



Elektrotehniška in računalniška šola

Dokumentacija zaključnega izdelka

ProjectManager

Izdelal: Aljaž Starc

Mentor: Iztok Gajšek

Mentor na šoli: Aljaž Gec

Program: Računalnikar (SPI)

Nova Gorica, Maj 2019

IZVLEČEK

Končni izdelek mora biti lahek za uporabo vsakodnevnemu uporabniku. Inštalacija sistema ter uporabniški vmesnik morata biti dokumentirana, uporabniški vmesnik se mora prilagoditi na uporabnikovo napravo ter strežniški del aplikacije mora delovati na vseh bolj znanih okoljih.

Največja problema sta bila dejstvo, da sem ob začetku komaj poznal tehnologije, ki sem jih uporabil in se jih naučil brez kakršnega koli prejšnjega znanja, ter to, da igralniške industrije ter terminologije na tem področju nisem poznal.

KLJUČNE BESEDE

- Spintec d.o.o.
- Docker
- JavaScript
- Spletna aplikacija
- Organizacija proizvodnje in prodaje

ABSTRACT

The finished product must be easy to use for every day user. System installation and user interface must be fully documented, user interface must align and resize according to user's device screen size and the server-side part of application must be able to run in any well-environment.

Biggest problems were the fact that I barely knew any of the technologies I used, so I learned most of them without any previous knowledge, and that I was new to gaming industry, so I was not familiar with the terminology.

KEYWORDS

- Spintec d.o.o.
- Docker
- JavaScript
- Web application
- Production and product organization

KAZALO

1 Uvod.....	5
1.1 Namen.....	5
1.2 Cilji.....	5
2 Sestava sistema.....	6
2.1 Docker.....	6
2.1.1 Kontejnerji.....	7
2.2 Backend.....	7
2.2.1 Api.....	8
2.2.2 Frontend server.....	8
2.3 Frontend.....	9
2.4 Podatkovna baza.....	9
2.4.1 MongoDB.....	9
2.4.2 MongoExpress.....	10
3 Izdelava aplikacije.....	11
3.1 Načrtovanje zasnove.....	11
3.2 Izdelava osnove, "temeljev" sistema.....	11
3.3 Postavitev podatkovne baze.....	11
3.4 Razvoj API-ja.....	11
3.5 Razvoj uporabniškega vmesnika.....	12
3.6 UI ter API.....	12
4 Zaključek.....	13

4.1 Ugotovitve.....	13
4.2 pogoji dela.....	13
4.3 možnosti nadaljnega dela ali razvoja.....	14
5 Literatura in viri.....	15
6 Slovar kratic in tujih izrazov.....	16

1 UVOD

V dokumentaciji je predstavljena aplikacija "ProjectManager", koncept delovanja aplikacije, struktura, uporabljene tehnologije, prednosti ter slabosti aplikacije.

1.1 NAMEN

Glaven namen aplikacije je, da poenostavi organizacijo produktov, ter postavi neke "standarde" sporočanja med oddelki v podjetju Spintec d.o.o. (npr. Vsak zabeležen produkt mora tudi imeti točno zabeležene mere).

1.2 CILJI

Za to aplikacijo imam, brez upoštevanja tistih, ki izpolnjujejo sam namen aplikacije, postavljena le dva osebna cilja;

1. Uporabniški vmesnik mora delovati na vsaki napravi, se prilagoditi nanjo ter hkrati še vedno ponujati dobro uporabniško izkušnjo
2. Backend mora delovati v vsakem bolj znanem operacijskem okolju (Fedora, Debian, Arch, CentOS, Windows, Windows Server, MacOS)

2 SESTAVA SISTEMA

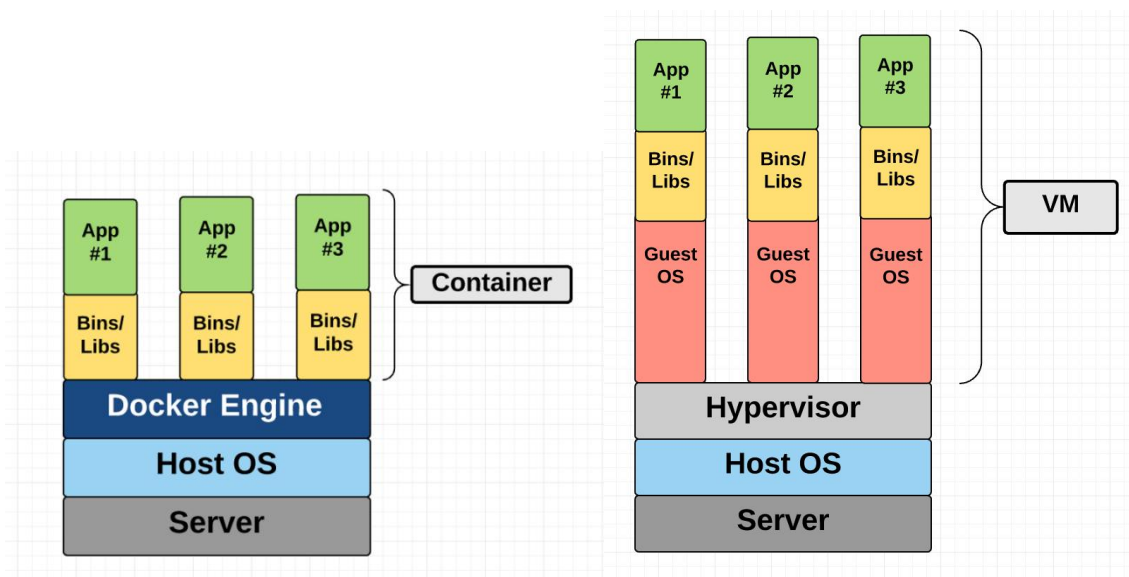
Sistem je sestavljen iz treh glavnih delov, in sicer iz “prednjega” dela (frontend), “zadnjega” dela (backend) ter podatkovne baze. Backend se nahaja v prvem kontejnerju¹, ki je javno dostopen preko vrat 81, podatkovna baza pa v drugem.

Če je aplikacija v “development mode”-u (načinu razvijanja), bosta na prvem kontejnerju dostopna API, preko vrat 81, ter uporabniški vmesnik preko vrat 82. Hkrati bo ob zagonu zagnan še tretji kontejner, v katerem se nahaja spletna aplikacija za manipuliranje podatkovne baze - aplikacija MongoExpress.

2.1 DOCKER

Docker je tehnologija, ki sem se je, tako kot tehnologije MongoDB, prvič dotaknil s tem projektom ter sem prepričan, da jo bom uporabljal v veliki večini projektov od sedaj naprej, saj ponuja ogromno fleksibilnosti, svobode, stabilnosti ter več-platformno uporabo.

Nisem se še poglobil kako tehnologija sama deluje v jedru, na površju pa si lahko Docker predstavljamo kot orodje za virtualizacijo (npr. [VirtualBox](#), [VmWare](#), [HyperV](#)), le da deluje na aplikacijskem nivoju in ne na nivoju operacijskega sistema. Prednost Docker tehnologije je, da lahko en kontejner zaženemo na več različnih operacijskih sistemih, kot na primer Windows, razne Linux distribucije, MacOS.



Shema sestave Docker kontejnerja (aplikacijski nivo)

Shema sestave virtualnega sistema (nivo operacijskega sistema)

¹ Kontejner - zaprt virtualen sistem na aplikacijskem nivoju kontroliran s aplikacijsko tehnologijo Docker.

2.1.1 KONTEJNERJI

Kot opisano zgoraj, Docker tehnologija ponuja "kontejnerizacijo", posredno tudi mobilnost, aplikacij.

Dockerjeve kontejnerje si je najlažje predstavljati kot kontejnerje, ki so uporabljeni v prevozništvu / transportu. Ne glede na to kaj kontejner vsebuje, ga lahko prenesemo iz vlaka na ladjo, iz ladje na kamjon (tovorno vozilo), iz kamjona na vlak ter obratno. V kontejner se lahko dostopa le skozi določene odprtine, kot na primer vrata ali odprtine za električno napeljavo ali hlajenje kontejnerja.

Podobno delujejo tudi Docker kontejnerji; ne glede na to kaj je v kontejnerju, ga lahko prestavimo iz Windows na Linux ali MacOS operacijsko okolje. Kontejner lahko na svojem prenosnem računalniku, ki uporablja Windows operacijski sistem, nastavimo ter konfiguriramo, nato pa ga prestavimo v produkcijo na strežnike, ki delujejo na CentOS² operacijskem sistemu. Tako kot v transportni kontejner, lahko do Docker kontejnerjev dostopamo le preko točno določenih portov³, na katerih lahko omejimo tudi dostop glede na vir povezave. Na primer, do kontejnerja neke podatkovne baze lahko dostopa le strežniški kontejner te aplikacije in ne kontejner, ki skrbi za beleženje analitik, ki se sicer lahko poveže le na naš strežniški kontejner.

2.2 BACKEND

Backend^(en) aplikacije, ali "zaledni sistem", je programiran v NodeJS⁴-u ter je sestavljen iz dveh glavnih končnih točk: API⁵-ja ter strežnika frontend-a.

Za procesiranje HTTP zahtev sem uporabil, za to najbolj uporabljen NodeJS modul, ExpressJS.

² CentOS - Community Enterprise Operating System, je distribucija RedHat (rhel) operacijskega sistema, ki jo vzdržuje skupnost CentOS razvijalcev ter uporabnikov.

³ Port - ali "mrežna vrata" so virtualne poti na neki napravi, ki so identificirane s unikatnim številom v rangu od 0 do 65535. Vsak port je dodeljen eni "storitvi" ki deluje na napravi. Vsaka bolj poznana storitev ima standardizirana vrata, kot na primer HTTP ter HTTPS mrežni promet, ki delujeta na vratih 80 ter 443.

⁴ NodeJS - je odprtokodno izvajalno okolje, uporabljeno za grajenje strežniških aplikacij, ki temelji na Googlovem V8 JavaScript okolju.

⁵ API - ang. Application Programming Interface je del aplikacije na backendu, ki procesira zahteve za kreiranje, modificiranje, prejemanje, brisanje ter preračunavanje ali obdelovanje informacij, običajno iz podatkovnih baz, do katerih uporabnik nima, ali nima polnega, dostopa. Npr. Proces prijave v sistem, procesiranje gesla ter uporabniškega imena ter končno dovoljenja ali zavrnitve uporabnika do neke vsebine.



NodeJS logo



ExpressJS logo

2.2.1 API

API končna točka posluša ter procesira zahteve, ki so poslane na `/api` pot⁶. Api poda vse zahteve GraphQL modulu, ki deluje kot glavni procesor zahtev.

GraphQL je Facebookova tehnologija, ki izboljša konstrukcijo API-ja. Dovoljuje da klient poda v zahtevi podatek, ki definira katere informacije sploh potrebuje, da se infrastrukture ne obremenjuje s nepotrebniimi informacijami.

Na primer; "tradicionalen" API, ki je postavljen na RestfulAPI standardih bo zahtevi za podatke o uporabniku z identifikatorjem 1 podal vse podatke, ki jih ima na razpolago; npr. Ime, priimek, uporabniško ime, e-poštni naslov, naslov slike računa, opis, datum registracije, datum zadnje uspešne prijave, itd. - vse to se skupaj sešteje v velik sklop informacij, ki potrebuje neko procesiranje, enkodiranje, dekodiranje, transport preko mreže ter končno tudi procesiranje samih podatkov na strani klienta.

Medtem ko s tehnologijo GraphQL lahko klient pove API-ju katere podatke o uporabniku samem potrebuje. Npr. izključno ime, priimek ter e-poštni naslov. V tem primeru se vse ostale podatke sploh ne vključi v končni sklop podatkov, kaj posredno razbremeni strežnike, mrežno infrastrukturo ter klienta, saj ni potrebna dodatna obdelava ter post-procesiranje prejetih podatkov na strani klienta.

2.2.2 FRONTEND SERVER

Strežnik aplikacijskih datotek posluša ter streže zahtevam na `root`⁷ (`/`) URI poti. Če datoteka, ki je zahtevana ne obstaja, npr. `/some/random/file.nohere`, bo zahtevo izpolnil ter postregel datoteko `index.html`, kar prepreči, da bi, na primer, zahteva `/signin` bila zavrnjena z `HTTP/404 NOT FOUND`, saj datoteka `signin` ne v resnici obstaja, a želimo upobniku vseeno prikazati človeku razumljive ter preproste poti do raznih delov aplikacij.

Vse statične datoteke, ki so uporabljene v uporabniškem vmesniku, so ob zagonu sistema obdelane s NodeJS modulom Webpack, ki vsako datoteko kompresira, odstrani vse presledke ter breakline ^(en), spremeni imena vseh spremenljivk v čim krajše (npr. "a", "b",

⁶ Pot - v kontekstu kot HTTP "path" (eng). Del URI naslova, ki sledi avtoritetnemu delu URI naslova. Npr. pot URI naslova <https://cloud.aljaxis.eu/apps/calendar> bi bil `/apps/calendar`

⁷ Root - koreninska pot. To je začetek poti nekega naslova. V priperu "root" URI naslova <https://cloud.aljaxis.eu/apps/calendar> bi bil <https://cloud.aljaxis.eu/>

“c”), prevede celotno kodo v starejše, ali boljše, standarde / funkcije, ki so podprti tudi v starejših brskalnikih.

2.3 FRONTEND

Uporabniški vmesnik je spletna aplikacija, zgrajena na tehnologiji VueJS. To je ogrodje za razvijanje reaktivnih spletnih aplikacij. Uporabil sem tudi dodatek VueRouter, ki ponuja fluidno brskanje spletne aplikacije - to sestavo spletnih aplikacij imenujemo “single page application” (eno-stranska spletna aplikacija), saj je v resnici celotna aplikacija le ena stran, ki dinamično nalaga potrebavane vsebine ter programatično spreminja naslov v naslovni vrstici brskalnika.

2.4 PODATKOVNA BAZA

Podatkovna baza deluje na tehnologiji MongoDB. Nahaja se v svojem kontejnerju, do katerega se lahko dostopa le preko porta (vrat) 27017, do katerih lahko dostopa le kontejner za streženje aplikacije ter API-ja.

2.4.1 MONGODB

MongoDB je podatkovna baza, ki hrani podatke v dokumentih. Za razliko od bolj znane MySQL podatkovne baze, ki temelji na SQL konceptu delovanja, temelji na NoSql konceptu delovanja, kar pomeni da do podatkov dostopamo preko objektov⁸.

Izbral sem jo, ker sem druge podatkovne baze, ki delujejo na SQL tehnologiji, že poznal ter znal z njimi operirati in sem se želel naučiti uporabljati tudi podatkovne baze, ki delujejo na NoSQL tehnologiji.

Izkazalo se je, da je bila to zelo dobra odločitev, saj je MongoDB zelo fleksibilna podatkovna baza. Večino stvari sem moral sam sprogramirati, kot na primer funkcija, ki izbriše beleženje aktivnosti o uporabniku, ko se le-tega uporabnika odstrani, kar mi ne bi bilo treba delati, če bi uporabil, bolj tradicionalno podatkovno bazo MySQL - a kje je zabava v tem?

Pozneje sem izvedel, da dosti tudi zelo velikih platform uporablja MongoDB, kot na primer Facebook, Ebay, Google, Cisco, ter še mnogi drugi ([vir](#)).

⁸ Objekt - podatek, ki ima svoj podatkovni tip ter nič ali več atributov, ko katerih dostopamo preko unikatnega indeksa vsakega atributa.

2.4.2 MONGOEXPRESS

MongoExpress je spletna aplikacija za manipuliranje podatkovne baze MongoDB ter podatkov v podatkovni bazi. Ima enako funkcijo kot bi jo imel sistem PhpMyAdmin⁹. Uporabil sem verzijo, ki je že nameščena ter pripravljena za uporabo v Docker kontejnerju.

Collection (incl. system.namespaces)	Count
Collections (incl. system.namespaces)	8
Data Size	1.59 MB
Storage Size	2.95 MB
Avg Obj Size #	936 Bytes
Objects #	1699
Extents #	0
Indexes #	5
Index Size	311 KB

Metric	Value	Metric	Value
Hostname	12c886e41f56	MongoDB Version	4.0.9
Uptime	99 seconds	Server Time	Tue, 04 Jun 2019 07:42:07 GMT
Current Connections	5	Available Connections	538655
Active Clients	17	Queued Operations	0
Clients Reading	1	Clients Writing	0
Read Lock Queue	0	Write Lock Queue	0
Time Spent Flushing	ms	Average Flush Time	ms
Total Inserts	0	Total Queries	1
Total Updates	0	Total Deletes	0

Sliki uporabniškega vmesnika aplikacije MongoExpress

<https://github.com/mongo-express/mongo-express-docker>

⁹ PhpMyAdmin - Spletna aplikacija zgrajena na jeziku PHP za manipulacijo podatkovnih baz ter podatkov v podatkovnih bazah kot so MySQL ter MariaDB.

3 IZDELAVA APLIKACIJE

3.1 NAČRTOVANJE ZASNOVE

Aplikacijo sem zasnoval na podlagi zahtev ter potreb, ki sem jih zaznal, ko sem se pogovarjal z drugimi zaposlenimi, ki bodo končni uporabniki te aplikacije. Določil sem katere bodo glavne tehnologije, ki jih bom uporabljal ter približno kako bo izgledal uporabniški vmesnik.

Nisem imel, ter še vedno nimam, točnega načrta, saj mi je nesmiselno zarisovati ter načrtovati celotno strukturo, ki se lahko pozneje izkaže za popolnoma neuporabno, ker ne bi pomislil nek problem, ali bi nanj naletel pozneje. Zato sem večino stvari improviziral med učenjem ter razvijanjem aplikacije.

Namesto na načrtovanje vnaprej, sem bil rajši pozoren, da pišem dobro, razvidno ter logično razumljivo kodo, katere funkcije bom lahko razumel tudi par mesecev po tem, ko jo napišem, in moram kaj dodati ali odvzeti.

3.2 IZDELAVA OSNOVE, "TEMELJEV" SISTEMA

Razvoj aplikacije sem začel z glavnim, osnovnim, nosilnim delom sistema: Docker kontejnerji ter avtomatizacija upravljanja kontejnerjev. Poskrbel sem, da je vse dobro testirano, stabilno, da samo kontejnerji delujejo tako na raznih operacijskih sistemih; Ubuntu 18.04, CentOS 7, Debian 7, Windows 10 Pro, Windows Server 2016.

3.3 POSTAVITEV PODATKOVNE BAZE

Postavitev MongoDB-ja je bila zelo enostavna procedura; nastavil sem glavne sistemske spremenljivke za uporabniško ime, geslo ter vrata skozi katera se lahko dostopa do podatkovne baze, nato sem zagnal kontejner v katerem se programska oprema MongoDB nahaja ter preveril, da vse deluje kot sem želel.

3.4 RAZVOJ API-JA

Glavni del backend-a ^(en) sem končno razvil v prvih par tednih, če ne upoštevamo "pod-sekcije" API-ja (en), ki ga do sedaj nisem še dokončno razvil in potrebuje če mnogo izboljšav ter dodatkov.

Razvoj raznih sekcij APIja je potekal v pol-urejenem zaporedju. Najprej sem razvil dele, ki so skrbeli za prijavo v sistem kot obstoječi uporabnik - to je vsekakor osnova za nadaljnji razvoj. Nato sem se lotil sekcije, ki skrbi za upravljanje z uporabniškimi računi (kreiranje, brisanje, posodobljanje uporabniških računov).

3.5 RAZVOJ UPORABNIŠKEGA VMESNIKA

Za razvoj uporabniškega vmesnika (UI ^(en)) sem porabil bistveno več časa kot za razvoj API-ja, saj je eden mojih ciljev, da se lahko aplikacijo uporablja na katerikoli napravi (npr. tablica, namizni ali prenosni računalnik, telefon). Zato sem porabil več časa, saj sem testiral veliko večino funkcij ter elementov, da delujejo brez komplikacij na zaslonih vseh velikosti ter oblik.

Preden sem se spustil v razvoj samega uporabniškega vmesnika, sem najprej nastavil vse nastavitve, sistemske spremenljivke ter določil arhitekturo aplikacije ter vrstni red elementov v datotečni strukturi.

3.6 UI TER API

Vsekakor sta uporabniški vmesnik ter API ^(en) tesno povezana en z drugim, saj sta odvisna eden od drugega. UI ^(en) toliko bolj od API-ja ^(en), a je API ^(en) tudi ravno toliko neuporaben brez dobrega uporabniškega vmesnika, saj ga končni uporabnik ne bi znal uporabljati.

Zaključek

3.7 UGOTOVITVE

Ugotovil sem, da imam sposobnosti, znanje ter zagnanost za se dokazati da znam narediti nekaj več ter boljše. V obdobju razvijanja aplikacije sem izvedel za mnogo programske opreme, za katero sem gotov, da jo bom uporabljal v novih projektih.

3.7.1 PREDNOSTI IN SLABOSTI APLIKACIJE

Glavne prednosti sistema ProjectManager, v primerjavi s, na podjetju trenutno uporabljenimi, excel tabelami, so:

- Deluje na vseh napravah, ki podpirajo sodobne brskalnike (npr. Mozilla FireFox)
- Strežniški del aplikacije se lahko v primeru nujne prestavi na drugo platformo brez kakršnih koli težav
- Ponuja bolj pregleden ter organiziran pogled zapisov ter bolj enostaven način urejanja zapisov
- Ponuja pregled statistik ter dogodkov v aplikaciji (brisanje, dodajanje, posodabljanje podatkov)

Seveda pa ima, kot vsak drug sistem, tudi slabosti. To so:

- Aplikacija je namensko dostopna izključno preko lokalne mreže v podjetju, kar omeji delo izključno na sedež podjetja. Delo na daljavo ni mogoče.
- Spletni standardi se konstantno posodabljujejo ter spreminjajo. To pomeni, da bo moral biti sistem v prihodnosti vsaj enkrat do dvakrat letno pregledan za spremembe v standardih ter po potrebi posodobljen.
- Splet je nevaren in brskalniki so še bolj. Zagotoviti je treba, da končni uporabniki znajo poskrbeti za varnost svojih podatkov ter podatkov, ki pripadajo podjetju, in bi v primeru izpusta v javnost ali v nedovoljeno last ogrozili delovanje ter poslovno varnost podjetja.

3.8 POGOJI DELA

Pogoji dela so bili v obdobju PUD-a, na podjetju Spintec d.o.o., zelo ugodni. Prva dva meseca sem delal v proizvodnji, kjer sem spoznal njihove izdelke, nato sem se za naslednja dva meseca "preselil" v prostore razvojnega oddelka kjer sem začel izdelovati aplikacijo ProjectManager. Zadnji mesec sem večinoma preživel v pisarnah, kjer sem imel na dosegu roke druge, toliko bolj izkušene, programerje.

3.9 MOŽNOSTI NADALJNJEGA DELA ALI RAZVOJA

Aplikacijo imam namen še naprej razvijati ter izboljševati, saj sem mnenja, da je odprtih še veliko možnosti za izboljšave sistema in uporabniškega vmesnika ter dodatne funkcije, ki bodo končnemu uporabniku izboljšale izkušnjo.

4 LITERATURA IN VIRI

- Docker Inc. (2019). Docker docs [spletna stran] - docs.docker.com/
- Facebook Inc. (2019). GraphQL [spletna stran] - graphql.github.io
- IBM inc. (2019). ExpressJS 4.x API reference [spletna stran] - expressjs.com
- JS Foundation (2019). ESLint - Pluggable JavaScript linter - eslint.org
- MongoDB inc. (2019). MongoDB Documentation [spletna stran] - docs.mongodb.com
- Node.js Foundation (2019). Docs | NodeJS [spletna stran] - nodejs.org
- Organizacija Mozilla (2019). MDN Web Docs [spletna stran] - developer.mozilla.org
- Organizacija VueJS (2019). VueJS documentation [spletna stran] - vuejs.org
- Organizacija Vuetify (2019). Vuetify documentation [spletna stran] - vuetifyjs.com/
- Organizacija webpack (2019). webpack documentation [spletna stran] - webpkack.js.org
- Sebastian McKenzie, drugi sodelavci (2019). Babel - The compiler for next generation JavaScript [spletna stran] - babeljs.io
- Stack Exchange Inc. (2019). StackOverflow network [forumi] - stackoverflow.com
- Wikimedia Foundation Inc. (2019). Uniform Resource Identifier - Wikipedia [spletna stran] - wikipedia.com
- Wikimedia Foundation Inc. (2019). Web API - Wikipedia[spletna stran] - wikipedia.com

5 SLOVAR KRATIC IN TUJIH IZRAZOV

Kratica	Angleški/Tuji izraz	Slovenski izraz
*****	Backend	Zaledni sistemi
*****	Breakline	Prelom vrstice
*****	Environment	Sistemsko okolje
*****	Frontend	Začetni sistemi
*****	Middleware	Vmesna programska oprema
*****	Route	Pot
API	Application programming interface	Vmesnik aplikacijskega programiranja
App	Application	Aplikacija
DB	Database	Podatkovna baza
Docs	Documentation	Dokumentacija
OS	Operating system	Operacijski sistem
PB	*****	Podatkovna baza
PUD	*****	Praktično Usposabljanje na Delu
URI	Uniform Resource Identifier	Enotni identifikator vira

Web	World Wide Web	Svetovni splet
Webapp	Web application	Spletna aplikacija