Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Vasja Škarabot

Spletna aplikacija za glasbene festivale

diplomsko delo

visokošolski strokovni študijski program
prve stopnje
računalništvo in informatika

Mentor: viš. pred. dr. Aljaž Zrnec

Ljubljana, 2024

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Kandidat: Vasja Škarabot
Naslov: Spletna aplikacija za glasbene festivale
Vrsta naloge: Diplomska naloga na visokošolskem programu prve stopnje Računalništvo in informatika
Mentor: viš. pred. dr. Aljaž Zrnec

Opis:
Besedilo teme diplomskega dela študent prepiše iz študijskega informacijskega sistema, kamor ga je vnesel mentor. V nekaj stavkih bo opisal, kaj pričakuje od kandidatovega diplomskega dela. Kaj so cilji, kakšne metode naj uporabi, morda bo zapisal tudi ključno literaturo.

Title: Web application for music festivals

Description:
opis diplome v angleščini

Na tem mestu zapišite, komu se zahvaljujete za pomoč pri izdelavi diplomske naloge oziroma pri vašem študiju nasploh. Pazite, da ne boste koga pozabili. Utegnil vam bo zameriti. Temu se da izogniti tako, da celotno zahvalo izpustite.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
API	Application Programming Interface	Aplikacijski programski vmesnik
DBMS	database management system	sistem za upravljanje podatkovnih baz
SVM	support vector machine	metoda podpornih vektorjev

Povzetek

Naslov: Spletna aplikacija za glasbene festivale

Avtor: Vasja Škarabot

Diplomska naloga predstavlja razvoj in implementacijo spletne aplikacije za glasbene festivale. Glavni namen aplikacije je, uporabnikom olajšati dostop do vseh pomembnih informacij, ter jim omogočiti medsebojno komunikacijo in preko le te graditi skupnosti s pomočjo forumov in klepetov.

Spletna aplikacija uporabniku omogoča pregled nad festivali, vključno z iskanjem prenočišč v bližini ter navodili za pot do prizorišča, forum na katerem lahko uporabniki objavljajo novice, delijo nasvete ter postavljajo morebitna vprašanja, ki se jim porajajo, ter klepeti, katere lahko uporabniki uporabijo z namenom razpravljanja in dogovarjanja za skupni obisk.

Za implementacijo čelnega dela smo uporabili ogrodje Nuxt.js, za zaledni del pa Django z razširitvijo Rest Framework in podatkovno bazo PostgreSQL. Vse skupaj poganjamo v Dockerju. Zemljevidi so implementirani s pomočjo platforme Mapbox, realnočasovno komunikacijo v klepetih omogoča Pusher, slike pa shranjujemo s pomočjo Firebase shrambe.

Ključne besede: glasba, festival, socialno omrežje, spletna aplikacija.

Abstract

Title: Web application for music festivals

Author: Vasja Škarabot

The thesis presents the development and implementation of a web application for music festivals. The main purpose of the application is to make it easier for users access to all relevant information, and to allow users to communicate with each other and build communities through forums and chats.

The web application provides the user with an overview of the festivals, including a search for nearby accommodation and directions to the venue, a forum where users can post news, share tips and ask any questions they may have, and chat rooms which users can use to discuss and arrange to visit together.

For the front-end implementation, we used the Nuxt.js framework, and for the back-end we used Django with the Rest Framework extension and the PostgreSQL database. We run everything inside Docker containers. Maps are implemented using the Mapbox platform, real-time communication in chats is enabled by Pusher, and images are stored using the Firebase Storage.

Keywords: music, festival, social network, web application.

Poglavje 1

Uvod

1.1 Motivacija

Vsako leto veliko ljudi obišče kakšen glasbeni festival. Pridobivanje informacij in samo načrtovanje obiska pa lahko kar hitro postane nadležno, in posledično lahko hitro pokvari uporabniško izkušnjo, že pred samim obiskom festivala. Najprej se uporabnik odloči za enega izmed festivalov. Ponavadi to stori na podlagi različnih dejavnikov, kot so cena, lokacija in datum poteka, veliko pa pomenijo tudi predhodne izkušnje ostalih. Problem pa je, da je potrebno vse te informacije iskati po različnih spletnih straneh saj rešitve, s katero bi lahko pridobili vse informacije na enem mestu ni. Poleg tega bi bilo smiselno implementirati rešitev, kjer si lahko uporabniki medeseboj pomagajo, s tem da delijo predhodne izkušnje, novosti in mnenja v obliki foruma ali pa morda celo dobijo nove prijatelje, v živo razpravljajo in mogoče nekoč celo skupaj obiščejo kakšen festival s pomočjo klepetov v živo. Z razvojem namenske spletne aplikacije bi obiskovalcem festivalov lahko pomagali tako, da bi se za vsako pomoč lahko obrnili na našo rešitev, istočasno pa bi se z medsebojno pomočjo gradile skupnosti za glasbene festivale.

Poglavje 2

Obstoječe rešitve

V uporabi trenutno ni nobene podobne rešitve, zaradi česar se obiskovalci festivalov poslužujejo različnih aplikacij in spletnih strani. Za postavljanje vprašanj in objavljanje novic sta to Reddit ali pa Tripadvisor. Ti rešitvi nista primarno namenjeni glasbenim festivalom, zato težko najdemo koristne podatke, saj vsebujeta le forume za večje festivale. Poleg tega mora vsak uporabljati več aplikacij za različne namene in zato nimamo vsega na eni platformi. Uporabnik mora tako npr. za iskanje informacij o festivalu najprej na spletno stran festivala, nato prebira različne forume (Reddit, Tripadvisor), zraven tega še išče prenočišča na Bookingu, potem pa se še na tretji platformi z ostalimi pogovarja/dogovarja (npr. Messenger). V tem poglavju bodo opisane naštete alternative, njihove prednosti in slabosti.

2.1 Tripadvisor

Tripadvisor je spletna platforma za iskanje prenočišč [23]. Je ena večjih platform namenjenih planiranju potovanj, ki uporabnikom omogoča tudi podajanje kritik in ocen.

- Prednosti:
  - Velik nabor uporabnikov in posledično kritik in ocen za prenočišča in razne dogodke.

  - Širok nabor funkcionalnosti. Poleg osnovnih funkcionalnosti še orodja za načrtovanje poti, vodniki in priporočila.

- Slabosti:
  - Vsebuje le najpopularnejše festivale.
  - Ocene bolj osredotočene na nastanitve in potovanje kot na sam festival.
  - Ne vsebuje nekaterih pomembnih informacij, kot so datumi, informacije o cenah in izvajalcih.
  - Uporabnik lahko z ostalimi uporabniki komunicira le posredno preko foruma. Za neposredno komunikacijo se mora poslužiti drugih platform.

2.2 Reddit

Reddit je forumsko socialno omrežje, kjer lahko uporabniki objavljajo, ostali pa nato te objave komentirajo in glasujejo [13]. Objave so razvrščene v podstrani imenovane subredditi oz. skupnosti. Na Redditu najdemo skupnosti za nekatere glasbene festivale, kjer se uporabniki obveščajo o novicah o festivalu in podajajo različna vprašanja. Je pa teh skupnosti precej malo, še tiste ki so, pa so po večini skupnosti največjih glasbenih festivalov na svetu (Coachella, Tomorrowland, UMF).

- Prednosti:
  - Pri večjih festivalih ogromen nabor vprašanj, razprav, nasvetov in novic.
  - Možnost realno-časovne komunikacije z ostalimi.

- Slabosti:
  - Večino osnovnih informacij mora uporabnik pridobiti na ostalih straneh.

  - Težko je najti skupnosti, ki niso namenjene le največjim festivalom.
  - Čeprav lahko uporabniki komunicirajo v realnem času, lahko to počnejo le v privatnih skupinah, ki niso del subredditov.

2.3 Woov

Woov je namenska aplikacija za obiskovalce glasbenih festivalov [44]. Povezani so z več kot 1000 dogodki po 40 državah, uporabnikom pa omogočajo vpogled v časovnice, zemljevid prizorišč in komunikacijo z ostalimi uporabniki. Namen aplikacije je pomagati obiskovalcem, ki so že na prizorišču.

- Prednosti:
  - Uporabniki lahko aplikacijo prenesejo na telefon.
  - Pomembne informacije kot so datumi, prizorišče, nastopajoči izvajalci.
  - Uporabniki si lahko pomagajo z zemljevidom, ki vključuje lokacije stojnic, wc-jev, odrov in polnilnih postaj.
  - Realno-časovna komunikacija.
- Slabosti:
  - Dogodki so običajno dodani nekaj dni pred začetkom festivala. Posledično si uporabnik za planiranje ne more dosti pomagati.
  - Aplikacija včasih deluje nelogična za nove uporabnike, saj so nekatere funkcionalnosti precej skrite.
  - Uporabniki ne morejo podati svojih mnenj.
  - Aplikaciji lahko prispevajo le partnerji. To ni nujno slabo, vendar omeji aplikacije le na določene festivale.

Poglavje 3

Zajem zahtev

Eden ključnih delov pri postopku razvoja programske rešitve je zajem zahtev, saj omogoča lažje razumevanje problema, ter zagotavlja, da sistem izpolnjuje vsa pričakovanja in usmerja celoten razvojni proces od načrtovanja do testiranja. V osnovi jih delimo na funkcionalne in nefunkcionalne zahteve.

3.1 Funkcionalne zahteve

Sistem mora uporabniku omogočiti:

- Registracijo in prijavo.
- Pregled nad glasbenimi festivali z možnostjo iskanja in filtriranja.
- Dodajanje festivalov in dodelitev moderatorskih pravic.
- Izpis podrobnosti glasbenega festivala, vključno z datumi, žanri in povezavo do uradne spletne strani.
- Pogled na zemljevid z lokacijo glasbenega festivala, ki se lahko razširi v navodila za pot.
- Iskanje prenočišč v okolici glasbenega festivala glede na podane parametre.

- Branje objav z možnostjo filtriranja in iskanja na forumu za določen festival.
- Objavljanje na forumih.
- Všečkanje in komentiranje objav.
- Dodajanje javnih klepetov za festivale.
- Pošiljanje sporočil v klepetih.
- Naročanje na obvestila za klepete ob novih sporočilih.

Moderatorji pa lahko povrh vsega še:

- Urejajo podatke za svoje festivale.
- Brišejo neprimerne objave.
- Brišejo neprimerne komentarje.

3.2 Nefunkcionalne zahteve

- Javni klepeti ter obvestila naj omogočajo realnočasovno komunikacijo. Uporabniku se morajo nova sporočila takoj prikazati na zaslonu.

3.3 Diagram primerov uporabe

Poglavje 4

Načrtovanje sistema

Poglavje 5

Uporabljene tehnologije in orodja

Odločili smo se za uporabo modernih, dobro poznanih in pogosto uporabljenih tehnologij in orodij. Za čelni del smo uporabili Nuxt [16], ki temelji na ogrodju Vue.js [38]. Za zaledni del smo izbrali Django [9] s knjižnico Django REST Framework [12], ki močno olajša razvoj API vmesnikov in upravljanje s podatkovno bazo. Za podatkovno bazo smo uporabili PostgreSQL [26]. Realnočasovno komunikacijo smo omogočili s pomočjo Pusher Channels [27]. Zemljevidi in navodila za pot so osnovani na platformi Mapbox [20]. Podatkovno bazo ter Django strežnik smo poganjali v Dockerju [10]. Vse te tehnologije in orodja ter njihov namen in uporabo v naši rešitvi smo podrobneje predstavili v naslednjem poglavju.

Poleg omenjenih glavnih tehnologij smo sicer uporabili še Vuetify.js za oblikovanje spletne strani [39], ter Git [11] in GitHub [1] za nadzor različic, za shranjevanje slik smo se odločili za Firebase Storage [8], za urejanje kode pa smo uporabljali Visual Studio Code [37]. Poleg tega smo uporabili nekaj Python knjižnic, med njimi BeautifulSoup4 [4] za strganje podatkov iz bookinga, ter Djoser [15], ki ponuja vnaprej definirane končne točke za registracijo, prijavo, odjavo in aktivacijo uporabniškega računa.

5.1 Nuxt.js

Nuxt.js (na kratko Nuxt) je brezplačno odprtokodno orodje za razvoj čelnega dela spletnih aplikacij z Vue.js. V naši aplikaciji smo ga uporabili zato, ker omogoča hiter, enostaven, predvsem pa učinkovit razvoj aplikacij.

Omogoča pisanje varne kode s samodejno generiranimi tipi, brez potrebe po učenju TypeScript [36]. Poleg tega Nuxt nudi tudi SSR (Server-Side Rendering), ki omogoča hitrejše čase nalaganja strani in SEO (Search Engine Optimization). Pri tem strežnik po začetni zahtevi odjemalca pošlje v celoti renderirano stran nazaj odjemalcu. JavaScript na strani odjemalca nato omogoči, da statična stran postane interaktivna Vue.js aplikacija (hidracija) [30].

Ena večjih prednosti je tudi enostavna konfiguracija poti na spletni strani, saj Nuxt samodejno razbere strukturo glede na vsebino mape pages/. V Vue.js npr. je za vsako novo dodano stran to potrebno storiti ročno.

5.2 Django in DRF

Django je Python ogrodje za razvoj spletnih aplikacij, ustvarjeno z namenom da uporabniku omogoči hiter razvoj, s tem, da vsebuje širok nabor orodij, ki poskrbijo za pogosto uporabljena opravila spletnega razvoja (avtentikacija, administracija, upravljanje z podatkovno bazo...). Temelji na vzorcu Model-View-Template, ki je v osnovi zelo podoben bolj znanemu Model-View-Controller. Model skrbi za podatke in vzdržuje povezavo s podatkovno bazo. View sprejema zahtevke in vrača odgovore, Template pa definira strukturo in postavitev strani, tipično z uporabo DTL (Django Template Language) [21].

Django REST Framework je razširitev za Django, posebej namenjena za gradnjo RESTful API-jev. V naši aplikaciji smo DRF v uporabili predvsem zaradi enostavnega dela s podatkovno bazo, enostavne serializacije podatkov (pretvorba kompleksnih podatkovnih tipov kot so Django modeli in poizvedbe v format JSON ali XMl) in že vnaprej pripravljenih pogledov za

nekatere osnovne CRUD operacije, ki jih z lahkoto lahko priredimo za naše namene. Poleg tega omogoča tudi enostavno pisanje pravic za dostop do API končnih točk.

5.3 PostgreSQL

Za podatkovno bazo smo uporabili PostgreSQL. PostgreSQL je zmogljiv odprtokoden objektno-relacijski DBMS (sistem za upravljanje s podatkovnimi bazami). Kljub temu, da je odprtokoden se lahko kosa z ostalimi ponudniki, kot sta npr. Oracle in MySQL. Prav zaradi stroškov, se zelo pogosto uporablja v različnih startupih in za v raziskovalne namene [17].

5.4 Pusher Channels

Realnočasovno komunikacijo v klepetih smo zagotovili s pomočjo storitve Pusher Channels. Pusher Channels deluje na tehnologiji WebSockets, ki omogoča vztrajne TCP povezave med strežnikom in odjemalcem. S tem lahko strežnik in odjemalec dosežeta takojšnjo izmenjavo sporočil na zelo učinkovit način z majhno zakasnitvijo, brez da bi odjemalec poslal zahtevo za prejem podatkov [41].

Pusher Channels delujejo na modelu objavi/naroči. Aplikacije se lahko naročajo na kanale v sistemu. Ko pride do sprememb, pa sistem objavi spremembo v ta isti kanal. Vse aplikacije naročene na ta kanal, so potem obveščene [27].

5.5 Mapbox

Mapbox je spletna platforma, ki razvijalcem ponuja širok nabor orodij za delo z zemljevidi. Podatke črpa iz različnih virov, med drugim tudi iz MapStreetBoxa in NASE. V naši aplikaciji smo uporabili večino glavnih storitev, ki jih omogoča in sicer [29]:

- **Mapbox Maps**, ki omogoča prikaz zemljevidov po meri z lokacijami. V naši aplikaciji prikazuje vse zemljevide.
- **Mapbox Navigation**, ki omogoča iskanje in izrisovanje poti. V naši aplikaciji služi izrisovanju poti od vnešene lokacije do festivala.
- **Geocoding API**, ki spreminja koordinate v naslove in obratno. V naši aplikaciji je uporabljen za človeku prijazen prikaz lokacije.
- **Mapbox Search**, ki omogoča iskanje lokacij. V naši aplikaciji je uporabljen za iskanje začetnih lokacij pri navigaciji.
- **Mapbox GL Geocoder**, ki omogoča interaktivno iskanje po zemljevidu. V naši aplikaciji je na voljo moderatorjem, da lahko na interaktiven način poiščejo lokacijo festivala.

5.6 Docker

Docker je platforma, ki omogoča ustvarjanje, izvajanje in nameščanje aplikacij v kontejnerjih/vsebnikih. Temu procesu pravimo kontejnerizacija. Pri kontejnerizaciji zapakiramo programsko kodo s knjižnicami operacijskega sistema in odvisnostmi, ki so potrebne za izvajanje kode. S tem ustvarimo lahko izvedljivo datoteko, ki se lahko enako izvaja na kateri koli infrastrukturi [42]. Prednosti uporabe Dockerja je veliko, v našem primeru pa smo ga uporabili predvsem zaradi izolacije okolja in lahke prenosljivosti. V Dockerju smo poganjali Django aplikacijo ter podatkovno bazo.

Poglavje 6

Razvoj rešitve

Poglavje 7

Predstavitev aplikacije

Poglavje 8

Možne izboljšave

Poglavje 9

Zaključek

LaTeX bi lahko najbolj preprosto opisali kot programski jezik namenjen oblikovanju besedil. Tako kot vsak visokonivojski programski jezik ima tudi LaTeX številne ukaze za oblikovanje besedila in okolja, ki omogočajo strukturiranje besedila.

Vsi LaTeXovi ukazi se začnejo z levo poševnico `\`, okolja pa definiramo bodisi s parom zavitih oklepajev { in } ali z ukazoma `\begin{ }` in `\end{ }`. Ukazi imajo lahko tudi argumente, obvezni argumenti so podani v zavitih oklepajih, opcijski argumenti pa v oglatih oklepajih.

Z ukazi torej definiramo naslov in imena avtorjev besedila, poglavja in podpoglavja in po potrebi bolj podrobno strukturiramo besedila na spiske, navedke itd. Posebna okolja so namenjena zapisu matematičnih izrazov, kratki primeri so v naslednjem poglavju.

Vse besedilne konstrukte lahko poimenujemo in se s pomočjo teh imen nato kjerkoli v besedilu na njih tudi sklicujemo.

LaTeX sam razporeja besede v odstavke tako, da optimizira razmike med besedami v celotnem odstavku. Nov odstavek začnemo tako, da izpustimo v izvornem besedilu prazno vrstico. Da besedilo skoči v novo vrstico pa ukažemo z dvema levima poševnicama. Število presledkov med besedami v izvornem besedilo ni pomembno.

Poglavje 10

Matematično okolje in sklicevanje na besedilne konstrukte

Matematična ali popolna indukcija je eno prvih orodij, ki jih spoznamo za dokazovanje trditev pri matematičnih predmetih.

Izrek 10.1 *Za vsako naravno število n velja*

$$n < 2^n. \tag{10.1}$$

Dokaz. Dokazovanje z indukcijo zahteva, da neenakost (**??**) najprej preverimo za najmanjše naravno število – 0. Res, ker je $0 < 1 = 2^0$, je neenačba (**??**) za $n = 0$ izpolnjena.

Sledi indukcijski korak. S predpostavko, da je neenakost (**??**) veljavna pri nekem naravnem številu n, je potrebno pokazati, da je ista neenakost v veljavi tudi pri njegovem nasledniku – naravnem številu $n + 1$. Računajmo.

$$\begin{aligned} n + 1 &< 2^n + 1 && (10.2)\\ &\leq 2^n + 2^n && (10.3)\\ &= 2^{n+1} \end{aligned}$$

Neenakost (??) je posledica indukcijske predpostavke, neenakost (??) pa enostavno dejstvo, da je za vsako naravno število n izraz 2^n vsaj tako velik kot 1. S tem je dokaz Izreka ?? zaključen. □

Opazimo, da je LaTeX številko izreka podredil številki poglavja. Na podoben način se lahko s pomočjo ukazov `\label` in `\ref` sklicujemo tudi na druge besedilne konstrukte, kot so med drugim poglavja, podpoglavja in plovke, ki jih bomo spoznali v naslednjem poglavju.

Poglavje 11

Plovke: slike in tabele

Slike in daljše tabele praviloma vključujemo v dokument kot plovke. Pozicija plovke v končnem izdelku ni pogojena s tekom besedila, temveč z izgledom strani. LaTeX bo skušal plovko postaviti samostojno, praviloma na mestu, kjer se pojavi v izvornem besedilu, sicer pa na vrhu strani, na kateri se na takšno plovko prvič sklicujemo. Pri tem pa bo na vsako stran končnega izdelka želel postaviti tudi sorazmerno velik del besedila. V skrajnem primeru, če imamo res preveč plovk na enem mestu besedila, ali če je plovka previsoka, se bo LaTeX odločil za stran popolnoma zapolnjeno s plovkami.

Poleg tega, da na položaj plovke vplivamo s tem, kam jo umestimo v izvorno besedilo, lahko na položaj plovke na posamezni strani prevedenega besedila dodatno vplivamo z opcijami `here, top` in `bottom`. Zelo velike slike je najbolje postaviti na posebno stran z opcijo `page`. Skaliranje slik po njihovi širini lahko prilagodimo širini strani tako, da kot enoto za širino uporabimo kar širino strani, npr. `0.5\textwidth` bo raztegnilo sliko na polovico širine strani. Sliko lahko po potrebi tudi zavrtimo za 90 stopinj in jo razstegnemo na višino strani. Tako bodo podrobnosti na sliki lažje berljive in prostor na strani bo bolje izkoriščen.

Na vse plovke se moramo v besedilu sklicevati, saj kot beseda plovka pove, plovke plovejo po besedilu in se ne pojavijo točno tam, kjer nastopajo v izvornem besedilu. Vendar naj bosta sklic na plovko v besedilu in sama

plovka v oblikovanem besedilu čim bližje skupaj, tako da bralcu ne bo potrebno listati po diplomi. Upoštevajte pa, da se naloge tiska dvostransko in da se hkrati vidi dve strani v dokumentu! Na to, kje se bo slika ali druga plovka pojavila v postavljenem besedilu lahko torej najbolj vplivamo tako, da v izvorni kodi plovko premikamo po besedilu nazaj ali naprej!

Tabele je najbolje oblikovati kar neposredno v LaTeXu, saj za oblikovanje tabel obstaja zelo fleksibilno okolje `tabular`. Slike pa je po drugi strani pogosto najlažje oblikovati oziroma izdelati z drugimi orodji in programi, rezultate shraniti v formatu `.pdf` ali `.jpeg` in nato v LaTeXu le vključiti ustrezno slikovno datoteko. Za pisanje besed, ki so vključene v slike, uporabite pisave/fonte, ki so čimbolj podobne pisavam v samem besedilu.

Knjižnica `https://en.wikibooks.org/wiki/LaTeX/PGF/TikZ` pa omogoča risanje raznovrstnih grafov neposredno v okolju LaTeX.

Na vse tabele in slike se moramo v besedilu sklicevati, saj kot plovke v oblikovanem besedilu niso nujno na istem mestu kot v izvornem besedilu. Pri sklicevanju na slike uporabimo veliko začetnico, npr. "glej Sliko **??**", saj gre za ime slike.

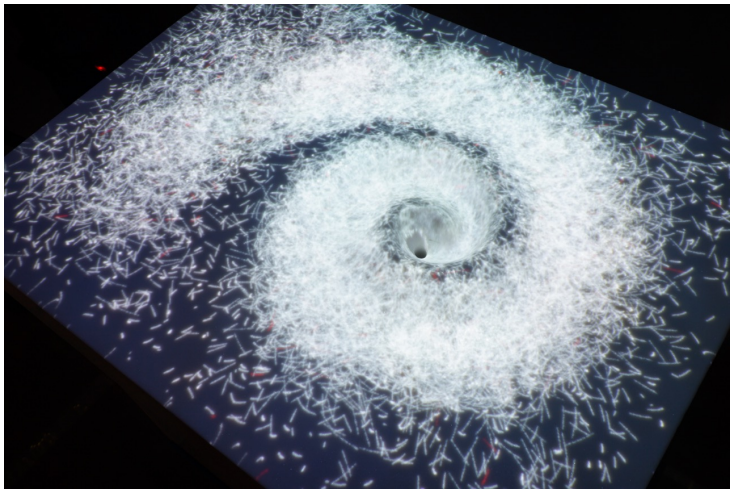
11.1 Formati slik

V dokument LaTeX lahko vključimo slike različnih formatov, tako bitne slike kot vektorske slike. Najbolj primerne so slike v formatu `.pdf`, saj je tudi samo oblikovano besedilo v tem formatu, in slike v formatu `.jpeg`. Slika **??** je npr. v formatu `.jpeg`.

11.2 Podnapisi k slikam in tabelam

Vsaki sliki ali tabeli moramo dodati podnapis, ki na kratko pojasnjuje, kaj je na sliki ali tabeli. Če nekdo le prelista diplomsko delo, naj bi že iz slik in njihovih podnapisov lahko na grobo razbral, kakšno temo naloga obravnava.

Če slike povzamemo iz drugih virov, potem se moramo v podnapisu k



Slika 11.1: Virtualno obogatena skulptura [35]. Rezultate računalniško generirane animacije z video projektorjem projeciramo na kamnito skulpturo, da ustvarimo vtis, kot da bi po skulpturi polzele vodne kapljice [33, 34].

taki sliki sklicevati na ta vir!

Poglavje 12

Struktura strokovnih besedil

Strokovna besedila imajo ustaljeno strukturo, da bi lahko hitreje in lažje brali in predvsem razumeli taka besedila, saj načeloma vemo vnaprej, kje v besedilu se naj bi nahajale določene informacije.

Najbolj osnovna struktura strokovnega besedila je:

naslov besedila, ki naj bo sicer kratek, a kljub temu dovolj poveden o vsebini besedila,

imena avtorjev so običajno navedena po teži prispevka, prvi avtor je tisti, ki je besedilo dejansko pisal, zadnji pa tisti, ki je raziskavo vodil,

kontaktni podatki – poleg imena in naslova institucije je potreben vsaj naslov elektronske pošte,

povzetek je kratko besedilo, ki povsem samostojno povzame vsebino in izpostavi predvsem glavne rezultate ali zaključke,

ključne besede so tudi namenjene iskanju vsebin med množico člankov,

uvodno poglavje uvede bralca v tematiko besedila, razloži kaj je namen besedila, predstavi področje o katerem besedilo piše (če temu ni namenjeno v celoti posebno poglavje) ter na kratko predstavi strukturo celotnega besedila,

poglavja tvorijo zaokrožene celote, ki se po potrebi še nadalje členijo na podpoglavja, namenjena so recimo opisu orodij, ki smo jih uporabili pri delu, teoretičnim rezultatom ali predstavitvi rezultatov, ki smo jih dosegli,

zaključek še enkrat izpostavi glavne rezultate ali ugotovitve, jih primerja z dosedanjimi in morebiti poda tudi ideje za nadaljno delo,

literatura je seznam vseh virov, na katere smo se pri svojem delu opirali, oziroma smo se na njih sklicevali v svojem besedilu.

Naslove poglavij in podpoglavij izbiramo tako, da lahko bralec že pri prelistavanju diplome in branju naslovov v grobem ugotovi, kaj je vsebina diplomskega dela.

Strokovna besedila običajno pišemo v prvi osebi množine, v nevtralnem in umirjenem tonu. Uporaba sopomenk ni zaželena, saj želimo zaradi lažjega razumevanja za iste pojme vseskozi uporabljati iste besede. Najpomembnejše ugotovitve je smiselno večkrat zapisati, na primer v povzetku, uvodu, glavnem delu in zaključku. Vse trditve naj bi temeljile bodisi na lastnih ugotovitvah (izpeljavah, preizkusih, testiranjih) ali pa z navajanjem ustreznih virov.

Največ se lahko naučimo s skrbnim branjem dobrih zgledov takih besedil.

Poglavje 13

Pogoste napake pri pisanju v slovenščini

V slovenščini moramo paziti pri uporabi pridevnikov, ki se ne sklanjajo, kot so npr. kratice. Pravilno pišemo "model CAD" in **ne** "CAD model"!

Pri sklanjanju tujih imen ne uporabljamo vezajev, pravilno je Applov operacijski sistem in **ne** Apple-ov.

Pika, klicaj in vprašaj so levostični: pred njimi ni presledka, za njimi pa je presledek. Klicajev in vprašajev se v strokovnih besedilih načeloma izogibamo. Oklepaji so desnostični in zaklepaji levostični: (takole).

Z narekovaji označujemo premi govor, naslove, citate ali pa z njim dajemo besedam poseben pomen. Narekovaji so stična ločila. Ločimo različne narekovaje, vendar je v LaTeXu najbolj enostavno uporabiti "dvojni narekovaj zgoraj". Za druge vrste narekovajev je potrebno uvoziti dodatne pakete ali fonte. Besede lahko vizualno označimo tudi z uporabo drugih pisav iz iste družine, npr. kurzivno in krepko pisavo, vendar pri uporabi teh fontov ne smemo pretiravati.

Vezaj je levo in desno stičen, npr. `slovensko-angleški slovar` in ga pišemo z enim znakom za pomišljaj. V slovenščini je presledek pred in po pomišljaju: Pozor – hud pes! (`Pozor -- hud pes!`). V angleščini pa je za razliko pomišljaj levo in desno stičen in se v LaTeXu piše s tremi pomišljaji:

`---`. S stičnim pomišljajem pa lahko nadomeščamo predlog od ... do, denimo pri navajanju strani, npr. preberite strani 7–11 (`7--11`).

"Pred ki, ko, ker, da, če vejica skače". To osnovnošolsko pravilo smo v življenju po potrebi uporabljali, dopolnili, morda celo pozabili. Pravilo sicer drži, ampak samo če je izpolnjenih kar nekaj pogojev (npr. da so ti vezniki samostojni, enobesedni, ne gre za vrivek itd.). Povedki so med seboj ločeni z vejicami, razen če so zvezani z in, pa, ter, ne–ne, niti–niti, ali, bodisi, oziroma. Sicer pa je bolje pisati kratke stavke kot pretirano dolge.

V računalništvu se stalno pojavljajo novi pojmi in nove besede, za katere pogosto še ne obstajajo uveljavljeni slovenski izrazi. Kadar smo v dvomih, kateri slovenski izraz je primeren, si lahko pomagamo z iskanjem na kakšnem od slovenskih spletnih slovarjev [31], še posebej v *Islovarju* Slovenskega društva Informatika [14] in Slovarju Slovenskega društva za razpoznavanje vzorcev [40]. Sicer pa glavni vir za reševanje slovenskih jezikovnih zadreg spletišče *Fran* [28].

Poglavje 14

Koristni nasveti pri pisanju v LaTeXu

Programski paket LaTeX je bil prvotno predstavljen v priročniku [19] in je v resnici nadgradnja sistema TeX avtorja Donalda Knutha [43], znanega po svojih knjigah o umetnosti programiranja ter Knuth-Bendixovem algoritmu [18]. TeX in njegove izpeljanke so odprtokodni programi.

Različnih implementacij LaTeXa je cela vrsta. Za OS X priporočamo TeXShop, za Windows PC pa MikTeX. Spletna verzija, ki poenostavi sodelovanje pri pisanju, je Overleaf.

Včasih smo si pri pisanju v LaTeXu pomagali predvsem s tiskanimi priročniki [19], danes pa je enostavneje in hitreje, da ob vsakem problemu za pomoč enostavno povprašamo Google, saj je na spletu cela vrsta forumov za pomoč pri TeXiranju.

LaTeX včasih ne zna pravilno deliti slovenskih besed, ki vsebujejo črke s strešicami. Če taka beseda štrli preko desnega roba, lahko LaTeXu pokažemo, kje se tako besedo deli takole: `ra\-ču\-nal\-ni\-štvo`. Katere vrstice so predolge lahko vidimo tako, da dokument prevedemo s vključeno opcijo `draft`: `\documentclass[a4paper, 12pt, draft]{book}`.

Predlagamo, da v izvornem besedilu začenjate vsak stavek v novi vrstici, saj LaTeX sam razporeja besede po vrsticah postavljenega besedila. Bo pa

zato iskanje po izvornem besedilu in popravljanje veliko hitrejše. Večina sistemov za TEXiranje sicer omogoča s klikanjem enostavno prestopanje iz prevedenega besedila na ustrezno mesto v izvornem besedilu in obratno.

Boljšo preglednost dosežemo tako kot pri pisanju programske kode – z vizualnim urejanjem kode in izpuščanjem praznih vrstic. Pri spreminjanju in dodajanju izvornega besedila je najbolje pogosto prevajati, da se sproti prepričamo, če so naši nameni pravilno izpolnjeni.

Kadar besedilo, ki je že bilo napisano z nekim vizualnim urejevalnikom (npr. z Wordom), želimo prenesti v LATEX, je tudi najbolje to delati postopoma s posameznimi bloki besedila, tako da lahko morebitne napake hitro identificiramo in odpravimo. Za prevajanje Wordovih datotek v LATEX – in obratno – sicer obstajajo prevajalniki, ki pa običajno ne generirajo tako čisto logično strukturo besedila, kot jo sicer LATEX omogoča. Hiter in enostaven način prevedbe besedila, ki zahteva sicer ročne dopolnitve, lahko poteka tudi tako, da besedilo urejeno z vizualnim urejevalnikom najprej shranimo v formatu pdf, nato pa to besedilo uvozimo v urejevalnik, kjer urejamo izvorno besedilo v formatu LATEX.

14.1 Pisave v LATEXu

V LATEXovem okolju lahko načeloma uporabljamo poljubne pisave. Izbira poljubne pisave pa ni tako enostavna kot v vizualnih urejevalnikih besedil. Posamezne oblikovno medseboj usklajene pisave so običajno združene v družine pisav. V LATEXu se privzeta družina pisav imenuje Computer Modern, kjer so poleg navadnih črk (roman v LATEXu) na voljo tudi kurzivne črke (*italic* v LATEXu), krepke (**bold** v LATEXu), kapitelke (SMALL CAPS v LATEXu), linearne črke (sans serif v LATEXu), pisava pisalnega stroja (`typewriter` v LATEXu) in nekatere njihove kombinacije, npr. krepke linearne črke (**sans serif** v LATEXu). V istem dokumentu zaradi skladnega izleda uporabljamo običajno le pisave ene družine. Pomembna je tudi konsistentna raba večih pisav in da ne pretiravamo z mešanjem različnih pisav.

Ko začenjamo uporabljati LaTeX, je zato najbolj smiselno uporabljati kar privzete pisave, s katerimi je napisan tudi ta dokument. Z ustreznimi ukazi lahko nato preklapljamo med navadnimi, kurzivnimi, krepkimi in drugimi pisavami. Zelo enostavna je tudi izbira velikosti črk. LaTeX odlično podpira večjezičnost, tudi v sklopu istega dokumenta, saj obstajajo pisave za praktično vse jezike, tudi take, ki ne uporabljajo latinskih črk.

Za prikaz programske kode se pogosto uporablja pisava, kjer imajo vse črke enako širino, kot so črke na mehanskem pisalnem stroju (`typewriter` v LaTeXu).

Najbolj priročno okolje za pisanje kratkih izsekov programske kode je okolje `verbatim`, saj ta ohranja vizualno organizacijo izvornega besedila in ima privzeto pisavo pisalnega stroja.

```
for (i = 0; i < 100; i++)
   for (j = i; j < 10; j++)
      some_function(i, j);
```

Poglavje 15

Kaj pa literatura?

Kot smo omenili že v uvodu, je pravi način za citiranje literature uporaba BibLaTeXa [6]. BibLaTeX zagotovi, da pri določeni vrsti literature ne izpustimo nobene obvezne informacije in da vse informacije dosledno navajamo na enak način in po istem vrstnem redu. BibLaTeX je nadgradnja starejšega sistema BibTeX. Novejši sistem je bolje prilagojen slovenščini in navajanju spletnih virov. Sicer pa so starejše datoteke `.bib` kompatibilne z BibLaTeXom.

Osnovna ideja BibLaTeXa je, da vse informacije o literaturi zapisujemo v posebno datoteko, v našem primeru je to `literatura.bib`. Vsakemu viru v tej datoteki določimo simbolično ime. V našem primeru je v tej datoteki nekaj najbolj značilnih zvrsti literature, kot so knjige [19], članki v revijah [32] in zbornikih konferenc [7], poglavja v knjigah [25], spletni viri [31, 33], tehnično poročilo [2], diplome [3] itd. Diploma [3] iz leta 1990 je bila prva diploma na tedanji Fakulteti za elektrotehniko in računalništvo, ki je bila oblikovana z LaTeXom! Reference, ki so na spletnih straneh arhivirane v elektronski obliki, imajo običajno številko DOI (`http://dx.doi.org`), ki jo zato tudi vključimo v izpis literature in tako bralcu elektronske verzije naše publikacije ponudimo neposredno povezavo do elektronske kopije te reference.

Po vsaki spremembi pri sklicu na literaturo moramo najprej prevesti izvorno besedilo s prevajalnikom LaTeX, nato s prevajalnikom BibLaTeX, ki

ustvari datoteko `vzorec_dip_Seminar.bbl`, in nato še dvakrat s prevajalnikom LaTeX. V okolju Overleaf je to večkratno prevajanje z različnimi prevajalniki uporabniku skrito. Zato tudi začetnim uporabnikom LaTeXa svetujemo uporabo Overleafa.

Kako se spisek literature nato izpiše (ali so posamezni viri razvrščeni po vrstnem redu sklicevanja, ali po abecedi priimkov prvih avtorjev, ali se imena avtorjev pišejo pred priimki itd.) je odvisno od parametrov paketa BibLaTeX. V diplomi bomo uporabili parametre `style=numeric`, kar pomeni, da bodo sklici na literaturo v besedilu označeni z zaporednimi številkami, za vrstni red izpisa referenc pa `sorting=nty`, kar pomeni, da bodo reference urejene po priimkih prvih avtorjev, nato po naslovu reference in nazadnje po letu izdaje [5]. Zato je potrebno pri določenih zvrsteh literature, ki nima avtorjev, dodati parameter `key`, ki določi vrstni red vira po abecedi.

Ko začenjamo uporabljati BibLaTeX je lažje, če za urejanje datoteke `.bib` uporabljamo kar isti urejevalnik kot za urejanje datotek `.tex`, čeprav obstajajo tudi posebni urejevalniki oziroma programi za delo z datotekami `.bib`.

Le če se na določen vir v besedilu tudi sklicujemo, se bo ta vir pojavil tudi v spisku literature. Tako je avtomatično zagotovljeno, da se na vsak vir v seznamu literature tudi sklicujemo v besedilu diplome. V datoteki `.bib` imamo sicer lahko veliko več virov za literaturo, kot jih bomo uporabili v diplomi.

15.1 Zbiranje virov za seznam literature

Vire v formatu `.bib` lahko enostavno poiščemo in prekopiramo iz spletnih strani založnikov ali različnih akademskih spletnih portalov za iskanje znanstvene literature. Izvoz referenc v Google učenjaku še dodatno poenostavimo, če v nastavitvah izberemo BibTeX kot želeni format za izvoz navedb. Navedbe, ki jih prekopiramo iz Google učenjaka in drugih podobnih akademskih portalov, moramo pred uporabo nujno preveriti, saj so taki navedki pogosto generirani povsem avtomatično in lahko vsebujejo napačne ali nepopolne

podatke. Najpogosteje je napačen tip publikacije!

Pri sklicevanju na literaturo na koncu stavka moramo paziti, da je pika po ukazu `\cite{ }`. Da LaTeX ne bi delil vrstico ravno tako, da bi sklic na literaturo v oglatih oklepajih začel novo vrstico, lahko pred sklicem na literaturo dodamo nedeljiv presledek: `~\cite{ }`.

Običajno se v besedilu sklicujemo na nek vir ali več virov na koncu trdilnega stavka. Kadar pa omenimo avtorja nekega vira, pa sklic običajno vstavimo za njegovim priimkom.

Dandanes se skoraj vsi pri iskanju informacij vedno najprej lotimo iskanja preko svetovnega spleta. Rezultati takega iskanja pa so pogosto spletne strani, ki danes obstajajo, jutri pa jih morda ne bo več, ali pa vsaj ne v taki obliki, kot smo jo prebrali. Smisel navajanja literature pa je, da tudi po dolgih letih nekdo, ki bo bral vašo diplomo, lahko poišče vire, ki jih navajate v diplomi.

Znanstveni rezultati, ki so objavljeni v obliki recenziranih člankov, bodisi v konferenčnih zbornikih, še bolje pa v znanstvenih revijah, so veliko bolj izčiščen in zanesljiv vir informacij, saj so taki članki šli skozi recenzijske postopke. Predvsem pa so taki članki stabilen vir informacij, saj se načeloma po njihovi objavi ne spreminjajo več. Skoraj vsi ti članki so dandanes dosegljivi tudi v elektronski obliki, bodisi v arhivih založnikov, univerzitetnih repozitorijih ali tudi na osebnih spletnih straneh njihovih avtorjev. Zato na svetovnem spletu začenjamo iskati vire za strokovna besedila predvsem preko akademskih spletnih portalov, kot so npr. Google učenjak, Research Gate ali Academia, saj so na teh portalih rezultati iskanja le akademske publikacije. Če je za dostop do nekega članka potrebno plačati, se obrnemo za pomoč in dodatne informacije na našo knjižnico.

Za označevanje člankov, ki so na voljo v elektronski obliki, se je v zadnjem času uveljavila oznaka DOI (`https://www.doi.org`), kar močno olajša iskanje teh referenc na spletu. Založniki tudi starejšim člankom, ki so na voljo v elektronski obliki, za nazaj določajo oznake DOI. Zato poskusite poiskati ustrezno oznako DOI za vsak članek, ki ga citirate in jo vključite v seznam

literature.

Če res ne gre drugače, pa je pomembno, da pri sklicevanju na običajni spletni vir vedno navedemo tudi datum, kdaj smo dostopali do tega vira.

Z uporabo BIBLATEXa je možno natisniti seznam literature posebej za določene vrste referenc, na primer za članke v znanstvenih revijah, članke v konferenčnih zbornikih in poglavja v knjigah, kot je prikazano tudi v tem vzorcu diplome. Pri zbiranju literature si zato prizadevajte čimbolj napolniti sezname teh treh vrst referenc.

Poglavje 16

Skladnost s standardom PDF/A

Elektronsko verzijo diplome je potrebno oddati preko sistema STUDIS v formatu PDF/A [22, 24], natančneje v formatu PDF/A-1b. PDF/A format je namenjen dolgoročnemu arhiviranju elektronskih dokumentov. Dokument v formatu PDF/A mora vsebovati vse potrebne informacije za prikazovanje in tiskanje dokumenta. To pomeni, da mora dokument vsebovati vso besedilo, vse slike, fonte in barvne informacije. Prva verzija standarda PDF/A, to je PDF/A-1 je bil objavljena leta 2005. Standard PDF/A-1 določa dva nivoja skladnosti: PDF/A-1a in PDF/A-1b. Nivo a (accesible) mora ustrezati vsem zahtevam standarda. Nivo b (basic) pa zahteva le, da se ohrani vizualni izgled dokumenta. Diplome, ki jih je potrebno oddati na sistemu STUDIS, morajo ustrezati nivoju standarda PDF/A-1b.

LaTeX in omenjeni format imata še nekaj težav s sobivanjem. Paket `pdfx.sty`, ki naj bi LaTeXu omogočal podporo formatu PDF/A ne deluje vedno v skladu s pričakovanji.

Zato raje priporočamo uporabo enega od mnogih spletnih mest, ki omogočajo konverzijo pdf datotek v obliko, ki je skladna s standardom PDF/A-1b, npr. `https://pdf.online/pdf-to-pdfa`, kjer je možno tudi testirati, ali je neka pdf datoteka skladna s tem standardom.

V predlogi so poleg izvornega dokumenta `diploma-FRI-vzorec.tex`, še vložena slika `galaksija.jpeg`, datoteka `literatura.bib` za uporabljeno literaturo ter ikone za licenco Creative Commons.

Poglavje 17

Sklepne ugotovitve

Uporaba LaTeXa in BibLaTeXa je v okviru Diplomskega seminarja **obvezna**! Izbira – LaTeX ali ne LaTeX – pri pisanju dejanske diplomske naloge pa je prepuščena dogovoru med diplomantom in njegovim mentorjem.

Res je, da so prvi koraki v LaTeXu težavni. Ta dokument naj služi kot začetna opora pri hoji. Pri kakršnihkoli nadaljnih vprašanjih ali napakah pa svetujemo uporabo Googla, saj je spletnih strani za pomoč pri odpravljanju težav pri uporabi LaTeXa ogromno.

Preden diplomo oddate na sistemu STUDIS, še enkrat preverite, če so slovenske besede, ki vsebujejo črke s strešicami, pravilno deljene in da ne segajo preko desnega roba. Poravnavo po vrsticah lahko kontrolirate tako, da izvorno datoteko enkrat testno prevedete z opcijo `draft`, kar vam pokaže predolge vrstice.

Članki v revijah

[32] Franc Solina. “15 seconds of fame”. V: *Leonardo* 37.2 (2004), str. 105–110. DOI: 10.1162/0024094041139274.

[35] Franc Solina in Blaž Meden. “Light fountain a virtually enhanced stone sculpture”. V: *Digital Creativity* 28.2 (2017), str. 89–102. DOI: 10.1080/14626268.2016.1258422.

Članki v zbornikih

[7] Peter Ciuha, Bojan Klemenc in Franc Solina. "Visualization of concurrent tones in music with colours". V: *Proceedings of the 18th ACM international conference on Multimedia*. ACM. Firenze, 2010, str. 1677–1680. DOI: 10.1145/1873951.1874320.

Poglavja v knjigah

[18] Donald E Knuth in Peter B Bendix. "Simple word problems in universal algebras". V: *Automation of Reasoning: Classical papers on computational logic 1957–1966*. Ur. Jörg H. Siekmann in Graham Wrightson. Springer, 1983, str. 342–376. DOI: 10.1007/978-3-642-81955-1_23.

[25] Peter Peer in Borut Batagelj. "Art—A Perfect Testbed for Computer Vision Related Research". V: *Recent Advances in Multimedia Signal Processing and Communications*. Springer, 2009, str. 611–629. DOI: 10.1007/978-3-642-02900-4_23.

Literatura

[1] *About GitHub*. URL: https://github.com/about (pridobljeno 2. 8. 2024).

[2] Michael Riis Andersen in sod. *Kinect depth sensor evaluation for computer vision applications*. 6. Department of Engineering, Aarhus University, 2012. URL: https://tidsskrift.dk/ece/article/view/21221 (pridobljeno 10. 5. 2021).

[3] Andreja Balon. *Vizualizacija*. Diplomska naloga. Fakulteta za elektrotehniko in računalništvo, Univerza v Ljubljani, 1990.

[4] *beautifulsoup4 · PyPI*. URL: https://pypi.org/project/beautifulsoup4/ (pridobljeno 2. 8. 2024).

[5] *BibLaTeX – Sophisticated Bibliographies in LaTeX*. URL: https://ctan.org/pkg/biblatex?lang=en (pridobljeno 11. 5. 2021).

[6] *Bibliography management with BibLaTex*. 1988. URL: https://www.overleaf.com/learn/latex/Bibliography_management_with_biblatex (pridobljeno 7. 5. 2021).

[7] Peter Ciuha, Bojan Klemenc in Franc Solina. "Visualization of concurrent tones in music with colours". V: *Proceedings of the 18th ACM international conference on Multimedia*. ACM. Firenze, 2010, str. 1677–1680. DOI: 10.1145/1873951.1874320.

[8] *Cloud Storage for Firebase*. URL: https://firebase.google.com/docs/storage (pridobljeno 2. 8. 2024).

[9] *Django overview — Django*. URL: https://www.djangoproject.com/start/overview/ (pridobljeno 2. 8. 2024).

[10] *Docker: Accelerated Container Application Development*. URL: https://www.docker.com/ (pridobljeno 2. 8. 2024).

[11] *Git*. URL: https://git-scm.com/ (pridobljeno 2. 8. 2024).

[12] *Home - Django REST framework*. URL: https://www.django-rest-framework.org/ (pridobljeno 2. 8. 2024).

[13] *Homepage - Reddit*. URL: https://www.redditinc.com/ (pridobljeno 1. 8. 2024).

[14] Slovensko društvo Informatika. *Islovar*. URL: http://www.islovar.org/islovar/islovar (pridobljeno 10. 6. 2021).

[15] *Introduction — djoser 2.2.2 documentation*. URL: https://djoser.readthedocs.io/en/latest/introduction.html (pridobljeno 3. 8. 2024).

[16] *Introduction · Get Started with Nuxt*. URL: https://nuxt.com/docs/getting-started/introduction (pridobljeno 2. 8. 2024).

[17] Salahaldin Juba, Achim Vannahme in Andrey Volkov. *Learning PostgreSQL*. Packt Publishing Ltd, 2015.

[18] Donald E Knuth in Peter B Bendix. "Simple word problems in universal algebras". V: *Automation of Reasoning: Classical papers on computational logic 1957–1966*. Ur. Jörg H. Siekmann in Graham Wrightson. Springer, 1983, str. 342–376. DOI: 10.1007/978-3-642-81955-1_23.

[19] Leslie Lamport. *LaTEX: A Document Preparation System.* Addison-Wesley, 1986.

[20] *Mapbox — Maps, Navigation, Search, and Data.* URL: https://www.mapbox.com/ (pridobljeno 2. 8. 2024).

[21] *MVC vs MVT Architectural Pattern. Like any other curious person, you must... — by Tejaswi Chaudhari — GDSC UMIT — Medium.* URL: https://medium.com/dsc-umit/mvc-vs-mvt-architectural-pattern-d306a56dce55 (pridobljeno 3. 8. 2024).

[22] Jarmo Niemelä. *How to create a PDF/A file with LATEX.* URL: https://webpages.tuni.fi/latex/pdfa-guide.pdf (pridobljeno 12. 5. 2021).

[23] *Our purpose.* URL: https://www.purpose.tripadvisor.com/ (pridobljeno 1. 8. 2024).

[24] *PDF/A.* 2005. URL: http://en.wikipedia.org/wiki/PDF/A (pridobljeno 5. 6. 2016).

[25] Peter Peer in Borut Batagelj. "Art—A Perfect Testbed for Computer Vision Related Research". V: *Recent Advances in Multimedia Signal Processing and Communications.* Springer, 2009, str. 611–629. DOI: 10.1007/978-3-642-02900-4_23.

[26] *PostgreSQL: The world's most advanced open source database.* URL: https://www.postgresql.org/ (pridobljeno 2. 8. 2024).

[27] *Pusher Channels Docs.* URL: https://pusher.com/docs/channels/ (pridobljeno 3. 8. 2024).

[28] ZRC SAZU. *Slovarji Inštituta za slovenski jezik Frana Ramovša ZRC SAZU.* URL: https://fran.si.

[29] *Search, Geocoding and Autofill Services — Mapbox.* URL: https://www.mapbox.com/search-service (pridobljeno 2. 8. 2024).

[30] *Server-Side Rendering (SSR) — Vue.js*. URL: https://vuejs.org/guide/scaling-up/ssr.html (pridobljeno 2. 8. 2024).

[31] *SLOVARJI.SI seznam slovenskih spletnih slovarjev*. URL: https://www.slovarji.si (pridobljeno 11. 5. 2021).

[32] Franc Solina. "15 seconds of fame". V: *Leonardo* 37.2 (2004), str. 105–110. DOI: 10.1162/0024094041139274.

[33] Franc Solina. *Light Fountain II – Galaxy*. 2015. URL: https://youtu.be/y6NAiXlNm20 (pridobljeno 9. 6. 2021).

[34] Franc Solina. *Skulpture/Sculptures 2012-2020, 2. izdaja / 2nd edition*. Ljubljana: Društvo likovnih umetnikov Ljubljana, Založba UL FRI, 2021. DOI: 10.51939/0001.

[35] Franc Solina in Blaž Meden. "Light fountain a virtually enhanced stone sculpture". V: *Digital Creativity* 28.2 (2017), str. 89–102. DOI: 10.1080/14626268.2016.1258422.

[36] *TypeScript: JavaScript With Syntax For Types.* URL: https://www.typescriptlang.org/ (pridobljeno 2. 8. 2024).

[37] *Visual Studio Code - Code Editing. Redefined*. URL: https://code.visualstudio.com/ (pridobljeno 2. 8. 2024).

[38] *Vue.js - The Progressive JavaScript Framework — Vue.js*. URL: https://vuejs.org/ (pridobljeno 2. 8. 2024).

[39] *Vuetify — A Vue Component Framework*. URL: https://vuetifyjs.com/en/ (pridobljeno 2. 8. 2024).

[40] Slovensko društvo za razpoznavanje vzorcev. *Slovar, Razpoznavanje vzorcev*. URL: https://slovar.vicos.si (pridobljeno 10. 6. 2021).

[41] *What are WebSockets? — Pusher*. URL: https://pusher.com/websockets/ (pridobljeno 3. 8. 2024).

[42] *What Is Containerization? — IBM*. URL: https://www.ibm.com/topics/containerization (pridobljeno 4. 8. 2024).

[43] Wikipedia contributors. *Donald Knuth — Wikipedia, The Free Encyclopedia*. 2021. URL: https://en.wikipedia.org/w/index.php?title=Donald_Knuth&oldid=1020717520 (pridobljeno 7. 5. 2021).

[44] *Woov – Supercharge your Events*. URL: https://woov.com/ (pridobljeno 1. 8. 2024).