|  |  |  |
| --- | --- | --- |
| ZnakUniverziteta | Univerzitet u Novom Sadu  Tehnički fakultet »MihajloPupin«  Zrenjanin | LOGOpupin |

***Školska 2022/23 godina***

***Nastavni predmet: Softversko inzenjerstvo 2***

**SEMINARSKI RAD**

Tema: Evidencija primene agrotehničkih mera na parcelama

|  |  |
| --- | --- |
| Predmetni nastavnik: | Student: |
| Prof. dr Ljubica Kazi | Denis Bajin IT46/2018 |

Zrenjanin,  2023. godina

**Sadržaj:**

1. **ANALIZA POSLOVNOG SISTEMA, SPECIFIKACIJA ZAHTEVA I PLANIRANJE RAZVOJA**
   1. **Analiza poslovnog sistema**
      1. Svrha postojanja sistema i kategorizacija poslova (osnovne, pomoćne i upravljačke delatnosti)
      2. Osnovni objekti obrade i opis toka poslovnog procesa (životnog ciklusa obrade za svaki objekat, opis posla)
      3. Organizaciona struktura sistema i pripadnost drugim organizacionim strukturama
      4. Radna mesta i opis zaduženja
      5. Analiza dokumentacije (obrazaca) i arhiva (knjige evidencije, arhive dokumenata)
      6. Zakonska regulativa, standardi i interni propisi organizacionog sistema
      7. Poslovna pravila
   2. **Specifikacija zahteva korisnika**
      1. Profili korisnika i očekivane **šire** funkcionalne mogućnosti softvera (navodi se sve šta je potrebno uraditi, ali se ne navode granice šta će stvarno biti urađeno)
      2. Spisak potrebnih ulaznih ekranskih formi
      3. Spisak potrebnih tabelarnih prikaza sa kriterijumima filtriranja
      4. Spisak potrebnih štampi dokumenata (štampi, reporta), statističkih izveštaja za podršku odlučivanju i eksporta podataka
      5. Spisak poslovnih pravila, automatizama i ograničenja
      6. Potrebne karakteristike korisničkog interfejsa i ostali nefunkcionalni zahtevi
   3. **Elementi projektnog planiranja**
      1. OBUHVAT - Ciljevi, očekivani rezultati, obuhvat i granice projekta (od svih potrebnih funkcija ovde se navode samo one koje će biti realizovane i koje funkcije su izvan obuhvata)
      2. NABAVKA – potreban materijal, alati i ostala nabavka za realizaciju softvera
      3. VREMENSKI PLAN - plan aktivnosti i njihovog trajanja, ključne kontrolne tačke
2. **POSLOVNA SISTEMSKA ANALIZA I DIZAJN SOFTVERSKOG RЕŠENJA**
   1. **Model poslovnih procesa**
      1. Algoritamski model poslovnih procesa (tip Analysis, a ne Data flow dijagram)
      2. Spisak skladišta podataka (simbol valjka na dijagramu, resources, data store – potrebne arhive podataka)
      3. Analiza jednog poslovnog dokumenta (elementarni podaci, domeni kao uži skupovi dozvoljenih vrednosti)
   2. **Dizajn softverskih funkcija**
      1. Tabela preslikavanja primitivnih procesa u softverske funkcije
      2. USE CASE dijagram softverskih funkcija
      3. Specifikacija slučaja korišćenja za unos i tabelarni prikaz podataka (preduslovi, postuslovi, action steps, extension points, exceptions)
   3. **Dizajn modela podataka**
      1. Konceptualni model podataka
      2. Fizički (relacioni) model podataka
   4. **Test-bazirana specifikacija zahteva korisnika** (sa test primerima za ispravno i neispravno ponašanje korisnika)
      1. Test-bazirana specifikacija za ekransku formu unosa
      2. Test-bazirana specifikacija za ekransku formu tabelarnog prikaza sa filterom

**3. IMPLEMENTACIJA, TESTIRANJE I DOKUMENTOVANJE REALIZOVANOG SOFTVERA**

* 1. **Dokumentovanje implementacije softverskog rešenja**
     1. Kratak opis primenjenih tehnologija, programskih jezika i razvojnih alata
     2. Modeli opisa implementacije rešenja
        1. Dijagram komponenti (prikaz arhitekture softvera – višeslojna, MVC, mikroservisi…)
        2. Djagram klasa
     3. Tabelarni prikaz slojeva i podslojeva aplikacije sa ilustracijom konkretnim fajlovima u rešenju
     4. SQL script (minimum 2 tabele povezane relacijom + 1 tabela korisnik)
     5. Delovi koda sa objašnjenjima po slojevima
        1. Sloj za rad sa podacima (tehnološke klase, klase podataka, tj. repository)
        2. Sloj servisa
        3. Sloj poslovne logike sa opisom poslovnog pravila, kodom klase poslovne logike, strukturom I sadržajem datoteka u kojima su poslovna ograničenja
        4. Sloj prezentacione logike - sa klasama koje odgovaraju ekranskim formama
           1. Klase za pripremu prikaza podataka - klase sa metodama za preuzimanje i formatiranje podataka iz baze podataka kroz poziv klasa sloja za rad sa podacima
           2. Klase za preuzimanje podataka – klase za preuzimanje komandi i podataka sa korisničkog interfejsa i osnovne validacije
        5. Korisnički interfejs – koristi klase prezentacione logike
     6. Delovi koda i opis implementacije podrške personalizaciji aplikacije i bezbednosti korišćenja (prijava korisnika, rad sa sesijama, razmena podataka putem post metode)
     7. Delovi koda i opis implementacije osnovnih principa OOP (enkapsulacija, nasleđivanje, polimorfizam) i SOLID principa (single responsibility, interface segregation, dependency inversion sa dependency injection tehnikom)
     8. Delopvi koda objašnjenja realizacije clean code pristupa (čitljivost koda kroz nazive polja, procedura, klasa, komentare (uključujući CRC card komentar o svakoj klasi), konvencije oblika koda i dr)
  2. **Korisničko i tehničko uputstvo**
     1. Korisničko uputstvo (ekrani i kratak opis načina korišćenja)

# ANALIZA POSLOVNOG SISTEMA, SPECIFIKACIJA ZAHTEVA I PLANIRANJE RAZVOJA

## 1.1 Analiza poslovnog sistema

1.1.1 Svrha postojanja sistema I kategorizacija poslova

Svrha samog projekta je jednostavna, daje mogućnost da za vreme kampanje imamo prikaz useva i mašina, uz to dobijamo prikaz kolicine pracenja prijavljenih mašina. Više nivoa sadrži ova aplikacija, računajući na to da pristup programu ima osoba koja može da kreira masinu i dodaje useve koje koristi ta mašina.

Tok poslovnog procesa je :

* Registrovanje korisnika
* Sastavljanje spiskova useva i mašina
  + 1. Osnovni objekti obrade i opis toka poslovnog procesa

U ovom projektu imamo prikaz svih elementata evidencije mašina i useva po korisniku. Mogućnost dodavanja mašine i useva sa svim neophodnim informacijama posle logina je omogućena, sa datim ograničenjima poslovne logike.

* + 1. Organizaciona struktura sistema i pripadnost drugim organizacionim strukturama

Ne postoji nikakva organizaciona struktura koja učestvuje u ovom projektu.

* + 1. Radna mesta i opis zaduženja

Korisnik aplikacije – osoba koja evidentira useve I mašine date na aplikaciji.

* + 1. Analiza dokumentacije (Obrazaca) i arhiva
* Obrazac je jednovan prikaz svih mašina i useva.
  + 1. Zakonska regulativa, standardi i interni propisi organizacionog sistema

Trenutno ne postoji nikakva zakonska regulativa za mašinaa pošto se sam mašina bazira na lokalnu upotrebu.

* + 1. Poslovna pravila

U formi postoji combo box u kome se mora poštovati

## Specifikacija zahteva korisnika

* + 1. Profili korisnika i očekivane šire funkcionalne mogućnosti softvera

Korisnik aplikacije je radnik u poljoprivredi. Od korisnika se očekuje da je računarski pismen, da uz upustvo koje dolazi uz aplikaciju bude sposoban da iskoristi sve fukncionalnosti aplikacije.

Osnovna funkcija na kojoj se bazira cela aplikacija je unos podataka u bazu podataka i manipulacija podacima iz baze podataka. Aplikacija takođe treba da poseduje tabelarni prikaz podataka iz baze, uz mogućnost sortiranja i filtiranja podataka u tabeli.

Aplikacija treba da poseduje prozor u kojem će biti moguć odabir prijemnog ispita iz padajućeg menija, i tabelarni prikaz podataka za odabran us.

Dodatne tabele koje aplikacija treba da poseduje su tabela za prikaz svih prijavljenih mašina, sa mogućnošću filtriranja i sortiranja,druga tabela treba da služi za prikaz mašina koji su u nekom usu.

1.2.2 **Spisak potrebnih ulaznih ekranskih formi**

Aplikacija treba da sadrži sledeće ulazne forme:

* Forma za unos mašina sa poljima : NazivMasine, Model, Potrosnja, Primena, Slika, UsID
  + 1. Spisak potrebnih tabelarnih prikaza sa kriterijumima filtriranja

Aplikacija treba da ima sledeće tabelarne prikaze sa specifičnim kriterijumima filtriranja:

* Tabelarni prikaz svih mašina
* Tabelarni prikaz svih prijavljenih mašina za dat us: filtriranje prema nazivu mašine
  + 1. Spisak potrebnih izveštaja

Neophodni izveštaji koje aplikacija treba da obezbedi su:

* Spisak prijavljenih mašina
  + 1. Spisak potrebnih štampi dokumenata (štampi, reporta), statističkih izveštaja za podršku odlučivanju i eksporta podataka
* Spisak svih useva
* Spisak svih mašina sa filtriranjem
  + 1. Spisak poslovnih pravila, automatizama i ograničenja
* Maksimalna protrošnja po mašini je 20 litara.
  + 1. Potrebne karakteristike korisničkog interfejsa i ostali nefunkcionalni zahtevi

Korisnički interfejs treba izgledom da se svidi korisniku, ali isto tako je obavezno to da pruža način za intuitivan i lak rad sa aplikacijom. Tekst koji se pojavljuje u aplikaciji treba da bude jasan i čitljiv, ne sme biti napisan nekim fontovima koji su teški za čitanje poput cursive fontova.

Mora postojati lak način za navigaciju između formi aplikacije, labele i njihova polja moraju biti jasno definisani da bi pravilan unos podataka bio lakši.

U tabelama se moraju jasno staviti nazivi kolona u kojima se izlistani podaci nalaze.

## Elementi projektnog planiranja

### OBUHVAT - Ciljevi, očekivani rezultati, obuhvat i granice projekta

U ciljeve koje je potrebno postići spadaju: pružanje intuitivnog korisničkog interfejsa, pružanje jasnog i jednostavnog prikaza dostupnih funkcija i dostupnih podataka, dugoročna upotrebljivost softvera, visok stepen stabilnosti softvera,

Cilj aplikacije je da pruži jednu univerzalnu platformu koja će obuhvatiti kompletan poslovni proces koji okružuje korisnika da će time omogućiti da se na lak i efikasan način upravlja podacima koji se prikupljaju tokom tog procesa. Aplikacija treba da omogući korisniku da pristupi bazi podataka, da doda sadržaj u nju, da ga kasnije izmenii konačno treba da omogući korisniku izvoz tih podataka tako što će aplikacija pružiti mogućnost štampanja na papiru.

Očekivani rezultat upotrebe aplikacije je pojednostavljenje procesa rada unosom i manipulacijom podataka, povećanje efikasnosti i redukovanje broja grešaka.

Mašina obuhvata jednu aplikaciju koja treba da pruži pomenutu platformu za unos, manipulaciju, prikaz, obradu i štampanje podataka iz baze podataka.

U GRANICAMA OBUHVATA, ALI NECE BITI URADJENO:

* Registracija useva na veb sajtu, unos podataka, upload dokumenata
* Ekranske forme za verifikaciju unetih podataka od  strane useva, koje koriste referenti mašina, nakon uvida u originalne dokumente

VAN GRANICA:

Računanje bilansa useva.

* + 1. Nabavka

Za samog mašinaa neophodno je hardverski imati računar i internet. Softverski potrebno je imati alate poput Visual Studio Code, SQL management i SQL server i xampp local server.

* + 1. Vremenski plan

Vremenski plan projekta je dizajniran da bude za efektivne 3 nedelje od početka planiranja projekta.

# 2.POSLOVNA SISTEMSKA ANALIZA I DIZAJN SOFVERSKOG REŠENJA

2.1.1 Algoritamski model poslovnih procesa

Diagram

Description automatically generated

2.1.2 Spisak skladišta podataka

Podaci se skladište u sledećim skladištima:

* Skladište podataka o mašinama
* Sklatište podataka o usevima
* Skladište rezultata mašina
* Skladište spiska mašina za određeni us

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| POSLOVNI PROCES | | PRVI PRIORITET | | DRUGI PRIORITET | | PREDUSLOV | | TEHNOLOGIJA IMPLEMENTACIJE |
| Primitivni poslovni proces | Radna uloga | Softverska funkcija \*1 po prioritetu | Actor | Softverska funkcija \*2 po prioritetu | Actor | Softverska funkcija \* preduslov za uspešno izvršavanje | Actor | Tip softvera (c/s LAN, WEB) |
| Akreditovanje liste mašina |  | Unos podataka o novim mašinama |  | Tabelarni prikaz unetih mašina  Filtriranje mašina  Štampanje podataka o mašinama |  |  |  |  |
| Pribavljanje useva | Marketing sluzba | Unos podataka o novim usevima | Kadrovska služba | Tabelarni prikaz unetih useva  Filtriranje useva  Štampanje spiska mašina |  | Unos podataka o mestu  Unos podataka o radnom mestu  Unos podataka o radnom ugovoru | Kadrovska služba |  |
| Formiranje konkursa |  | Unos podataka o  novom konkursu | Kadrovska služba | Tabelarni prikaz svih unetih useva  Filtriranje useva za dati mašina  Štampanje konkursa |  | Unos podataka o studijskom programu  Unos podataka o instituciji  Unos pravila konkurisanja |  |  |
| Formiranje komisije za građane |  | Unos podataka o  novoj komisiji | Kadrovska služba | Tabelarni prikaz komisije  Filtriranje komisije prema imenu i prezimenu člana komisije  Štampanje liste članova komisije |  | Unos podataka o zajednici  Unos podataka o konkursu | Sluzba za građane |  |
| Organizovanje prijemnog ispita | Kadrovska sluzba | Unos podataka o građanima | Kadrovska služba | Tabelarni prikaz svih unetih mašinaa  Filtriranje prema predmetu, šifri ispita  Štampanje spiska mašinaa |  | Unos podataka o projektima | Sluzba za projekte | c/s LAN |
| Prijavljivanje useva za građane | Sponzor | Unos podataka o novom zajednica | Referent | Tabelarni prikaz prijavljenih useva  Filtriranje useva  Štampanje spiska useva |  | Unos podataka o useva  Unos podataka o projektu | Sluzba za projekte | c/s LAN |
| Evidentiranje rezultata mašinaa | Radnik | Ažuriranje podataka o zajednicama | Nastavno osoblje | Tabelarni prikaz svih useva  Filtriranje mašine prema imenu i prezimenu |  | Unos podataka o usevima  Unos podataka o građanima | Radnik | c/s LAN |
| Objavljivanje rezultata | Referent | Štampanje rezultata | Referent | Tabelarni prikaz useva  Filtriranje prema imenu i prezimenu mašinaa |  | Unos podataka o usevima  Izmena podataka o usevima |  | c/s LAN |

2.1.3 Analiza jednog poslovnog dokumenta

|  |  |  |
| --- | --- | --- |
| **Naziv elementarnog podatka** | **Tip podatka** | **Domen** |
| ID | Int |  |
| NazivMasine | String |  |
| Model | String |  |
| Potrosnja | Number |  |
| Primena | String |  |
| Slika | String |  |
| UsID | Int |  |

2.2. Dizajn softverskih funkcija

2.2.1 Tabela preslikavanja primitivnih procesa softverske funkcije

2.2.2 USE CASE dijagram softverskih funkcija

Diagram

Description automatically generated

2.2.3 Specifikacija slučaja korišćenja za unos i tablearni prikaz podataka

Specifikacija slučaja korišćenja za unos prikazana je na slikama ispod.

2.3 Dizajn modela podataka

2.3.1 Konceptualni model podataka

Graphical user interface, text, application

Description automatically generated

2.3.2 Fizicki model podataka

2.4 Test-bazirana specifikacija zahteva korisnika

2.4.1. Test-bazirana specifikacija za ekransku formu unos useva

Naredna test bazirana specifikacija pisana je za formu za unos novog mašinaa:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Redni broj test slučaja** | **Radno okruženje testiranja : računar(memorija, procesor), operativni sistem** | **Kategorija i tip validacije (osnovna, poslovna)** | **Obim testiranja** | **Radna operacija u softveru (unos, brisanje)** | **Deo programa (ekranska forma, polje)** | **Tip testiranja (ispravni podaci, neispravni podaci)** | **Test podaci** | **Postupak primene testa** | **Očekivano ponašanje (kriterijum prihvatljivosti i ponašanja)** |
|  | Razvojno okruženje |  |  |  |  |  |  |  |  |
|  |  | **Osnovne validacije** |  |  |  |  |  |  |  |
|  |  | Da li je popunjeno – not null | Jedno polje po izboru | Unos | Ekranska forma za unos podataka o stranci  Polje za unos imena useva | Ispravni | Popunjeno polje za ime useva | Korisnik popunjava polje za ime useva, bira taster za snimanje | Poruka o uspešnom unosu novog useva |
|  |  |  |  |  |  | Neispravni | Polje za ime useva ostaje prazno | Korisnik ostavlja polje za ime korisnika prazno | Poruka o tome da sva polja obavezna polja nisu popunjena, povratak na formu za unos i obeležavanje praznog polja |
|  |  | Da li su odgovaraući karakteri | Jedno polje po izboru | Unos | Ekranska forma za unos podataka o stranci  Polje za unos imena useva | Ispravni | U polje za ime useva se unose samo slovni karakteri | Korisnik unosi niz slovnih karaktera u polje ime useva | Poruka o uspešnom unosu novog useva |
|  |  |  |  |  |  | Neispravni | U polju za ime se nalaze karakteri koji nisu slovni karakteri | Korisnik u polje za ime unosi karaktere koji nisu slovni karakteri | Poruka o pogrešnom unosu, povratak na formu za unos i obeležavanje polja u pitanju |
|  |  | Da li su odgovarajući tipovi podataka | Jedno polje po izboru | Unos | Ekranska forma za unos podataka o stranci  Polje za unos imena useva | Ispravni | U polje za ime useva unosi se niz slovnih karaktera | Korisnik u polje za ime useva unosi niz slovnih karaktera | Poruka o uspešnom unosu novog useva |
|  |  |  |  |  |  | Neispravni | U polje za ime useva se unose brojevi ili specijalni karakteri | Korisnik u polje za ime useva unosi brojeve ili specijalne karaktere | Poruka o pogrešnom unosu, povratak na formu za unos i obeležavanje polja u pitanju |
|  |  | Da li je odgovarajuća dužina unetog podatka | Jedno polje po izboru | Unos | Ekranska forma za unos prijemnog ispita  Polje za unos šifre prijemnog ispita | Ispravni |  |  |  |
|  |  |  |  |  |  | Neispravni | U polje za šifru prijemnog ispita se upisuje niz karaktera kraći od 6 | Korisnik u polje za unos šifre prijemnog ispita unosi niz karaktera kraći od 6 | Poruka o pogrešnom unosu, vraćanje na formu za unos, obeležavanje polja u pitanju |
|  |  | Da li je jedinstveni zapis po primarnom ključu | Ceo zapis, proveza za primarni ključ | Unos |  | Ispravni | U tabelu baze podataka se unosi nova stavka kojoj se automatski dodeljuje jedinstven identifikacioni broj | Korisnik izvršava snimanje priloženih podataka, novoj stavci neke od tabela baze podataka se automatski dodeljuje jedinstveni identifkacioni broj | Poruka o uspešnom unosu podataka preko neke od dostupnih formi |
|  |  | Da li se kombinacija neključnih obeležja ponavlja | Ceo zapis, provera za neključna obeležja | Unos |  | Ispravni | U bazi podataka postoje zapisi prijemnih ispitima sa različitim šiframa ispita | Korisnik unosi šifru novog ispita | Poruka o uspešnom unosu podataka |
|  |  |  |  |  |  | Neispravni | U bazi podataka | Korisnik unosi |  |
|  |  | Da li su povezani podaci različitih tabela | Zapisi povezanih tabela |  |  | Ispravni |  |  | Postoji veza između tabela podataka  Tabela useva koristi podatke iz tabele mašinaa |
|  |  | **Poslovne validacije** |  |  |  |  |  |  |  |
|  |  | Validacije realizovane u poslovnoj logici, a odnose se na realizaciju poslovnih pravila (AKO uslov ONDA automatska akcija ili zabrana) |  |  | Metoda za proveru broja bodova useva | Ispravni |  |  | Ukoliko je uslov ispunjen, dozvoljena je izmena podatka o broju bodova useva |
|  |  |  |  |  | Metoda za proveru broja bodova useva | Neispravni |  |  | Ukoliko uslov nije ispunjen, ažuriranje tabele se obustavlja, ispisuje se poruka o grešci prilikom unosa i vraća se na ekran za ažuriranje podataka |
|  |  |  |  |  | Metoda za proveru jedinstvenosti prijave useva | Ispravni | Sponzor je prijavljen za samo jedan mašina u konkursnom roku | Korisnik unosi podatke o novom stranci koji se nije prijavio za drugi mašina u istom konkursnom roku | Unos novog useva je uspešan |
|  |  |  |  |  | Metoda za proveru jedinstvene prijave useva | Neispravni | Zajednica je već prijavljena za jednog mašinaa | Korisnik unosi podatke o zajednici koji je već prijavljen za mašinaa. | Blokiran je unos novog useva, prikazuje se poruka o postojanju, povratak na formu za unos |

2.4.2 Test-bazirana specifikacija za ekransku formu tabelarnog prikaza mašine sa filterom

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Redni broj test slučaja** | **Redni broj test slučaja** | **Radno okruženje testiranja : Računar (Memorija, procesor), operativni sistem, razvojno okruženje** | **Kategorija i tip validacije (osnovna, poslovna)** | **Obim testiranja** | **Radna operacija u softveru (unos, brisanje)** | **Deo programa (ekranska forma, polje)** | **Tip testiranja (Ispravni podaci, neispravni, prazno polje)** | **Test podaci** | **Postupak primene testa** | **Očekiv**  **ano pon**  **ašanje (kriterij**  **um prihvatlj**  **ivosti)** | **Radno okruženje testiranja : Računar (Memorija, procesor), operativni sistem, razvojno okruženje** | **Kategorija i tip validacije (osnovna, poslovna)** | **Obim testiranja** | **Radna operacija u softveru (unos, brisanje)** | **Deo programa (ekranska forma, polje)** | **Tip testiranja (Ispravni podaci, neispravni, prazno polje)** | **Test podaci** | **Postupak primene testa** | **Očekiv**  **ano pon**  **ašanje (kriterij**  **um prihvatlj**  **ivosti)** |
|  |  |  | **Osnovne validacije** |  |  |  |  |  |  |  |  | **Osnovne validacije** |  |  |  |  |  |  |  |
|  |  |  | Slučaj kada je polje prazno | Jedno polje po izboru | Unos | Polje Filter | Polje za filtriranje je prazno | Prazno polje za filtriranje | Korisnik ostavlja polje za filtriranje prazno | Klikom na dugme Filter dok je filter polje prazno prikazuje sve podatke u tabeli |  | Slučaj kada je polje prazno | Jedno polje po izboru | Unos | Polje Filter | Polje za filtriranje je prazno | Prazno polje za filtriranje | Korisnik ostavlja polje za filtriranje prazno | Klikom na dugme Filter dok je filter polje prazno prikazuje sve podatke u tabeli |
|  |  |  | Slučaj kada je polje popunjeno | Jedno polje po izboru | Unos | Polje Filter | Ispravni | Polje za filtriranje je popunjeno validnim sadržajem | Korisnik unosi validan niz karaktera u filter | Klikom na dugme filter, u tabeli se prikazuju samo one stavke koji odgovaraju filteru |  | Slučaj kada je polje popunjeno | Jedno polje po izboru | Unos | Polje Filter | Ispravni | Polje za filtriranje je popunjeno validnim sadržajem | Korisnik unosi validan niz karaktera u filter | Klikom na dugme filter, u tabeli se prikazuju samo one stavke koji odgovaraju filteru |
|  |  |  |  |  | Unos | Polje Filter | Neispravni | Polje za filtriranje nije popunjeno validnim sadržajem | Korisnik u polje za filter unosi niz karaktera koji nije validan | Ispisuje se poruka o nepostojećem kriterijumu filtriranja, povratak na tabelarni prikaz bez filtera |  |  |  | Unos | Polje Filter | Neispravni | Polje za filtriranje nije popunjeno validnim sadržajem | Korisnik u polje za filter unosi niz karaktera koji nije validan | Ispisuje se poruka o nepostojećem kriterijumu filtriranja, povratak na tabelarni prikaz bez filtera |

* 1. Plan realizacije elemenata kvaliteta softvera
     1. Funkcionalni aspekt kvaliteta softvera
        1. Planirane funkcionalne mogućnosti softvera i profil korisnika

Planirano je da softver koriste : dežurni, referent i administrator

Dežurni ima pristup sledećim funkcijama :

* Tabelarni prikaz mašine
* Filtriranje mašine

Referent ima pristup sledećim funkcijama :

* Unos, ažuriranje i brisanje useva
* Tabelarni prikaz useva
* Unos i ažuriranje mašine
* Tabelarni prikaz mašine
* Tabelarni prikaz useva za svaki pojedinačni mašina
* Stranica za štampu svih navedenih tabela

Administrator ima pristup istim funkcijama kao i referent, sa sledećim proširenjima :

* Brisanje mašine
* Unos, ažuriranje i brisanje useva
* Pristup svim tabelarnim prikazima
* Pristup svim mašinaima za štampu
  + - 1. Plan realizacije osnovne validacije podataka

Radnik:

* Polja za unos ne smeju biti prazna
* Šifra useva mora biti jedinstvena kombinacija slova i brojeva dužine 6 karaktera
* Ime i prezime moraju sadržati samo slovne karaktere

Mašina :

* Polja za unos ne smeju biti prazna
* Šifra predmeta mora biti jedinstvena kombinacija slova i brojeva dužine 6 karaktera

Stranka :

* Polja za unos ne smeju biti prazna
* ID useva mora biti jedinstven
* Naziv masine mogu sadržati samo slovne karaktere
  + - 1. Plan realizacije podrške poslovnoj logici
* Jedan mašina može biti samo u jednoj usevima.
  + - 1. Plan podrške personalizaciji aplikacije i bezbednosti korišćenja

Korišćenju aplikacije se pristupa preko korisničkog naloga. Aplikacija prepoznaje tri uloge korisnika : radnik, referent i administrator

Administrator vrši unos mašine, ima privilegije da briše sadržaj iz baze podataka. Ima pristup svim tabelarnim prikazima i svim stranicama za štampu.

Referent vrši unos i ažuriranje useva. Ima omogućen pristup tabelarnom prikazu useva, mašinaa i tabelarnog prikaza useva za pojedinačni mašina. Ima pristup stranicama za štampu useva, mašinaa i spiska useva za svaki mašina.

Radnik vrši ažuriranje podataka o stranci – konkretno se očekuje da unese donaciju koje sponzor daje projektu. Ima pristup tabelarnom prikazu useva za pojedinačni ispit.

Pristup tabelarnom prikazu automatski omogućava pristup svim filterima koje tabelarni prikaz podržava.

* 1. Strukturni aspekt kvaliteta softvera
     + 1. Plan ahritekture softvera



* + - 1. Plan realizacije clean code pristupa

Pristup čistog koda realizovaće se na sledeći način:

* Za rešavanje problema koristiće se objektno-orijentisan pristup,
* Pišu se .net klase i koristi se princip nasleđivanja,
* .net kod koji se više puta ponavlja, izdvaja se u zaseban fajl za pomoćne funkcije,
* Piše se semantički HTML,
* Komentari opisuju svaku logičku strukturu unutar HTML dokumenta,
* CSS kod se izdvaja u zaseban fajl,
* Koriste se CSS varijable zbog ponavljanja vrednosti,
* Imena CSS klasa precizno definišu funkciju ili karakteristike elementa kojem pripadaju
* Nazivi baze podataka, tabela i atributa u SQL koriste pascal case
* Prilikom pisanja asp.net koda koristi se pascal case
  1. 3.IMPLEMENTACIJA, TESTIRANJE I DOKUMENTOVANJE REALIZOVANOG SOFTVERA  
     Dokumentovanje implementacije softverskog rešenja

3. 1. 1. Kratak opis primenjenih tehnologija, programskih jezika i razvojnih alata

Za izradu ovog projekta upotrebljeni su sledeći programski jezici : HTML, CSS, .net i CSS framework – Bootstrap

Program u kojem je izvršena simulacija servera i baze podataka je XAMPP

Projekat je razvijen u alatu Microsoft Visual Studio Code

HTML – programski jezik namenjen za kreiranje veb stranica koje će se nalaziti na internetu. Zasnovan je na upotrebi tagova unutar kojih se smešta željeni sadržaj koji će se kasnije prikazivati na stranici. Najnovija verzija HTML-a je HTML 5 i on je korišten za izradu ovog projekta.

HTML dokument mora posedovati sledeće tagove

* !Doctype HTML – precizira pretraživaču da se radi o HTML dokumentu
* html – koren HTML dokumenta, između html tagova se nalaze head i body tagovi
* head – sadrži meta podatke o veb stranici, i ovde se postavljaju linkovi do eksternih Stylesheet fajlova i drugih sličnih dokumenata
* body – sadrži sve ono što će se prikazati na sajtu prilikom pokretanja

CSS – je programski jezik koji se koristi za stilizaciju veb sajtova, odnosno za stilizaciju HTML elemenata. CSS može da se upotrebi na tri načina, iz eksternog fajla, u style atributu HTML elementa i u okviru head tagova unutar style tagova. U ovom projektu korišćen je isključivo CSS iz eksternog fajla, svi neophodni fajlovi linkovani su u okviru head dela dokumenta.

Bootrap – je framework koji objedinjuje CSS i JavaScript. Bootstrap pruža veliki broj gotovih CSS klasa koje se mogu dodavati HTML elementima u svrhe postizanja željenih stilizacija na brz i efikasan način. Naravno, upotreba Bootstrap klasa ne isključuje upotrebu vlastitih klasa developera, ali postoji mogućnost da će doći do konflikta između nekih stilova predefinisanih klasa i vlastitih klasa developera. Boostrap se može integrisati u mašina preko CDN-a (Content Delivery Network), tj. linka do online fajlova za Bootstrap CSS i Bootstrap JavaScript, ili tako što se Bootstrap fajlovi preuzmu sa izvora, smeste u mašina i potom adekvatno linkuju. Head treba da sadrži link do BootstrapCSS a na dno body elementa se smešta script tag sa putanjom do BootstrapJS. Da bi Bootstrap funkcionisao neophodno je uvesti Jquery.

Jquery – je JavaScript biblioteka koja se koristi za laku i efikasnu selekciju raznih elemenata HTML dokumenta i za omogućavanje lake manipulacije tim elementima. Može se preuzeti u formi source koda ili linkovati uz pomoć CDN. U projektu je Jquery uključen samo zbog zahteva Bootstrap framework-a.

ASP.NET je okvir veb aplikacija koji je razvio Microsoft koji omogućava programerima da grade dinamičke veb stranice, veb aplikacije i veb usluge. Izgrađen je na zajedničkom jeziku za izvršavanje (CLR), što omogućava programerima da pišu ASP.NET kod koristeći bilo koji podržani .NET jezik. ASP.NET je popularan izbor za pravljenje veb aplikacija jer pruža jednostavan, lak za učenje i moćan skup alata za pravljenje dinamičkih veb stranica. Pored toga, uključuje bogat skup biblioteka i okvira za zadatke kao što su pristup bazi podataka, bezbednost i autentifikacija korisnika.XAMPP – je besplatnan, open-source višeplatformski simulator veb servera koji je razvio Apache Friends i sačinjen je iz Apache HTTP servera, MariaDB baze podataka i interpretatora za skripte napisane u .net i Perl programskim jezicima. Pruža idealno okruženje za testiranje rešenja pre puštanja na realan server. U ovom projektu korišteni su samo Apache HTTP server i MariaDB baza podataka.

Microsoft Visual Studio Code – je besplatan tekstualni editor kompanije Microsoft koji pruža veoma veliki broj funkcionalnosti, mogućnosti proširenja, visok stepen kvaliteta za lak i efikasan rad prilikom razvijanja softvera i dostupan je na Windows, Linux i MacOS platformama. Ovaj editor može se prilagoditi radu sa veoma velikom brojem programskih jezika preko brojnih dodataka i proširenja koji su dostupni na tržištu. Takođe podržava ugrađivanje Git-a direktno u editor za lakšu kontrolu verzija projekta.

* + 1. Modeli opisa implementacije rešenja
       1. Dijagram komponenti



* + - 1. Dijagram klasa

Diagram

Description automatically generated

* + 1. Tabelarni prikaz slojeva aplikacije sa ilustracijom konkernih fajlova u rešenju

|  |  |  |
| --- | --- | --- |
| **Glavni Sloj** | **Podslojevi** | **Implementacija u radu** |
| Prezentacioni sloj | Korisnički interfejs | Stranice za unos podataka i stranice za prikaz podataka u formi liste |
| Prezentaciona logika | Prezentaciona logika implementirana je uz pomoć funkcija koje su u sklopu klasa „UsKlasa“ i „MašinaKlasa“ |
| Sloj servisa | Web servis | Nije realizovano |
| Servisni sloj kao biblioteka klasa | Nije realizovano |
| Sloj poslovne logike | Poslovni objekti | Klase „UsKlasa“ i „MašinaKlasa“ su poslovni objekti u ovom radu |
| Poslovna pravila | Poslovna pravila realizovana na stranicama iz foldera „LogicLayer“ u kojem se nalazi logika za unos, izmenu i brisanje stavki iz baze podataka |
| Radni tokovi | Nije realizovano |
| Sloj za rad sa podacima | Klase podataka | Postoji jedan tip klasa podataka. Klase „Korisniklasa“, „MašinaKlasa“ i „UsKlasa“ nasleđuju baznu tehnološku klasu i koriste njene metode u svom kontekstu. |
| Tehnološke klase | Bazna klasa „BaseClass“ sadrži metode koje direktno komuniciraju sa bazom podataka. |
| Baza podataka | Realizovana je klasa za konekciju ka bazi podataka „KonekcijaKlasa“ koja ima metode za upravljanje konekcijom ka bazi podataka |

3.1.4 SQL skript

USE [Agrokultura]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Masina] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Masina](

[ID] [int] NOT NULL,

[NazivMasine] [nvarchar](50) NULL,

[Model] [nvarchar](50) NULL,

[Potrosnja] [int] NULL,

[Primena] [nvarchar](50) NULL,

[Slika] [nvarchar](50) NULL,

[UsID] [int] NULL,

CONSTRAINT [PK\_Masina] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Us] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Us](

[UsID] [int] NOT NULL,

[NazivUseva] [nvarchar](50) NULL,

[Velicina] [int] NULL,

CONSTRAINT [PK\_Us] PRIMARY KEY CLUSTERED

(

[UsID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Users] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Users](

[ID] [int] NULL,

[Username] [nvarchar](50) NULL,

[Password] [nvarchar](50) NULL

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spIzbrisiMasinu] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spIzbrisiMasinu]

-- Add the parameters for the stored procedure here

@ID int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT \* FROM Masina WHERE @ID = ID

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spPostaviMasinu] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spPostaviMasinu]

-- Add the parameters for the stored procedure here

@ID int,

@NazivMasine nvarchar(50),

@Model nvarchar(100),

@Potrosnja int,

@Primena nvarchar(50),

@Slika nvarchar(100),

@UsID int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

INSERT INTO Masina(ID,NazivMasine,Model,Potrosnja,Primena,Slika,UsID)

VALUES (@ID, @NazivMasine, @Model, @Potrosnja, @Primena, @Slika, @UsID)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spUpdateMasina] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spUpdateMasina]

-- Add the parameters for the stored procedure here

@ID int,

@NazivMasine nvarchar(50),

@Model nvarchar(100),

@Potrosnja int,

@Primena nvarchar(50),

@Slika nvarchar(100),

@UsID int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

UPDATE Masina

SET NazivMasine = @NazivMasine,

Model = @Model,

Potrosnja = @Potrosnja,

Primena = @Primena,

Slika = @Slika,

UsID = @UsID

WHERE ID = @ID

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spUzmiMasinuPoId] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spUzmiMasinuPoId]

-- Add the parameters for the stored procedure here

@ID int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT \* FROM Masina WHERE @ID = ID

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spUzmiMasinuPoUsevima] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spUzmiMasinuPoUsevima]

-- Add the parameters for the stored procedure here

@UsID int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT \* FROM Masina

WHERE @UsID = UsID

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spUzmiSveMasine] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spUzmiSveMasine]

-- Add the parameters for the stored procedure here

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT \* FROM Masina

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[spUzmiSveUseve] Script Date: 3/19/2023 9:49:38 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Author,,Name>

-- Create date: <Create Date,,>

-- Description: <Description,,>

-- =============================================

CREATE PROCEDURE [dbo].[spUzmiSveUseve]

-- Add the parameters for the stored procedure here

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT \* FROM Us

END

GO

### 3.1.5 Delova koda

3.1.5.1 Mašina klasa

Ovo je glavna klasa koja daje mogućnosti API programu da generiše podatke iz baze. U ovom primeru kao što možemo videti dole, imamo prikaz čitanja podataka iz baze (uzmiSveMasine) preko te rute koristimo podatke koje izbacujemo u JSON formatu za korisnički interfejs. Takođe imamo sličan do isti princip za menjanje i ubacivanje mašine kao klase. using Agrokultura.Models;

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Cors;

namespace Agrokultura.Controllers

{

[EnableCors(origins: "\*", headers: "\*", methods: "\*")]

public class MasinaController : ApiController

{

private SqlConnection connection;

public MasinaController()

{

this.connection = new SqlConnection(ConfigurationManager.AppSettings["ConnectionString"]);

}

// GET api/values

[HttpGet]

[Route("uzmiSveMasine")]

public HttpResponseMessage UzmiSveMasine()

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spUzmiSveMasine", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

[HttpGet]

[Route("uzmiSveUseve")]

public HttpResponseMessage UzmiSveUseve()

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spUzmiSveUseve", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

[HttpGet]

[Route("UzmiSveMasinePoId/{id}")]

public HttpResponseMessage UzmiSveMasinePoId(int id)

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spUzmiMasinuPoId", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

adapter.SelectCommand.Parameters.Add("@ID", SqlDbType.Int);

adapter.SelectCommand.Parameters[0].Value = id;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

[HttpGet]

[Route("UzmiMasinuOdUseva/{idUsa}")]

public HttpResponseMessage UzmiMasinuOdUseva(int idUsa)

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spUzmiMasinuPoUsevima", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

adapter.SelectCommand.Parameters.Add("@UsID", SqlDbType.Int);

adapter.SelectCommand.Parameters[0].Value = idUsa;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

// POST api/values

//public void Post([FromBody] string value)

//{

//}

[HttpPost]

[Route("NapraviMasinu")]

public IHttpActionResult NapraviMasinu(Masina masina)

{

using (var connection = new SqlConnection(ConfigurationManager.AppSettings["ConnectionString"]))

{

using (var command = new SqlCommand("spNapraviMasinu", this.connection))

{

command.CommandType = CommandType.StoredProcedure;

// Add the parameters for the stored procedure

command.Parameters.Add("@ID", SqlDbType.Int).Value = masina.ID;

command.Parameters.Add("@NazivMasine", SqlDbType.NVarChar, 50).Value = masina.NazivMasine;

command.Parameters.Add("@Model", SqlDbType.NVarChar, 100).Value = masina.Model;

command.Parameters.Add("@Potrosnja", SqlDbType.Int).Value = masina.Potrosnja;

command.Parameters.Add("@Primena", SqlDbType.Char, 10).Value = masina.Primena;

command.Parameters.Add("@Slika", SqlDbType.NVarChar, 50).Value = masina.Slika;

command.Parameters.Add("@UsID", SqlDbType.Int).Value = masina.UsID;

// Execute the stored procedure

this.connection.Open();

command.ExecuteNonQuery();

this.connection.Close();

}

}

// Return a success status code

return Ok();

}

// PUT api/values/5

[HttpPut]

[Route("IzmeniMasinu/{id}")]

public IHttpActionResult IzmeniMasinu(Masina masina)

{

using (var connection = new SqlConnection(ConfigurationManager.AppSettings["ConnectionString"]))

{

using (var command = new SqlCommand("spIzmeniMasinu", this.connection))

{

command.CommandType = CommandType.StoredProcedure;

// Add the parameters for the stored procedure

command.Parameters.Add("@ID", SqlDbType.Int).Value = masina.ID;

command.Parameters.Add("@NazivMasine", SqlDbType.NVarChar, 50).Value = masina.NazivMasine;

command.Parameters.Add("@Model", SqlDbType.NVarChar, 100).Value = masina.Model;

command.Parameters.Add("@Potrosnja", SqlDbType.Int).Value = masina.Potrosnja;

command.Parameters.Add("@Primena", SqlDbType.Char, 10).Value = masina.Primena;

command.Parameters.Add("@Slika", SqlDbType.NVarChar, 50).Value = masina.Slika;

command.Parameters.Add("@UsID", SqlDbType.Int).Value = masina.UsID;

// Execute the stored procedure

this.connection.Open();

command.ExecuteNonQuery();

this.connection.Close();

}

}

// Return a success status code

return Ok();

}

// DELETE api/values/5

[HttpDelete]

[Route("IzbrisiMasinu/{id}")]

public HttpResponseMessage IzbrisiMasinu(int id)

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spIzbrisiMasinu", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

adapter.SelectCommand.Parameters.Add("@ID", SqlDbType.Int);

adapter.SelectCommand.Parameters[0].Value = id;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

}

}

3.1.5.2 Korisnički interfejs app.js za mašinu

U ovoj dole prikazanoj funkciji imamo prikaz kako pozivamo API iz prethodnog primera i oblikujemo tako da prikažemo sve građane za glasanje.

const URL = "localhost/Agrokultura";

const API\_GET\_PRODUCT\_FOR\_CATEGORY = "http://localhost/Agrokultura/api/Products/GetItemsByCategory/";

const GET\_ALL\_AGROKULTURA = "http://localhost/Agrokultura/uzmiSveMasine";

function getJson(url, callback) {

    fetch(url).then((response) => response.json())

    .then(data => callback(data.Table))

    .catch(error => console.log(error))

}

function getMenu(){

    fetch("menu.html").then(response =>

        response.text()

    ).then(data => {

        document.querySelector('.meni').innerHTML = data;

    })

}

function getPageImage(){

    fetch("pageImage.html").then(response =>

        response.text()

    ).then(data => {

        document.querySelector('.pageImage').innerHTML = data;

    })

}

function getOnama() {

    fetch("izmena.html").then(response =>

        response.text()

    ).then(data => {

        document.querySelector('.izmena').innerHTML = data;

    })

}

function getUbacaj() {

    fetch("ubacaj.html").then(response =>

        response.text()

    ).then(data => {

        document.querySelector('.ubacaj').innerHTML = data;

    })

}

function getMainContent(){

    fetch("mainContent.html").then(response =>

        response.text()

    ).then(data => {

        document.querySelector('.mainContent').innerHTML = data;

    })

}

// Funkcije filtriranja

function ispisiSveMasine() {

    getJson(GET\_ALL\_AGROKULTURA,(data) => {

        let masineData ="";

        data.map((values)=> {

            masineData+=`

            <div class="back-arrow" onclick="ispisiSveMasine();">Povratak</div>

             <div class="card-wrapper" onclick="singleItem(this)">

             <div class="show-id">${values.ID}</div>

             <div class="brisanje-masine" onclick=brisanje(${values.ID})>Brisanje</div>

             <div class="izmena-biraca" onclick="showOnama();">Izmena</div>

             <div class="image-wrapper">

              <img src=/images/products/${values.Slika}>

             </div>

             <h1>${values.NazivMasine}</h1>

             <div class="description">${values.Model}</div>

             </div>

             `;

        });

        document.querySelector('.main-cards').innerHTML = masineData;

    })

    let mainCards = document.querySelector('.main-wrapper');

}

// Prikaz jednu masinu

function singleItem(selekt) {

    let elements = document.querySelectorAll('.card-wrapper');

    // console.log(selekt);

    let item = document.querySelector('.card-wrapper');

    let nonSelectedItems = Array.from(elements).filter(item => {

        return item !== selekt;

    }).forEach(item => {

        item.style.display = 'none';

    })

    selekt.classList.add('selectedItem');

    selekt.parentNode.querySelector('.back-arrow').style.display = 'block';

    selekt.querySelector('.brisanje-masine').style.display = 'block';

    selekt.querySelector('.description').style.display = 'block';

}

function UzmiMasinuOdUsa(usID){

    getJson('http://localhost/Agrokultura/UzmiMasinuOdUseva/' + usID,(data) => {

        let productData="";

       data.map((values)=> {

             productData+=`

             <div class="back-arrow" onclick="ispisiSveMasine();">Povratak</div>

             <div class="card-wrapper" onclick="singleItem(this)">

             <div class="image-wrapper">

              <img src=/images/products/${values.ImageUrl}>

             </div>

             <h1>${values.NazivMasine}</h1>

             <div class="description">${values.Model}</div>

             </div>`;

         });

         document.querySelector('.main-cards').innerHTML =  "<div class='category-title'>" + "</div>"  + productData;

    })

    let mainCards = document.querySelector('.main-wrapper');

}

// Brisanje jedne masine

async function brisanje(id){

    const response = await fetch(`http://localhost/Agrokultura/IzbrisiMasinu/${id}`,{

    method: 'DELETE'

});

    if (response.ok){

        alert('Машина избрисана');

        ispisiSveMasine();

    }else {

        console.log('Nesto ne radi');

    }

    }

getMenu();

getPageImage();

getOnama();

getUbacaj();

getMainContent();

ispisiSveMasine();

function showPocetna() {

    let onamaPage = document.querySelector(".izmena");

    onamaPage.style.display = 'none';

    let mainContent = document.querySelector(".mainContent");

    mainContent.style.display = 'block';

    let ubacaj = document.querySelector(".ubacaj");

    ubacaj.style.display = 'none';

}

function showOnama(){

    let onamaPage = document.querySelector(".izmena");

    onamaPage.style.display = 'block';

    let mainContent = document.querySelector(".mainContent");

    mainContent.style.display = 'none';

    let ubacaj = document.querySelector(".ubacaj");

    ubacaj.style.display = 'none';

}

function prikaziUbacaj(){

    let ubacaj = document.querySelector(".ubacaj");

    ubacaj.style.display = 'block';

    let mainContent = document.querySelector(".mainContent");

    mainContent.style.display = 'none';

    let onamaPage = document.querySelector(".izmena");

    onamaPage.style.display = 'none';

}

3.1.5.4 Klasa mesne useva

Klasa mesne zajenice je ista klasa kao i gore prikazana klasa mašine sa limitiranim funkcijama računajući na zahtevnost aplikacije.

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web;

using System.Web.Http;

using System.Web.Http.Cors;

using System.Web.Mvc;

namespace Gradjani.Controllers

{

[EnableCors(origins: "\*", headers: "\*", methods: "\*")]

public class MesnaZajednicaController : ApiController

{

private SqlConnection connection;

public MesnaZajednicaController()

{

this.connection = new SqlConnection(ConfigurationManager.AppSettings["ConnectionString"]);

}

// GET: Category

public DataSet GetAllCategories()

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spGetAllCategories", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return ds;

}

public HttpResponseMessage GetCategoryById(int id)

{

DataSet ds = new DataSet();

SqlDataAdapter adapter = new SqlDataAdapter("spGetCategoryById", this.connection);

adapter.SelectCommand.CommandType = CommandType.StoredProcedure;

adapter.SelectCommand.Parameters.Add("@CategoryID", SqlDbType.Int);

adapter.SelectCommand.Parameters[0].Value = id;

this.connection.Open();

adapter.Fill(ds);

this.connection.Close();

return Request.CreateResponse(HttpStatusCode.OK, ds);

}

}

}

3.1.5.5 Index home stranica

Index home stranica je početna stranica komponenti javascripta koje se pozivaju u zavisnosti na korisnost aplikacije.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <div class="meni"></div>

    <div class="pageImage"></div>

    <div class="izmena"></div>

    <div class="ubacaj"></div>

    <div class="mainContent"></div>

    <div class="footer"></div>

    <script src="app.js" type = "text/javascript" ></script>

    <script src="/jquery/jquery-3.6.3.min.js" rel="stylesheet"></script>

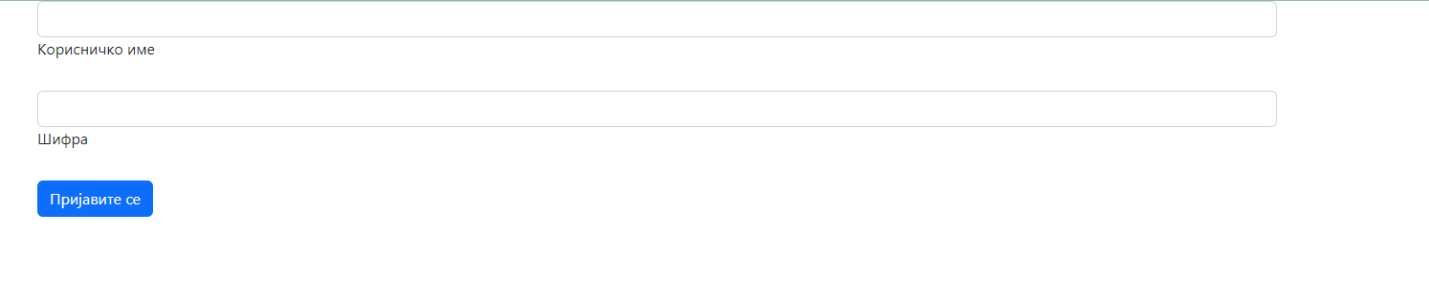
    <script src="ubacaj.js"></script>

    <script src="update.js"></script>

</body>

</html>

## 3.2 Korisnički prikaz aplikacije



Imamo login u trenutno kada ubacimo dobre kredencijale za login, dobijamo homepage kao osnovni prikaz.

Graphical user interface

Description automatically generated

Imamo priliku da izaberemo kategorije useva, takodje klikom na polje masine imamo priliku da izaberemo pojedinacnu masinu I pristupimo osnovnim informacijama o njima.

Druge 2 stranice su izmena I ubačaj novih mašinaa. Dok takođe imamo search opciju gore.

A screenshot of a video game

Description automatically generated with medium confidence