, Calorie, Picture in Description. V atribut Picture se zapiše ime slike, ki se prikaže v aplikaciji. Vse slike sem hranil v datoteki na lokalnem strežniku Apache.

Drink tabela je namenjan opisu pijače in je sestavljena iz enakih atributov kot Food le da ima še enega dodatnega z imenom Alcohol. Ta atribut je vrste BOOLEAN, kar pomeni da je lahko vrednosti 0 ali 1. Namenjena je označevanju pijač, ki vsebujejo alkohol.

OrderFood tabela je namenjena shranjevanju hrane v določenem naročilo. Zapis ne more obstajati če nima definiranega naročila. Tabela je nastala zaradi razmerja mnogo-proti-mnogo med tabelo Food in Order. Zato tabela vsebuje veliko tujih ključev in dveh atributov: Quantity in TotalPrice, katero ime pove zakaj sta namenjan. Prav tako je nastala tabela OrderDrink in vsebuje enake atribute, vendar za naročila pijač.

Table tabela je namejnea shranjevanju miz v restavracijah. Sestavljajo jo atributi: ID, Name in Position, v katerega se lahko bolj podrobno opiše lokacija mize.

Order tabela je namenjena zapisovanju naročil. Sestavljena je iz atributov: ID, Start, End. Vsebuje tudi tuji ključ ID od table Table, ki je zelo pomemben saj določuje na katero mizo je vezano naročilo.

Seveda končne različice podatkovne baze nisem naredil v prvem poskusu, npr. prvič sem pozabil na tabeli FoodType in DrinkType, kateri mi zelo poenostavijo delo pri selektivnem prikazovanju pijač in jedi v aplikaciji.

Po končanem urejanju fizične sheme sem naredil izvoz v MySQL 5.0 kodo, ki sem jo potem uvozil v spletni vmesnik phpMyAdmin. Uvoz ni nič posebnega, kreiral sem podatkovna baza in njeno ime ter naredil uvoz.

Ko je bila podatkovna baza uvožena sem jo potreboval samo še napolniti s podatki. Zadeve sem se prav tako lotil preko phpMyAdmin vmesnika, ki je zelo enostaven in omogoča dodajanje več podatkov na enkrat. Podatke sem črpal znanih slovenskih aplikacij za naročanje hrane na dom, eHrana in Wolt.

* 1. Strežnik

Za izdelavo strežnika sem uproabil Python s knjižnico Flask. Začel sem z povezavo s podatkovno bazo t

Opis CORS!

* 1. Aplikacija

1. Delovanje aplikacije
   1. Gost

V uporabniškem pogledu za gosta bi bila tudi tipka katera bi poklicala natakarja v primeru nejasnosti oziroma želji po osebnemu naročilu. Ko bi uporabnik izbral željene pijače in jedi bi s tipko zaključil naročilo katero bi se prikazalo potem v pogledu natakarja in kuharja. Ko natakar odda naročilo v svoji aplikaciji izbere, da ga je zaključil

* 1. Natakar
  2. Kuhar
  3. Admin
  4. Baza

1. Izboljšave in ugotovitve
2. chElektronsko naročanje se dan danes pojavlja na vsakem področju, saj ne glede na to ali ga potrebujemo ali ne nas sama tehnologija vodi v to smer. Sam sem se odločil za elektronsko naročanje v gostinstvu, in sicer zaradi lastnih izkušenj. Kot študent sem moral tudi sam služiti denar, zaradi želje ne obremenjevanja svojih staršev z denarnimi sredstvi. Delal sem kot natakar v eni izmed bližnjih gostiln. Tam sem ugotovil koliko stvari bi se delo bolje optimizirati, da bi stranke bile še bolj zadovoljne in dobile v svoje roke toplejšo hrano oziroma bi bilo celotno doživetje med stranko in natakarjem še boljše. Rešitev sem videl v elektronskih tablicah, saj bi jih lahko uporabili na več načinov. Gostinci bi jih lahko uporabili samo kot pripomoček oziroma meni jedi in pijač ali pa kot sistem za naročanje ob veliki zasedenosti natakarja.