Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.3
DOI: https://doi.org/10.24867/22BE14Vujicic

RAZVOJ WEB APLIKACIJE „Sistem za prihvatanje rezultata ispita i statistika predmeta“

DEVELOPMENT OF WEB APPLICATION „System for accepting exam results and course statistics“

Milena Vujičić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je prikazan razvoj web aplikacije – Sistem za prihvatanje rezultata ispita i statistiku predmeta Programski jezici i strukture podataka. Pri razvoju aplikacije korišćeni su Django i Angular framework, Google Drive API, kao i Docker alat za kontejnerizaciju. Aplikacija obavlja konverziju podataka iz najčešće korišćenih formata za beleženje rezultata ispita u format Google Sheet tabele i omogućava da svi korisnici imaju na raspolaganju sve podatke o rezultatima testova i ispita na jednim mestu, u istom formatu, podesnom za statističku obradu.*

Ključne reči: *web aplikacija, rezultati ispita, statistička obrada podataka*

Abstract – *This paper presents development of a web application - System for accepting exam results and statistics of the subject Programming Languages and Data Structures. Django and Angular framework, Google Drive API, and Docker tool for containerization were used in the development of the application. The application converts data from the most commonly used formats for recording results into a Google Sheet table format and allows all users to have available all data on test and exam results in one place, in the same format, suitable for statistical processing.*

Keywords: *web application, exam results, statistical data processing*

1. UVOD

Zadatak rada je izrada web aplikacije koja olakšava unos i obradu rezultata ispita iz predmeta Programski jezici i strukture podataka, na Fakultetu tehničkih nauka u Novom Sadu. Rezultati polaganja preispitnih obaveza i ispita unose se u različitim formatim i ručno se prenose u tabelu dostupnu svim učesnicima u postupku. Teorijski deo ispita se automatski pregleda i rezultati se dostavljaju dežurnom nastavniku u podrazumevanom formatu korišćenog programa.
Aplikacija koja je predmet rada omogućava konverziju iz najčešće korišćenih formata za beleženje rezultata u format Google Sheet tabele. Time će biti omogućeno da svi korisnici imaju sve podatke raspoložive na jednom mestu, u istom formatu, podesnom za statističku obradu.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Srđan Popov, red. prof.

Praćenje promena, implementirano u aplikaciji, jeste poboljšanje koje predmetnom profesoru daje mogućnost da proveri da li su svi uključeni u izvođenje nastave, pregled radova i evidentiranje rezultata ispita uneli potebne informacije, koje se prosleđuju Studentskoj službi. Takođe, jedinstven format i postojanje centralne baze podataka omogućava primenu statističkih metoda obrade podataka.

Aplikacija je razvijena korišćenjem Django [1] i Angular [7] framework-a, Google Drive API-ja, kao i Docker [4] alata za kontejnerizaciju rešenja.

2. OPIS Aplikacije

Serverska aplikacija kreirana je korišćenjem Django framework-a, biblioteke djangorestframework i Google Drive API. Kontejnerizacija aplikacije je obavljena u Docker-u.

Klijentska aplikacija implementirana je upotrebom Angular framework-a. Ona služi za posrednu interakciju korisnika sa centralnim Google Sheet-om. U klijentskoj aplikaciji implementirane su funkcije za dodavanje korisnika, unos i upload podataka na server, kao i prikaz logova i statistike.

2.1. Serverska aplikacija

Model podataka

U programu postoje sledeći modeli:
AppUser - modeluje korisnika aplikacije. Nasleđuje klasu django.contrib.auth.models.AbstractBaseUser. Ova klasa sadrži prototip za kreiranje korisnika aplikacije,
LogEntry - modeluje podatke unete ili izmenjene od strane korisnika
FileUpload - modeluje fajl koji je upload-ovan na server.

Klase AppUser i LogEntry čuvaju se trajno u bazi podataka, dok klasa FileUpload služi za mapiranje i privremeno čuvanje putanje fajla na serveru
Za korisnika aplikacije kreirana je i AppUserManager klasa. Klasa nasleđuje ugrađenu klasu django.contrib.auth.models.BaseUserManager.
AppUserManager klasa definiše način na koji će se kreirati korisnik i superuser aplikacije. Funkcijom django.contrib.hashers.make_password() omogućeno je čuvanje heširane lozinke.

Serijalizatori podataka

S obzirom da se podaci potrebni za kreiranje instance klase FileUpload ne prosleđuju putem REST zahteva, za ovu klasu nije bilo potrebno kreirati serijalizatore.
Klasa AppUser ima implementirane tri vrste serijalizatora: UserSerializer, EmailSerializer, SuperUserSerializer. UserSerializer se koristi za serijalizaciju podataka vezanih za običnog korisnika. EmailSerializer serializuje samo email i id korisnika. SuperUserSerializer ima mogućnost da serializuje sva polja vezana za superuser-a aplikacije.
Klasa LogEntry ima implementirana dva serijalizatora: LogEntrySerializer i TableSerializer. Razlika između ova dva serijalizatora je samo u poljima user i date. LogEntrySerializer serijalizuje podatke vezane za podatke o izmenama. TableSerializer se koristi za statistiku izmena koju vidi korisnik, pa podaci o datumu izmene i korisniku nisu potrebni.

Rad sa korisnicima programa

U aplikaciji postoje dve vrste korisnika: običan korisnik i superuser. Superuser ima mogućnost da vidi podatke koji se čuvaju u log-u, kao i da dodaje nove korisnike aplikacije. Prilikom pokretanja aplikacije prvi put, potrebno je ručno dodati superuser-a aplikacije. Funkcija user_count() proverava broj korisnika aplikacije. Operacije nad korisnicima, koje nisu dodavanje prvog superusera, nisu moguće, ako u aplikaciji ne postoji ni jedan korisnik. Broj korisnika aplikacije se pamti u request sesiji.
Sa serverske strane prijavljivanje na aplikaciju se vrši preko funkcije user_login(). Nakon slanja zahteva proveravaju se email i password korisnika. Funkcija django.contrib.auth.hashers.check_password() proverava ispravnost heša unete lozinke.

Rad sa unetim podacima

Rad sa unetim podacima se sastoji iz tri dela: preuzimanje i parsiranje podataka poslatih u zahtevu, obrada unetih podataka i postavljanje podataka na Google Sheet putem Google Drive API-ja.

Rad sa csv i zip fajlovima

Korisnik, u zavisnosti od vrste testa čije rezultate unosi, na server šalje ili .csv ili .zip datoteku. Datoteke se šalju u string base64 formatu. Nakon što se datoteka prevede iz string u binarni format, privremeno se čuva kao statički fajl, a njena putanja u bazi podataka. Nakon što je završena obrada datoteke, ona se briše iz baze podataka i sa servera. Pandas dataframe podržava direktno učitavanje .csv datoteka funkcijom pandas.read_csv(). Struktura svakog .zip fajla je takva da u njoj postoji jedna .csv datoteka. Pošto pandas ne podržava direkno čitanje .zip datoteke, potrebno je prvo ekstrahovati .csv datoteku. Za ekstrakciju .csv datoteke korišćena je biblioteka ZipFile. Ona poseduje funkcije za čitanje, pregled strukture i ekstrahovanje .zip fajlova.
Funkcija parse_file_data, koja pravi fajl i stavlja ga na server.

Obrada podataka

Obrada unetih podataka obavlja se putem pandas biblioteke. Nakon što se od unete datoteke napravi DataFrame nad njim se vrši niz neophodnih operacija. Tabela koja se čuva na centralnom Google Sheet dokumentu ima različit način imenovanja kolona. Zbog toga je potrebno prvo formatirati imena kolona i ukloniti nepotrebne kolone iz DataFrame-a. Format broja indeksa, koji se koristi kao ključ po kome se podaci upisuju u tabelu, je drugačiji na centralnom Google Sheet-u i u .csv fajlu. Parsiranje i reformatiranje broja indeksa izvršeno je putem regex izraza. Regex izraz očitava smer, broj upisa i godinu iz indeksa i čuva te podatke za dalju upotrebu. Na osnovu godine upisa određuje se na koji list u centralnom Google Sheet-u je potrebno upisati podatke.
Program u svom settings.py fajlu sadrži promenljivu CURRENT_YEAR koja čuva podatak o tekućoj školskoj godini. Studenti upisani pre ove školske godine se razrstavaju na poseban list. Ostali studenti se razvrstavaju na osnovu smera na koji su upisani. Nakon očitavanja podataka iz centralnog Google Sheet-a i njihovog prevođenja u DataFrame, uneti podaci se upoređuju sa njima. Funkcija pandas.merge() i pozivanje agregatora .groupby() vrši spajanje podataka dva DataFrame-a na osnovu broja indeksa. Ovakvim spajanjem se ažuriraju podaci u odgovarajućem redu tabele.

Rad sa Google Sheet tabelama

Za rad sa Google Sheet [2] tabelama potrebno je koristiti Google Drive API. google_auth_oauthlib.flow biblioteka sadrži funkcije i klase za autorizaciju korisnika. Klasa InstalledAppFlow iz ove biblioteke implementira funkciju from_client_secrets_file uzima putanju do ključa kojim se vrši autentifikacija korisnika. Putanja do ovog fajla čita se iz environment variable. Nakon što je autentifikacija uspešno izvšena moguće je vršiti operacije nad Google Sheet tabelom. Funkcije open_sheet_for_reading i open_sheet_for_writing implementirane su tako da omoguće čitanje tabele prema unetom id-ju. Upis i čitanje tabele se vrši na osnovu imena listova u tabelama.

Log i statistika

Log korišćenja aplikacije se čuva nakon upisa podataka na centralni Google Sheet. Log sadrži sve unete izmene. Pošto je potrebno kreirati više od jedne instance klase LogEntry, koristi se funkcija bulk_create koju implementiraju objekti modela.
Statistika koju korisnik vidi su svi podaci koji postoje u centralnom Google Sheet-u. Na klijentski deo aplikacije potrebno je poslati sve podatke iz centralnog Google Sheet-a. Funkcija get_statistics se svodi na čitanje podataka iz Google Sheet tabele.

2.2. Klijentska aplikacija

Model podataka

Model podataka implementiran je putem interfejsa AppUser, Log, Count i ResultsFormat. Osim interfejsa

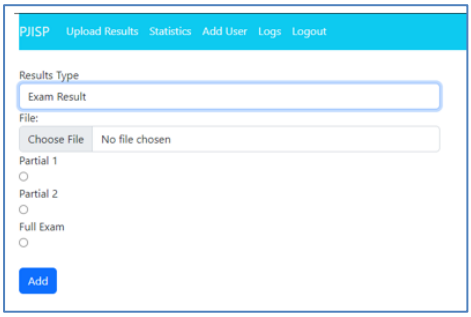
Count, model se poklapa sa modelom na serverskoj strani. Interfejs Count služi za proveru trenutnog broja korisnika u aplikaciji.

Rad sa korisnicima

Da bi bio omogućen rad sa ostalim funkcionalnostima aplikacije, tokom prvog pokretanja, potrebno je dodati superuser-a.
Unos podataka za kreiranje superuser-a vrši se putem forme. Forma je mapirana na model korisnika. U TypeScript [6] fajlu komponente definisan je način slanja REST zahteva na serverski deo aplikacije. REST zahtev se šalje putem klase HttpClient. Ova klasa stvara publish subscribe relaciju za REST zahtev.
Superuser aplikacije ima mogućnost da dodaje druge korisnike aplikacije. Forma slična onoj za dodavanje superuser-a postoji i za dodavanje drugih korisnika aplikacije. Tokom dodavanja korisnika traži se autentifikacija superuser-a koji dodaje novog korisnika.

Unos podataka

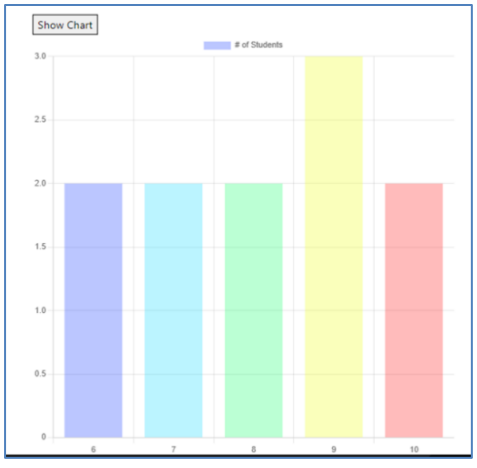
Forma za unos podataka sadrži polje i radio button-e za izbor tipa testa za koji se unose rezultati i polje za upload fajla. Polje za upload fajla učitava fajl sa uređaja korisnika i prosleđuje ga u tekstualnom formatu na server, sl. 1.



Slika 1. Forma za unos podataka

Prikaz statistike i log-a

Statistički proračuni obavljaju se na osnovu podataka iz centralnog Google Sheet-a. Korisnik ima mogućnost da vidi dve vrste podataka, sve podatke o studentima, zajedno sa ocenama ili statistiku na nivou predmeta (prolaznost, prosečna ocena, histogram raspodele ocena), slika 2.

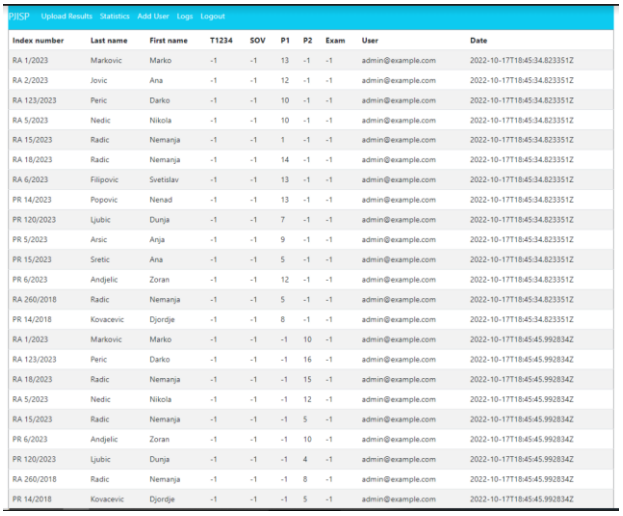


Slika 2 – Histogram

Na klijentskoj strani aplikacije implementirane su funkcije za računanje prolaznosti i prosečne ocene.
Tabela koja prikazuje podatke o studentima ima mogućnost filtriranja podataka po broju indeksa, imenu ili prezimenu.
Crtanje histograma raspodele ocena urađeno je sa Chart.js bibliotekom. Ova biblioteka nudi različite funkcije za grafičko predstavljanje podataka korišćenjem canvas-a.

Log podaci

Log podatke može da vidi samo superuser aplikacije. Podaci koji se dobiju sa serverske strane aplikacije se smeštaju u tabelu Change log, slika 3.



Slika 3 – Change log

2.3 Kontejnerizacija rešenja

Serverska i klijentska aplikacija su kontejnerizovane uz pomoć Docker i Docker compose alata. U CMD direktivi Dockerfile-a serverske aplikacije poziva se skript fajl koji vrši pokretanje produkcionog servera (Gunicorn [8]) i migraciju modela u bazu podataka.
Pre nego što je kontejnerizovana klijentska aplikacija, pozvana je komanda ng build, Angular framework-a. Ova komanda vrši prevođenje svih fajlova Angular aplikacije u HTML i JavaScript format. Produkcioni server koji se koristi za serviranje statičkih fajlova je NGINX [5]. U konfiguracionom fajlu navedena je putanja do početne stranice. Takođe, navedena je i direktiva za rutiranje adresa.
Docker compose fajl objedinjuje serversku i klijentsku aplikaciju, kao i bazu podataka i interfejs za upit podataka. Svi osetljivi podaci iz aplikacija stavljeni su u docker_vars.env fajl. Nalaze se u formatu ključ=vrednost.

3. ZAKLJUČAK

Korišćenjem Django i Angular framework-a, Google Drive API-ja, kao i Docker alata za kontejnerizaciju rešenja ostvaren je cilj rada: konverzija podataka iz najčešće korišćenih formata za beleženje rezultata u format Google Sheet tabele i omogućeno je da svi korisnici imaju na raspolaganju sve podatke o ispitima i testovima na jednom mestu, u istom formatu, podesnom

za statističku obradu. Kreiranom Web aplikacijom poboljšan je postupak unosa i obrade rezultata ispita iz predmeta Programski jezici i strukture podataka, čime se ostvaruje ušteda vremena utrošenog na evidentiranje podataka, smanjuje mogućnost pogrešnog unosa i omogućava analiza ostvarenih rezultata.

4. LITERATURA

[1] Django - https://docs.djangoproject.com
[2] Google Sheets - https://developers.google.com/sheets/api/quick start/python
[3] Django REST biblioteka - https://www.django-rest-framework.org/api-guide/filtering/
[4] Docker - https://docs.docker.com/
[5] NGINX - https://nginx.org/en/docs/
[6] Typescript https://www.typescriptlang.org/docs/
[7] Angular https://angular.io/docs
[8] WSGI https://peps.python.org/pep-0333/

Kratka biografija:

Milena Vujičić rođena je u Vršcu 1997. godine. Gimnaziju „Jovan Jovanović-Zmaj“u Novom Sadu završila je 2016. godine i upisala je Fakultet tehničkih nauka, odsek Računarstvo i automatika. Osnovne akademske studije završila je 2020. godine. Iste godine upisala je master studije na istom fakultetu, odsek Primenjene računarske nauke i informatika-Računarstvo visokih performansi.
kontakt:
vujicic.milena.97@gmail.com