Skrbnica projekta: dr. Tina Beranič

Gregor Pojbič, Blazhe Manev, Jan Lukač

Sistem za upravljanje konference

Dokumentacija projekta

Maribor, april 2023

Vsebina

[1. Vizija 1](#_Toc137583292)

[2. Povzetek projekta 4](#_Toc137583293)

[Kratek opis projekta 4](#_Toc137583294)

[Cilj aplikacije 4](#_Toc137583295)

[Povzetek funkcionalnosti 4](#_Toc137583296)

[Pregled končnih uporabnikov 4](#_Toc137583297)

[3. Organizacija 5](#_Toc137583298)

[Ekipa 5](#_Toc137583299)

[Komunikacija 5](#_Toc137583300)

[Razdelitev dela 5](#_Toc137583301)

[Beleženje napredka 5](#_Toc137583302)

[4. tehnologije 6](#_Toc137583303)

[Izbrane tehnologije za podatkovno bazo 6](#_Toc137583304)

[Izbrane tehnologije za backend 6](#_Toc137583305)

[Izbrane tehnologije za frontend 7](#_Toc137583306)

[5. Arhitektura 8](#_Toc137583307)

[Uporabniške zgodbe 8](#_Toc137583308)

[Diagram primerov uporabe 10](#_Toc137583309)

[Sheme za MongoDB 10](#_Toc137583310)

[6. Zagotavljanje kakovosti 12](#_Toc137583311)

[7. Zagotavljanje varnosti 14](#_Toc137583312)

[8. Vzpostavitev 15](#_Toc137583313)

[Lokalni zagon projekta 15](#_Toc137583314)

[Vzpostavitev gostovanja 15](#_Toc137583315)

[9. Izzivi 18](#_Toc137583316)

[10. posnetki zaslona 19](#_Toc137583317)

[11. repozitorij 23](#_Toc137583318)

[Povezava do našega repozitorija: 23](#_Toc137583319)

[Povezava do naše gostujoče spletne strain: 23](#_Toc137583320)

[Podatki o uporabnikih: 23](#_Toc137583321)

# Vizija

**Učna enota Projekt**

ITK / IPT UNI 3. Letnik

**Ekipa, ki predlaga projekt**

Ekipa praviloma šteje 3 člane/ice, izjemoma 2 ali 4.

|  |  |  |
| --- | --- | --- |
|  | **Naziv ekipe:** Konfi | |
| **Član/ica** | **Ime in priimek** | **E-poštni naslov** |
| 1. | Jan Lukač | jan.lukac@student.um.si |
| 2. | Blazhe Manev | manev.blazhe@student.um.si |
| 3. | Gregor Pojbič | gregor.pojbic@student.um.si |
| 4. |  |  |

**Projekt bo** (označite)**:**

* rešitev na osnovi predlogov koordinatorjev (št. predloga: 1 )

**Vizija izdelka / storitve:**

|  |
| --- |
| Organizacija konference terja od vpletenih veliko truda, zato želimo smiselno določena opravila avtomatizirati s pomočjo spletne in/ali mobilne platforme. |

**Podatki o izdelku / storitve**

|  |  |
| --- | --- |
| **Ključni uporabniki** | Ključni uporabniki rešitve so udeleženci konferenc in organizatorji konferenc. |
| **Glavne funkcionalnosti**  (omejite se na funkcionalne zahteve, opremite s prioritetami) | - vodenje prijav udeležencev  - spremljanje kontaktov preteklih konferenc  - spremljanje prispevkov (člankov, predstavitev) na konferenci; oblikovanje programa  - obveščanje udeležencev o novostih in morebitnih spremembah  - omogočanje sodelovanja udeležencev na predstavitvah |
| **Glavne omejitve**  (npr. varnostne, performančne, omejitve okolja ipd.) | Ker bomo beležili določene podatke uporabnikov, moramo biti pazljivi na način shranjevanja in da so ti podatki varni. Omogočiti moramo tudi, da bo lahko rešitev uporabljalo vsaj toliko ljudi, kot se jih želi udeležiti konference(načeloma še več). |
| **Meje izdelka / storitve**  (česar NE boste vključili) | Video konference, nalaganje videoposnetkov konference, |
| **Uporabniški vmesnik(i)**  (npr. mobilni, spletni, namizni, API, konzolni ipd.) | Spletni vmesnik |
| **Zahteve rešitve / storitve**  (npr. posebna strojna ali programska oprema, veljavna naročnina na storitve ipd. ) | Visual Studio Code, CompassDB, Postman, GitHub/GitLab, Microsoft Word |
| **Alternativne rešitve / storitve na trgu** | https://www.cvent.com/  <https://whova.com/>  https://www.engineerica.com/conferencetracker/ |
| **Konkurenčna prednost**  Zakaj bo predlagan izdelek /storitev za uporabnika boljši od obstoječih? | Sistem omogoča avtomatizacijo številnih opravil, kar zmanjšuje čas in napor potreben za organizacijo konference. Poleg tega, sistem zagotavlja enostavno vodenje prijav udeležencev, spremljanje stikov s preteklimi udeleženci, oblikovanje programa in spremljanje prispevkov na konferenci.  Engineerica je plačljiva konkurenčna rešitev. |
| **Planirane integracije**  (npr. sistemi za overjanje, sistemi za shranjevanje vsebin na oblaku, napovednimi modeli v oblaku, zunanjimi viri podatkov ipd.) | Sistem za overjanje uporabnikov – Firebase, sistemi za shranjevanje vsebin na oblaku - Atlas |
| **Zagotavljanje trajnosti podatkom**  (npr. relacijska baza podatkov, NoSql baza, blockchain platforma, PB v oblaku, trajnosti podatkov ne bo ipd.) | MongoDB |
| **Namestitev zaledja izdelka / storitve**  (npr. oblak - zabojniki, oblak - serverless, lasten strežnik, zalednega sistema ne bo.) | Heroku (v primeru da je zastonj) |
| **Plan organizacijskega vidik razvoja**  (uporabljena razvojna metoda – npr. Scrum, Kanban…; Uporabljeni standardi dokumentiranja – npr. UML ipd...) | Scrum (eno-tedenski sprinti)  UML(DPU) |

# Povzetek projekta

## Kratek opis projekta

Sistem za vodenje konferenc je spletna aplikacija, ki je namenjena predvsem organizatorjem konferenc, kot središčni sistem, preko katerega lahko na hiter in preprost način organizirajo in na nek način vzpostavijo konferenco. Posledično je namenjena tudi udeležencem same konference, saj lahko preko te sledijo poteku konference in pridobijo vse novosti in obvestila o samem dogajanju na konferenci.

## Cilj aplikacije

Cilj naše aplikacije je pomagati organizirati konferenco na lažji način, preko enotnega vmesnika. Cilj aplikacije je tudi izboljšati samo preglednost oseb ki se nameravajo udeležiti konference, ter udeležencev.

## Povzetek funkcionalnosti

* Ustvarjanje konference
* Vnos prostora konference
* Pošiljanje vabil za udeležbo
* Pošiljanje obvestil prijavljenim uporabnikom
* Vnos gradiva
* Ustvarjanje urnika
* Seznam prijavljenih uporabnikov
* Seznam udeležencev konference
* Ustvarjanje QR-kode
* Pregled podatkov konference
* Branje QR-kode
* Prenos objavljenega gradiva

## Pregled končnih uporabnikov

Sistem za upravljanje konference bodo uporabljali organizatorji konferenc in pa udeleženci konferenc. Torej se starost teh oseb predvidoma giblje med 20 in 60 leti. Predvidoma so to osebe, ki so že diplomirale, magistrirale, doktorirale, ali pa še študirajo.

# Organizacija

## Ekipa

V ekipi smo Gregor Pojbič, Blazhe Manev in Jan Lukač. Opredelili smo se glede na naše izkušnje, tako smo dosegli, da Gregor in Blazhe skrbita za backend in povezavo s frontend, Jan pa za frontend in izgled. Seveda je vsaka oseba še vedno lahko poskrbela za oba dela.

## Komunikacija

Komunikacija v skupini, je potekala preko Discord skupine. Predvsem je komunikacija potekla preko sporočil in pošiljanja slik, okvirno dvakrat na teden pa smo planirali tudi sestanke, kjer smo imeli pregled napredka znotraj skupine. Če je slučajno prišlo do morebitne napake, smo tudi to reševali s pomočjo video klica in delitve zaslona.

Komunikacija s skrbnico dr. Tino Beranič je potekala preko orodja MS Teams, kjer smo ustvarili svojo skupino, za namene komuniciranja. Konzultacije so potekale v živo, kjer smo pokazali napredek in se pogovorili o nadaljnjem predvidenem delu.

## Razdelitev dela

Predvidene smo imeli enotedenske sprinte. Pred začetkom vsakega sprinta smo imeli sestanek in se dogovorili kaj je potrebno narediti v tistem tednu/sprintu. Torej smo določili naloge, ki smo si jih razdelili tako da smo se med sabo odločili kdo bo kaj delal in če je že kdo vešč v tej tehnologiji, ki smo jo želeli implementirati. V primeru, da je oseba vse dodeljene naloge končala predčasno, si je lahko izbrala naloge mišljene za naslednji sprint ali pa je prevzel naloge od druge osebe.

## Beleženje napredka

Napredek smo beležili s pomočjo GitLab, kjer smo napredek samih nalog spremljali preko kanban table, ki smo jo ustvarili na GitLab. Vsaka naloga je bila dodeljena v svoj ustrezen sprint in tudi osebo, ki je odgovorna za to nalogo. Ob koncu vsakega sprinta, smo vse končane naloge tudi zaprli.

# tehnologije

## Izbrane tehnologije za podatkovno bazo

Za podatkovno bazo smo se odločili, da uporabimo MongoDB. To odločitev smo sprejeli kot ekipa na podlagi tega, da smo v preteklih letih tudi sodelovali skupaj kot skupina in tudi takrat smo uporabili MongoDB, torej smo mnenja, da smo v tem bolj vešči, kot pa v recimo SQL bazi, ali kakšni drugi rešitvi za shranjevanje podatkov. Ker smo v preteklosti že tudi vsi delali z MongoDB, smo že približno vedeli kako vzpostaviti celotno bazo, saj smo to že naredili. Prav tako se nam zdi preprosta povezava z MongoDB in tudi samo gostovanje baze. MongoDB smo izbrali tudi, ker lahko lažje delamo samo s json objekti.

MongoDB gostujemo s pomočjo MongoDB Atlas. Ker smo v preteklosti uporabili že tudi to tehnologijo vsakokrat, ko smo se posluževali MongoDB, smo se tudi tokrat odločili da gostujemo bazo s pomočjo MongoDB Atlas, saj je ta zanesljiva, preprosto se jo vzpostavi in za naše namene jo lahko uporabljamo zastonj.

Za avtentikacijo in hranjenje gesel uporabnikov smo se odločili za Firebase.

## Izbrane tehnologije za backend

Za avtentikacijo uporabnikov smo uporabili Firebase. V preteklosti smo že uporabili JWT avtentikacijo, vendar pa smo se za Firebase odločili zato, ker smo želeli uporabiti še prijavo s pomočjo Google računa, ki se nam je zdela lažja s pomočjo Firebase. Med drugim smo se za Firebase odločili tudi zato, ker jo več oseb priporoča.

Za sam backend pa smo se odločili, da uporabimo nodejs, saj smo s to tehnologijo seznanjeni že 3 leta vsi člani ekipe in smo skupaj že več krat razvijali s to tehnologijo. Z enakim razlogom smo se posledično odločili tudi za knjižnico express.

Za pošiljanje email sporočil smo uporabili knjižnico nodemailer, saj je najlažja rešitev za pošiljanje email sporočil iz same nodejs aplikacije.

Za nalaganje datotek, smo uporabili knjižnico multer. To smo izbrali, ker je med pogostejšimi knjižnicami za nalaganje datotek, ker je za njo napisane veliko dokumentacije in ker je precej preprosta za uporabo.

## Izbrane tehnologije za frontend

Za frontend smo se odločili uporabiti React Typescript. Ker smo vsi člani ekipe že delali z React, smo se odločili, da bomo uporabili že poznano tehnologijo. Diskusija je potekala tudi ali izbrati navaden React ali pa React Typescript, vendar smo zaradi robustnosti in sprotnega preverjanja tipiziranih napak izbrali React Typescript.

Za izgled smo uporabili knjižnico Material UI. Za to smo se odločili, ker se nam zdi, da je Bootstrap že malo zastarel in nismo želeli imeti »bootstrap izgled«. Kljub temu da tega še nismo uporabili pred samim začetkom tega projekta, smo bili pripravljeni vložiti dodaten trud v to, da smo se naučili uporabo te knjižnice za izgled.

Za skeniranje qr kode smo uporabili knjižnico @yudiel/react-qr-scanner. Za to knjižnico smo se odločili zaradi preprostosti uporabe.

Za generiranje QR kode, pa smo uporabili knjižnico qrcode.react.

# Arhitektura

## Uporabniške zgodbe

Uporabniške zgodbe, smo spremenili in dodelali. Najprej so naše uporabniške zgodbe bile:

1. Kot organizator konference, želim vnesti prostore v katerih se bo konferenca odvijala in specifikacije konference, da si bodoči udeleženci lahko ogledajo vse željene informacije in da mi ni potrebno obveščati vsakega posebej.

2. Kot organizator konference, želim dodati udeležence konferenci, s tem bo se vsak željen udeleženec lahko registriral in preko same registracije tudi obvestil, da se namerava konference udeležiti.

3.1 Kot organizator konference, želim videti možno invalidnost udeležencev, da lahko prostor prilagodim njihovim potrebam.

3.2 Kot bodoči udeleženec konference, želim imeti pregled nad dostopnostjo samega konferenčnega prostora, da vidim, ali bom imel dostop do prostora ali pa se konference ne bom udeležil.

4. Kot organizator konference, želim beležiti dejansko fizično prisotnost udeležencev, da bom na koncu imel pregled, koliko oseb se je dejansko udeležilo same konference.

5. Kot udeleženec konference, želim imeti možnost slediti vsebinam, ki bodo na konferenci predstavljene, da se bom lahko pripravil na predavanja in imel boljši pregled nad celotno konferenco.

6. Kot udeleženec konference, želim imeti možnost uporabljati live chat, da se bom lahko med konferenco povezal s predavatelji in drugimi udeleženci ter postavil vprašanja ali delil mnenja.

7. Kot udeleženec konference, želim biti obveščen o novostih in spremembah, ki se nanašajo na konferenco, da bom vedno na tekočem s pomembnimi informacijami in ne bom zamudil pomembnih dogodkov. Želim prejemati obvestila preko maila, ko organizatorji dodajo nove udeležence ali ko pride do sprememb v programu konference.

Po konzultaciji s skrbnico projekta, smo izpopolnili uporabniške zgodbe in tako smo prešli na naslednje in tudi končne uporabniške zgodbe:

1. Kot organizator konference, želim vnesti prostore v katerih se bo konferenca odvijala in specifikacije konference, da si bodoči udeleženci lahko ogledajo vse željene informacije in da mi ni potrebno obveščati vsakega posebej.

2. Kot organizator konference, želim dodati udeležence konferenci, s tem bo se vsak željen udeleženec lahko registriral in preko same registracije tudi obvestil, da se namerava konference udeležiti.

3. Kot organizator konference, želim videti možno invalidnost udeležencev, da lahko prostor prilagodim njihovim potrebam.

4. Kot organizator konference, želim beležiti dejansko fizično prisotnost udeležencev, da bom na koncu imel pregled, koliko oseb se je dejansko udeležilo same konference.

5. Kot udeleženec konference, želim imeti možnost slediti vsebinam, ki bodo na konferenci predstavljene, da se bom lahko pripravil na predavanja in imel boljši pregled nad celotno konferenco.

6. Kot udeleženec konference, želim biti obveščen o novostih in spremembah, ki se nanašajo na konferenco, da bom vedno na tekočem s pomembnimi informacijami in ne bom zamudil pomembnih dogodkov. Želim prejemati obvestila preko maila, ko organizatorji dodajo nove udeležence ali ko pride do sprememb v programu konference.

## Slika, ki vsebuje besede besedilo, diagram, vrstica, posnetek zaslona Opis je samodejno ustvarjenDiagram primerov uporabe

## Sheme za MongoDB

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, zaslon, programska oprema

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Za uporabo v MongoDB smo ustvarili 5 shem. Tako smo ustvarili shemo za gradivo, shemo za konferenco, shemo za organizatorja, shemo za prostor in shemo za udeleženca.

# Zagotavljanje kakovosti

Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema, računalniška ikona

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, diagram, posnetek zaslona, vrstica

Opis je samodejno ustvarjenZa zagotavljanje in spremljanje kakovosti smo uporabili [SonarCloud](https://sonarcloud.io/project/overview?id=PojbicGregor_sistem-za-vodenje-strokovne-konference). S pomočjo SonarCloud smo lahko sproti spremljali napake v naši kodi in jih posledično tudi odpravili. Za uporabo SonarCloud smo se odločili na podlagi priporočil skrbnice projekta. Odločili smo se da odpravimo vse napake, razen Code Smells, predvsem pa smo se osredotočili na Vulnerabilities.

Sliki prikazujeta grafe števila težav skozi čas in pa število težav na koncu. Razvidno je, da smo odpravili večino težav in da smo se na te osredotočili bolj proti koncu samega razvoja.

Slika, ki vsebuje besede besedilo, pisava, posnetek zaslona, številka

Opis je samodejno ustvarjenDodatno smo za kontrolo kvalitete uporabili tudi React teste, s katerimi smo testirali ključne komponente, oziroma pravilno delovanje teh ključnih komponent. Tako smo testirali 10 različnih komponent. Skupaj je bilo ustvarjenih 44 različnih testov, ki so testirali 10 različnih komponent.

Slika prikazuje zagon vseh testov in nam prikaže, da so se vsi testi pravilno izvedli, ter da ni prišlo do nobene napake.

# Zagotavljanje varnosti

En način kako zagotovimo varnost, je da sami ne hranimo gesel, ampak le token, ki ga ustvari Firebase.

Drugi način kako zagotovimo varnost shranjenih podatkov je, da omogočamo dostop do teh podatkov le določenim povezavam. Torej smo ustvarili whitelist povezav, ki lahko dostopajo do našega backenda.

Za določene operacije v povezavi z bazo, mora biti uporabnik prijavljen v samo spletno stran, da lahko dostopa do teh operacij preko samega uporabniškega vmesnika. Če uporabnik ali pa organizator ni prijavljen, potem teh operacij ne more izvajati, saj preverjamo, če se pošilja token prijavljene osebe na sam backend.

Ker spletno stran gostujemo na spletnem ponudniku za gostovanje Render, imamo omogočen tudi https, kar pomeni, da so podatki med samim prenosom in pošiljanjem šifrirani, posledično so podatki med samim prenosom in pošiljanjem varni.

Pri vnosu datotek smo se omejili na velikost 20MB, torej organizator (le ta lahko vnaša datoteke) lahko naloži datoteko največ velikost 20MB.

Seveda smo za varnost uporabili tudi SonarCloud, kjer smo posledično razrešili vse težave v zavihku Vulnerabilities in Security Hotspots.

V sami kodi ne beležimo nobenih gesel potrebnih za dostop do samih orodij. Vse kritične podatke za dostop, kot so gesla ali privatni ključi hranimo kot enviroment variables v datoteki .env, ki zaradi varnosti ni na Git.

# Vzpostavitev

## Lokalni zagon projekta

Za lokalni zagon projekta, si je potrebno projekt najprej namestiti na računalnik, kar se lahko naredi s kloniranjem projekta preko Git.

Ko je projekt kloniran na naš lokalen računalnik, se premestimo v koren projekta. Iz tam se premaknemo v mapo frontend in mapo backend s pomočjo Windows Powershell. Najprej naložimo vse knjižnice in generiramo node\_modules z ukazom npm i. Nato zaženemo backend s pomočjo ukaza *nodemon,* frontend pa zaženemo s pomočjo ukaza npm start.

Potrebujemo še .env datoteki za frontend in backend. Eno .env datoteko premaknemo v mapo frontend drugo .env datoteko pa premaknemo v mapo backend.

S tem bi spletna aplikacija morala delovati brez težav.

Kot skupina smo uporabljali lokalno vzpostavitev za razvoj same aplikacije.

## Vzpostavitev gostovanja

Za gostovanje smo se odločili uporabiti Render, saj je brezplačen in zelo preprost za vzpostaviti samo aplikacijo. Zraven tega pa ponuja še CD (continous delivery), česar pa smo se tudi posluževali.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, električno modra

Opis je samodejno ustvarjenZa vzpostavitev frontenda, je potrebno izbrati Static Site.

Nato izberemo repozitorij, katerega uporabimo za gostovanje. Ker je naš repozitorij sestavljen iz frontend in backend map, moramo korenski direktorij postaviti frontend mapo. Ker je potrebno še generirati sam frontend, ker ga med samim razvojem nismo, je potrebno vnesti ukaz *npm run build.* Za publish directory pa potem izberemo mapo build, ki se bo ustvarila s prejšnjim ukazom.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, številka

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, pisava, vrstica, posnetek zaslona

Opis je samodejno ustvarjenS tem smo vzpostavili frontend naše aplikacije in s tem pridobili tudi CD (continous delivery). Za popolnoma delujoč frontend, je potrebno še dodati enviroment variables.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, električno modra

Opis je samodejno ustvarjenZa vzpostavitev backenda je potrebno izbrati Web Service.

Nato izberemo repozitorij, katerega uporabimo za gostovanje. Ker je naš repozitorij sestavljen iz frontend in backend map, moramo korenski direktorij postaviti backend mapo. Za zagon backend izberemo ukaz npm start*.*

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, številka

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, pisava, posnetek zaslona

Opis je samodejno ustvarjen

S tem da se poslužujemo tudi Auto-Deploy, zagotovimo tudi CD (continous delivery) na samem backendu.

# Izzivi

Med samim razvojem smo imeli le en manjši izziv s komunikacijo, saj smo napačno dojeli funkcionalnosti, zato smo tudi naredili uporabniške zgodbe dvakrat. Torej smo prvič napisali nekaj napačnih uporabniških zgodb, katere smo v drugo izbrisali ali pa zamenjali.

Večji izziv je bil z vzpostavitvijo SonarCloud, saj smo najprej za projekt na GitLab-u naredili skupino v kateri smo naredili repozitorij. SonarCloud pa ne dopusti brezplačne uporabe če se ga želi uporabljati na repozitoriju za skupino, zato smo bili primorani ustvariti nov repozitorij, ki ni narejen na skupini ampak na enem od uporabnikov (torej članov naše skupine). Ko smo to uredili, je vzpostavitev potem bila mogoča brezplačno.

# Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema, večpredstavnostna programska oprema Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, večpredstavnostna programska oprema, grafična programska oprema Opis je samodejno ustvarjenposnetki zaslona

Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema, računalniška ikona

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema, računalniška ikona

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, oblikovanje

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, oblikovanje

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema, računalniška ikona

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, oblikovanje

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, pisava, plakat

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, pisava, oblikovanje

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, oblikovanje

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, posnetek zaslona, oblikovanje

Opis je samodejno ustvarjen

# repozitorij

## Povezava do naše gostujoče spletne strain:

<https://vodenje-konference-frontend.onrender.com>

## Podatki o uporabnikih:

Navaden Uporabnik:

Email: [123@yahoo.com](mailto:123@yahoo.com)

Geslo: 123456

Organizator:

Email: [jan@yahoo.com](mailto:jan@yahoo.com)

Geslo: 123456

Opomba! Backend je v stanju spanja ko ni v uporabi, zato prvoten zagon backenda mogoče traja 20 sekund, ko se Uporabnik želi prijaviti.