Web aplikacija za rješavanje sustava jedne linearne i jedne kvadratne jednadžbe

Katić, Marko

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Stručni studij

Web aplikacija za rješavanje sustava jedne linearne i jedne kvadratne jednadžbe

Završni rad

Marko Katić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za završni ispit na preddiplomskom stručnom studiju

Osijek, 19.09.2022.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za završni ispit na preddiplomskom stručnom studiju

Ime i prezime Pristupnika:	Marko Katić
Studij, smjer:	Stručni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	AR 4768, 20.09.2019.
OIB Pristupnika:	31366627637
Mentor:	Ivan Hrehorović, prof.
Sumentor:	izv. prof. dr. sc. Josip Job
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	doc. dr. sc. Tomislav Rudec
Član Povjerenstva 1:	Ivan Hrehorović, prof.
Član Povjerenstva 2:	Anja Šteko, mag. educ. math. et inf.
Naslov završnog rada:	Web aplikacija za rješavanje sustava jedne linearne i jedne kvadratne jednadžbe
Znanstvena grana završnog rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rada	Napraviti web aplikaciju koji će dati rješenje sustava linearne i kvadratne jednadžbe. Prikazati grafički. Sumentor s FERIT-a:Josip, Job
Prijedlog ocjene pismenog dijela ispita (završnog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	19.09.2022.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 06.10.2023.

Ime i prezime studenta:	Marko Katić
Studij:	Stručni prijediplomski studij Računarstvo
Mat. br. studenta, godina upisa:	AR 4768, 20.09.2019.
Turnitin podudaranje [%]:	9

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za rješavanje sustava jedne linearne i jedne kvadratne jednadžbe**

izrađen pod vodstvom mentora Ivan Hrehorović, prof.

i sumentora izv. prof. dr. sc. Josip Job

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

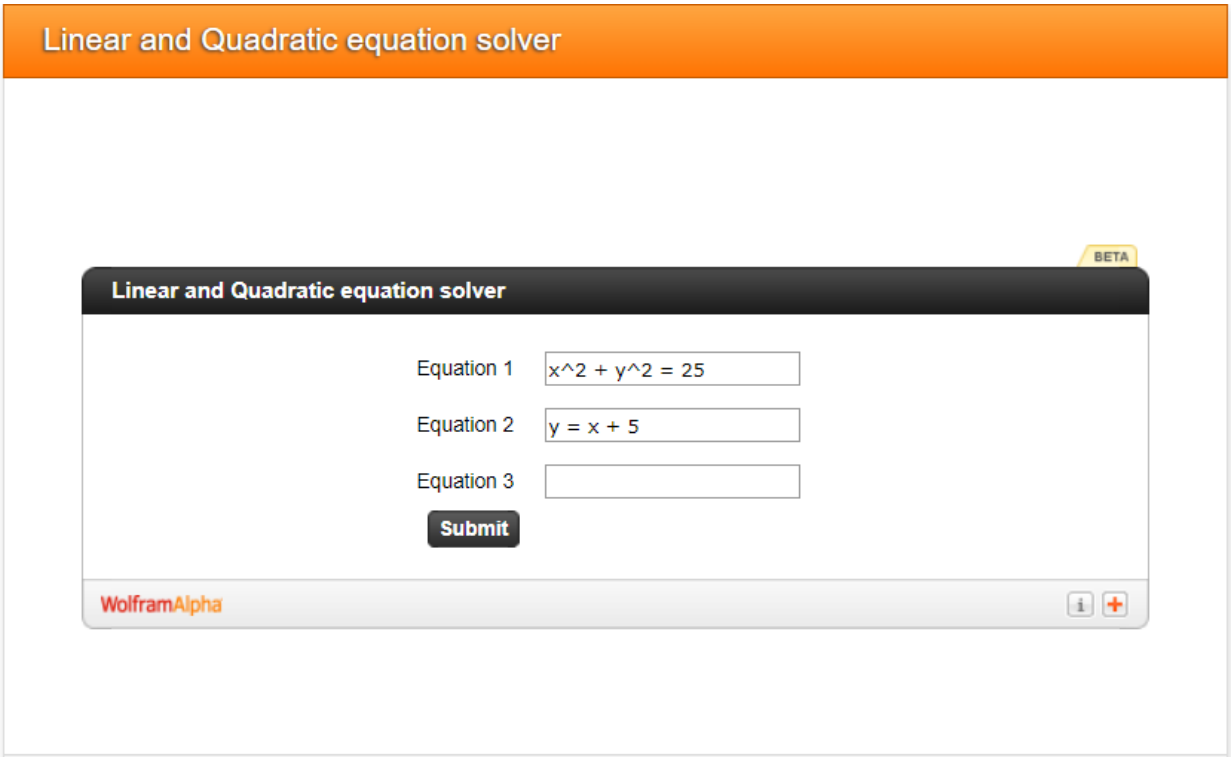
1. UVOD

Ako želimo napraviti aplikaciju koja ne zauzima puno memorije, moguće ju je napraviti kao web aplikaciju. Razvijanje korisničkog sučelja za web aplikaciju vrlo je jednostavno i brzo te zbog toga postoje brojne web aplikacije koje olakšavaju različite zadatke. Cilj ovog rada je napraviti aplikaciju za rješavanje sustava jedne linearne i kvadratne jednadžbe. Aplikaciju možemo napraviti kao web aplikaciju s obzirom na to da takve aplikacije ne zauzimaju puno memorije. Kvadratna jednadžba ima oblik $ax^2 + by^2 + cx + dy + e = 0$, a linearna jednadžba ima oblik $ax + by + c = 0$. Aplikacija ima dva polja za unos jedne linearne i jedne kvadratne jednadžbe. Nepoznanice možemo zapisati koristeći bilo koje slovo abecede. Uvjet za izračunavanje je da mora biti jedna ili dvije nepoznanice u obje jednadžbe. Ako želimo kvadrirati nepoznanicu x, u polje zapisujemo x2. Nakon rješavanja sustava, točke sjecišta zapisuju se u polje za rješenja te se crtaju grafovi na koordinatnom sustavu. U poglavlju 2 prikazane su dvije aplikacije koje imaju mogućnost računanja sustava jedne linearne i jedne kvadratne jednadžbe. Aplikacije imaju i druge mogućnosti tako da nisu u potpunosti prilagođene za računanje sustava jedne linearne i jedne kvadratne jednadžbe. U poglavlju 3 objašnjeni su koraci rješavanja sustava jedne linearne i jedne kvadratne jednadžbe. Metoda koja je korištena za rješavanje sustava je supstitucija. U poglavlju 4 opisani su korišteni alati i tehnologije te je objašnjen kod aplikacije. Od alata korišteni su Gitlab i Visual Studio Code. Korištene tehnologije su HTML, CSS i JavaSctipt. U poglavlju 5 prikazan je dizajn aplikacije koji nema nedostatke koji su navedeni u drugom poglavlju.

1.1. Zadatak završnog rada

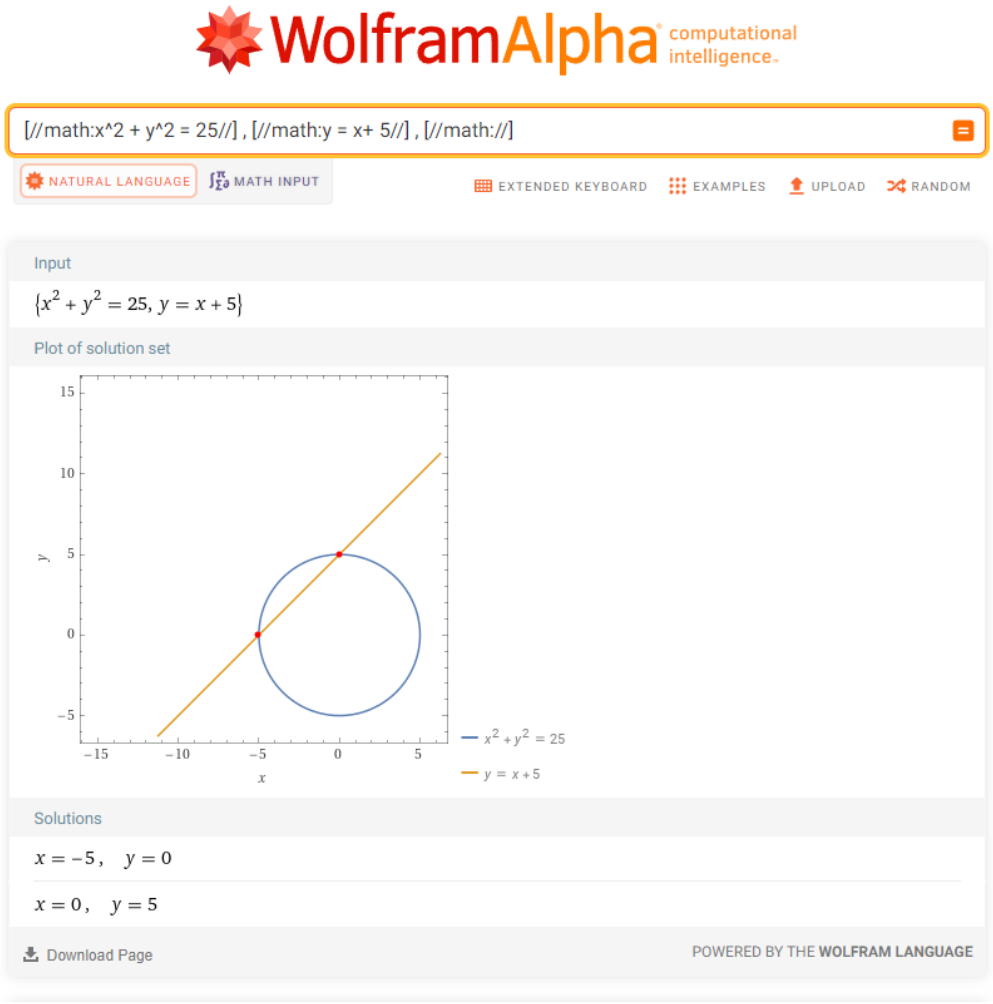
Potrebno je napraviti web aplikaciju za računanje sustava jedne linearne i jedne kvadratne jednadžbe koja osim što daje rješenje sustava, daje i grafički prikaz jednadžbi.

2. Pregled sličnih aplikacija



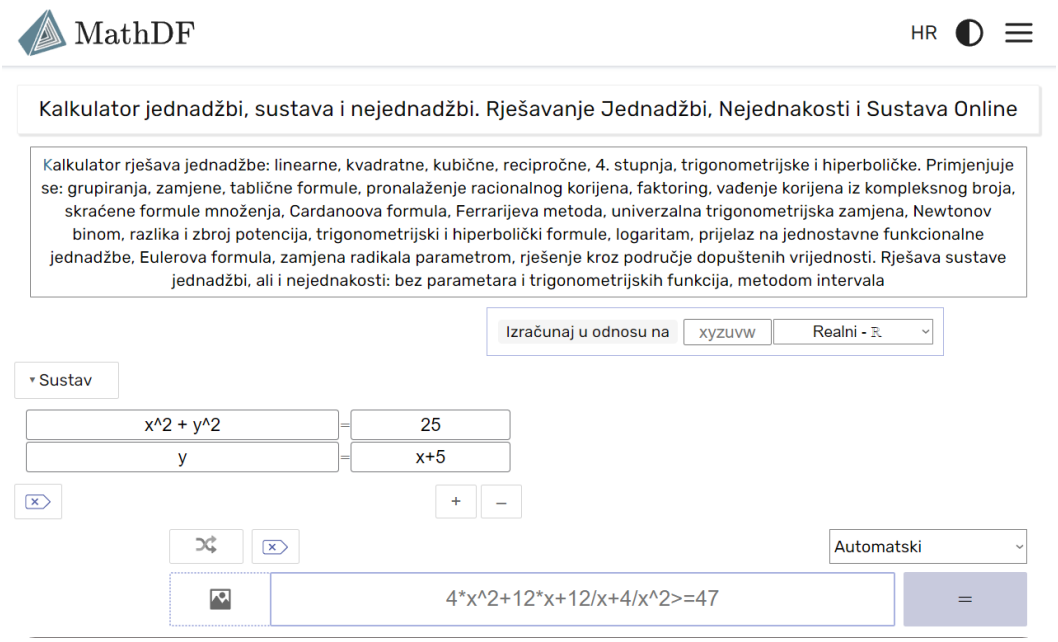
Slika 2.1 WolframAlpha

Sa slike 2.1 vidimo WolframAlpha aplikaciju. Aplikacija koristi jezik Wolfram koji je nastao iz jezika Mathematica [1]. Aplikacija ima mogućnost rješavanja tri jednadžbe. Nakon pritiska gumba Submit, stranica nas preusmjerava na drugu stranicu te je zbog toga potrebno malo čekati. Stranica na koju smo preusmjereni možemo vidjeti iz slike 2.2.



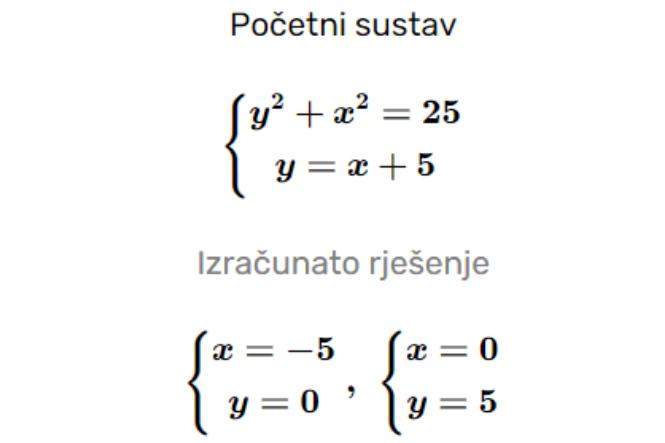
Slika 2.2 stranica sa rješenjem

Na stranici se ispisuje rezultat sustava i grafički se prikazuje rješenje. Stranica ima jedno polje za ponovno upisivanje jednadžbi. Nedostatak ove stranice je taj što ima samo jedno polje za upis novih jednadžbi i grafički prikaz ostavlja prostora za poboljšanje.



Slika 2.3 MathDF

Iz slike 2.3 vidimo aplikaciju MathDF. Aplikacija Ima mogućnost rješavanja sustava jednadžbi sa dvije nepoznanice. Aplikacija ima četiri polja za unos jednadžbi. Nakon pritiska gumba za izračunavanje sustava dobivamo rezultat bez preusmjeravanja na drugu stranicu.



Slika 2.4 rješenje koje je prikazao MathDF

Na slici 2.4 prikazan je rezultat koji je dala aplikacija MathDF. Nedostatak aplikacije je ne prikazuje jednadžbe grafički.

3. RJEŠAVANJE SUSTAVA LINEARNE I KVADRATNE JEDNADŽBE

Za rješavanje sustava linearne i kvadratne jednadžbe potrebno je znati sva pravila . Sustav možemo riješiti pomoću metode supstitucije tako da iz linearne jednadžbe izlučimo x ili y te u kvadratnoj jednadžbi nepoznanicu koju smo izlučili u linearnoj jednadžbi zamijenimo sa drugim dijelom linearne jednadžbe [2,3,4,5]. Kvadratna jednadžba ima oblik:

$$ax + by + cx^2 + dy^2 + e = 0. \tag{3-1}$$

Linearna jednadžba ima oblik:

$$fx + hy + i = 0. \tag{3-2}$$

Ako izlučimo x iz linearne jednadžbe (3-2), dobit ćemo:

$$x = \frac{-hy - i}{f}. \tag{3-3}$$

U jednadžbu (3-1) uvrstimo jednadžbu (3-2) te dobivamo:

$$a\left(\frac{-hy - i}{f}\right) + by + c\left(\frac{-hy - i}{f}\right)^2 + dy^2 + e = 0. \tag{3-4}$$

Kada sredimo jednadžbu (3-4) dobivamo:

$$\left(\frac{ch^2}{f^2} + d\right) \cdot y^2 + \left(b - \frac{ah}{f} + 2hci\right) \cdot y + \left(e - \frac{ia}{f} + \frac{i^2 c}{f^2}\right) = 0. \tag{3-5}$$

Jednadžbu (3-4) možemo riješiti pomoću formule:

$$y_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{3-6}$$

Gdje je koeficijent *a* jednak prvoj zagradi iz jednadžbe (3-5), koeficijent *b* drugoj zagradi i koeficijent *c* jednak trećoj zagradi. Moguće je da koeficijent a bude 0, što znači da ima samo jedno sjecište te u tom slučaju rješenje možemo dobiti formulom:

$$y = -\frac{c}{b}. \tag{3-7}$$

Ukoliko bi rješavali sustav tako da izlučimo y iz linearne jednadžbe, dobili bismo:

$$y = \frac{-fx - i}{h}. \tag{3-8}$$

Nakon uvrštavanja jednadžbe (3-8) u jednadžbu (3-1), dobivamo:

$$ax + b\left(\frac{-fx-i}{h}\right) + cx^2 + d\left(\frac{-fx-i}{h}\right)^2 + e = 0. \quad (3\text{-}9)$$

Nakon sređivanja jednadžbe (3-9), dobivamo:

$$\left(\frac{f^2d}{h^2} + c\right) \cdot x^2 + \left(\frac{2dfi}{h^2} - \frac{fb}{h} + a\right) \cdot x + \left(e + \frac{i^2d}{h^2} - \frac{fi}{h}\right) = 0. \quad (3\text{-}10)$$

Rješenja dobivamo formulom:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \quad (3\text{-}11)$$

U aplikaciji moramo imati obje metode u slučaju da kvadratna jednadžba ima samo jednu nepoznanicu. U tom slučaju pogledamo koju nepoznanicu imamo, *x* ili *y* te biramo metodu. S obzirom na to da se u aplikaciji jednadžbe crtaju u koordinatni sustav, moramo imati i formule za dobivanje koordinata. Formule dobivamo tako da izlučimo *x* i *y* nepoznanice. Formula za dobivanje *x* vrijednosti pomoću *y* vrijednosti je:

$$cx^2 + ax + (by + dy^2 + e) = 0. \quad (3\text{-}12)$$

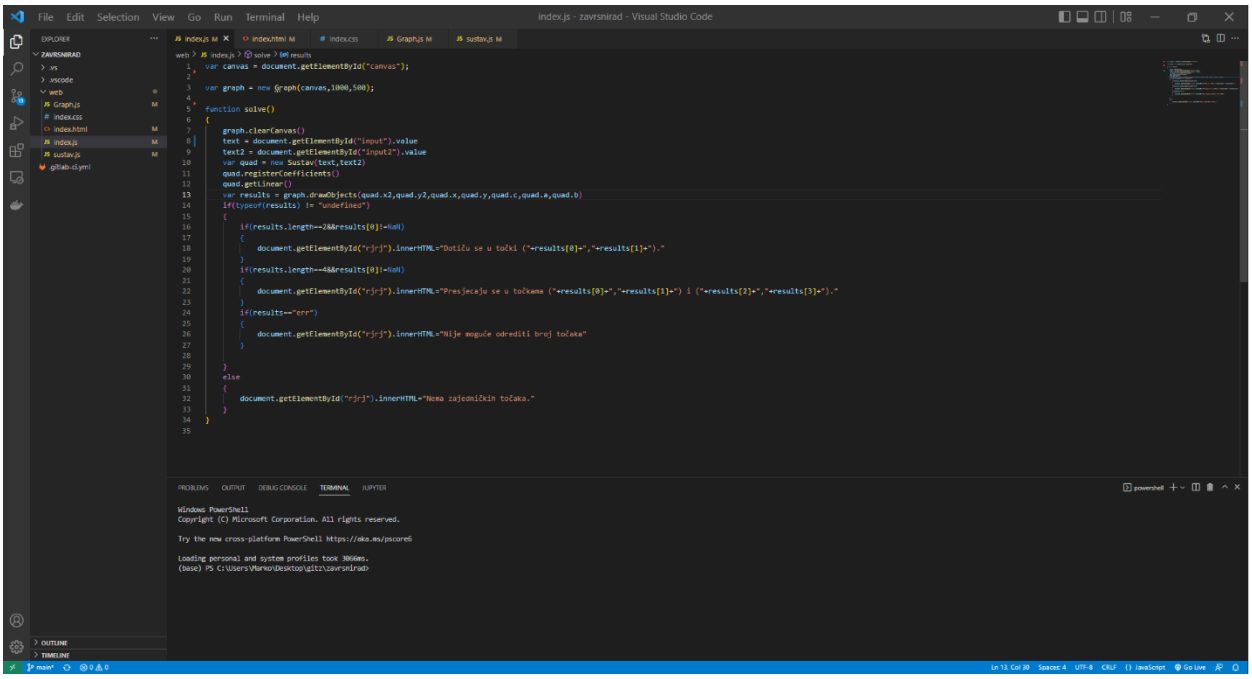
Formula za dobivanje *y* vrijednosti pomoću *x* vrijednosti je:

$$dy^2 + by + (ax + cx^2 + e) = 0. \quad (3\text{-}13)$$

Jednadžbe (3-12) i (3-13) rješavaju se pomoću formula (3-6) i (3-11). Krivulja se crta na grafu tako da se napravi puno kratkih linija. Ovisno o obliku krivulje, nekada je boje sakupljati uzorke tako da prolazimo po x osi, a nekada po y osi. Iz tog razloga obje jednadžbe, (3-12) i (3-13) koriste se u aplikaciji. Formule za dobivanje *x* i *y* vrijednosti linearne jednadžbe su formule (3-3) i (3-8).

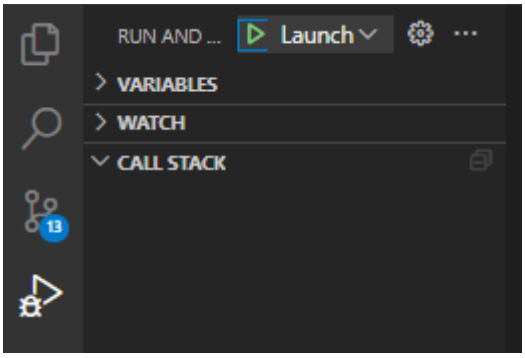
4. KORIŠTENI ALATI, TEHNOLOGIJE I OBJAŠNJENJE KODA

4.1. Visual Studio Code



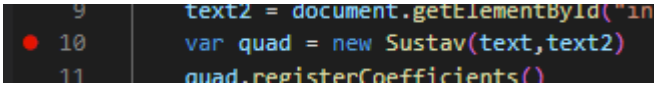
Slika 4.1 Visual Studio Code

Visual Studio Code (slika 4.1) je uređivač koda kojeg je napravio Microsoft za Windows, Linux i macOS operacijski sustav. Na Visual Studio Code moguće je instalirati različite dodatke kao što je live server što ubrzava razvoj web aplikacije. Sa live serverom pohranjene promjene možemo odmah vidjeti u pregledniku bez osvježavanja stranice. Visual Studio Code ima debuger pomoću kojega lakše i brže možemo pronaći pogrešku u kodu. Debuger pokrećemo tako da kliknemo na treću opciju u listi koja se nalazi na lijevoj strani programa te je potrebno kliknuti Launch (slika 4.2).



Slika 4.2 Pokretanje debugera

Nakon što je pokrenut debuger, program će se zaustaviti kada dođe do linije koja je označena kao brakepoint (slika 4.3).



Slika 4.3 Brakepoint

Kada se program zaustavi na označenoj liniji koda možemo pogledati sve vrijednosti varijabli do te linije te tako zaključiti gdje je problem ako ga ima.

4.2. Gitlab

GitLab je platforma za upravljanje kodom i alat za kolaboraciju koji omogućuje timovima razvojnih inženjera da zajedno rade na projektima, upravljaju izvorima koda, praćenju promjena, testiranju i isporuci aplikacija**.** S obzirom na to da GitLab nudi usluge isporuke aplikacije, ova aplikacija se isporučuje putem GitLab-a.

4.3. HTML

HTML (Hyper Text Markup Language) je osnovni jezik za izradu web stranica. Koristi se za definiranje strukture sadržaja web stranica pomoću različitih elemenata. HTML se razvijao tijekom godina, a neke od ključnih verzija su HTML 4.01 i HTML5. HTML 4.01 predstavljen je 1999. godine i donio je brojne poboljšane značajke u usporedbi s prethodnim verzijama. HTML5 predstavljen je 2014. godine, donosi brojne nove elemente i atribute, uključujući audio i video elemente, poboljšane forme i bolju podršku za mobilne uređaje. HTML ima 6 vrsta elemenata, a to su: Void elements, The template element, Raw text elements, Escapable raw text elements, Foreign elements i Normal elements [6]. Elementi imaju svoj početni i završni „tag“. Početak se

označava tako da između uglatih zagrada upisujemo naziv elementa. Kraj se označava isto kao i početak osim što se dodaje kosa crta prije naziva elementa. Void elements nemaju završni tag nego samo početni [7].

```
<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>Završni rad</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="title">
        <h1>Sustav jednadžbi</h1>
    </div>
    <div class="mainWrap">
        <div class="mainDiv">
            <label for="linearEquation">Linearna jednadžba:</label><br>
            <input type="text" name="linearEquation" id="linearEquationInput" class="equationInput"><br><br>
            <label for="quadraticEquation">Kvadratna jednadžba:</label><br>
            <input type="text" name="quadraticEquation" id="quadraticEquationInput" class="equationInput"><br><br>
            <input type="button" value="Izračunaj" id="calculateButton" onclick="calculate()"><br><br>
            <div class="result" id="result"></div>
        </div>

        <div class="graphWrap">
            <div class="graph">
                <div class="box" id="box1">
                    <button class="graphButton" onclick="zoomIn()">+</button>
                    <div class="zoomDiv">Zoom</div>
                    <button class="graphButton" onclick="zoomOut()">-</button></div>
                <div class="box"><button onclick="moveUp()" id="buttonUp" class="graphButton"></div>
                <div class="box"></div>
                <div class="box"><button onclick="moveLeft()" id="buttonLeft" class="graphButton"></div>
                <div class="box"> <canvas id="canvasId" width="600" height="600"></canvas></div>
                <div class="box"><button onclick="moveRight()" id="buttonRight" class="graphButton"></button></div>
                <div class="box"></div>
                <div class="box"><button onclick="moveDown()" id="buttonDown" class="graphButton box"></div>
                <div class="box"></div>
            </div>
        </div>
    </div>



    <script src="graph.js"></script>
    <script src="index.js"></script>
    <script src="linearEquation.js"></script>
    <script src="quadraticEquation.js"></script>
    <script src="equationSystem.js"></script>
</body>
</html>
```

Slika 4.4 HTML dokument

Slika 4.4 prikazuje HTML dokument aplikacije iz kojeg možemo vidjeti sve elemente sa njihovim klasama i id-evima. Elementi pod nazivom div korišteni su za grupiranje elemenata kao što su button i canvas. Svaki element pod nazivom button ima “onclick” koji poziva JavaScript funkciju. Iz naziva funkcija možemo zaključiti šta će koji gumb raditi. Element canvas je za grafički prikaz. Na kraju HTML dokumenta vidimo script elemente koji se koriste kako bi web preglednik znao koji JavaScript dokumenti su mu potrebni. Preglednik treba znati koji JavaScript dokumenti mu trebaju ali i koji CSS dokumenti. CSS je dodan na 7. liniji.

4.4. CSS

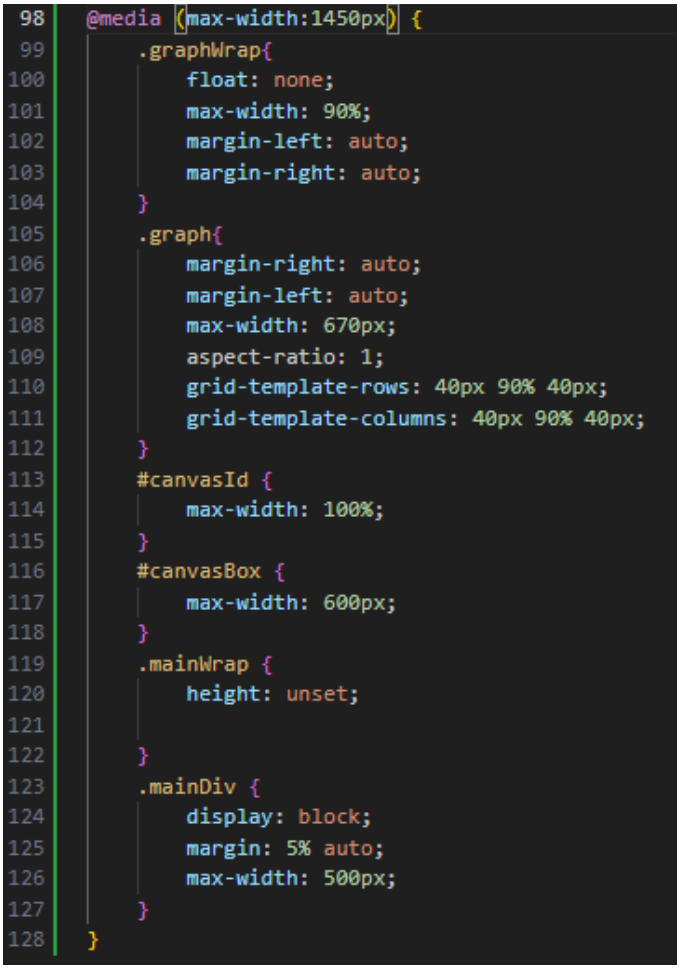
CSS (Cascading Style Sheets) je stilski jezik koji se koristi za opis prezentacije HTML dokumenta. CSS-om se uređuje izgled i raspored stranice. Svakom elementu u HTML dokumentu možemo definirati stilske atribute kao što su širina, visina, boja pozadine, boja teksta, margine, sjena... Elementima u HTML-u možemo zadati id i klasu. Ukoliko zadajmo atribute pomoću id-a samo jedan element sa tim id-om će imati definirani stil, a ako zadajemo pomoću klase ona će svi elementi te klase imati taj stil[8]. CSS-om možemo dodati i razne animacije, tako da korisničko iskustvo bude bolje [9].

```css
.graphButton{
    background-color: rgba(150,150,150,1);
    border: none;
    width: 20px;
    height: 20px;
    border-radius: 5px;
}
.graphButton:hover{
    background-color: rgba(150,150,150,0.7);
}
.graph {
    position: relative;
    display: grid;
    grid-template-rows: 40px 550px 40px;
    grid-template-columns: 40px 550px 40px;
}
.graphWrap{
    float: right;
    margin-right: 50px;
    margin-left: 50px;

}
.box {
  display: flex;
  align-items: center;
  justify-content: center;
}
.result{
    width: 460px;
    height: 360px;
    border: solid 1px gray;
    border-radius: 5px;
    padding: 20px;
    font-size: 24;
    max-width: calc(100% - 40px);
}
.equationInput{
    width: 500px;
    height: 40px;
    border-radius: 5px;
    border: solid 1px gray;
    font-size: 20;
    margin-top: 5px;
    max-width: 100%;
}
.equationInput:focus-visible {
    border: solid 2px rgb(150,150,150);
    outline: none;
}
```

Slika 4.5 CSS

```css
.mainDiv{
    position: relative;
    display: inline-block;
    margin: 50px auto 0 80px;
    max-width: 90%;
}

.title{
    margin: 20px;
    border-radius: 30px;
    padding: 30px;
    background-color: #BBB;
    color: rgb(161, 14, 14);
}

.mainWrap{
    padding: 20px;
    margin: 20px;
    border-radius: 30px;
    height: 700px;
    border: solid 1px #BBB;

}

#calculateButton{
    width: 100px;
    height: 40px;
    border-radius: 15px;
    background-color: rgb(161, 14, 14);
    border: none;
    color: white;
    font-weight: 600;
    font-size: 18;
}
label{
    font-size: 20;
    font-weight: 600;
}
.zoomDiv{
    display: inline-block;
    margin: 0 4px 0 4px;
}

#box1{
    z-index: 0;
    width: 90px;
}
```

Slika 4.6 CSS



Slika 4.7 CSS

Na slikama 4.5, 4.6 i 4.7 možemo vidjeti CSS od aplikacije, koji nam prikazuje koja su sve svojstva dodana elementima iz HTML dokumenta. Ako želimo dodavati svojstva elementima sa određenom klasom, stavljamo točku prije imena klase, a ako želimo dodavati svojstva elementu sa određenim id-om, ispred imena id-a stavljamo znak #.

4.5. JavaScript

JavaScript je programski jezik koji se kompilira u trenutku izvršavanja. Omogućava programerima da manipuliraju sa HTML-om, reagiraju na događaje korisnika i komunikaciju sa web serverima. Iako je najpoznatiji kao skriptni jezik za web stranice, koriste ga i mnoga okruženja bez preglednika, kao što su Node.js, Apache CouchDB i Adobe Acrobat. Osnovne komponente

JavaScripta uključuju: varijable, funkcije i objekte. Varijable se koriste za pohranu podataka kao što su brojevi, nizovi i tekst. Funkcije su blokovi koda koji se mogu izvršavati kad god su potrebni i obično izvršavaju određene radnje. JavaScript koristi objekte za organizaciju podataka i funkcionalnosti [10]**.** Kao što smo vidjeli iz HTML dokumenta, aplikacija koristi se više JavaScript dokumenata. Postoje 4 klase, a to su LinearEquation, QuadraticEquation, EquationSystem i Graph. LinearEquation i QuadraticEquation klase imaju sve potrebne metode za pronalaženje parametara iz prilagođenog teksta koji je linearna ili kvadratna jednadžba te imaju metode za izračunavanje nepoznanica ako predamo vrijednost druge nepoznanice. EquationSystem klasa je klasa koja prima objekte LinearEquation i QuadraticEquation klase. EquationSystem koristimo kako bismo dobili rješenja sustava ovih jednadžbi. Graph klasa služi nam za crtanje po elementu canvas.

```
1   class LinearEquation{
2       equationString = "";
3       equationValues = [[0,"x"],[0,"y"],[0,""]];
4
5       constructor(equationString){
6           this.equationString = equationString;
7           this.setVariables();
8       }
9
10  >    getY(x){ ···
16      }
17  >    getX(y){ ···
23      }
24
25  >    getEquationValues(){ ···
27      }
28
29  >    setVariables() { ···
59      }
60  }
```

Slika 4.8 LinearEquation klasa

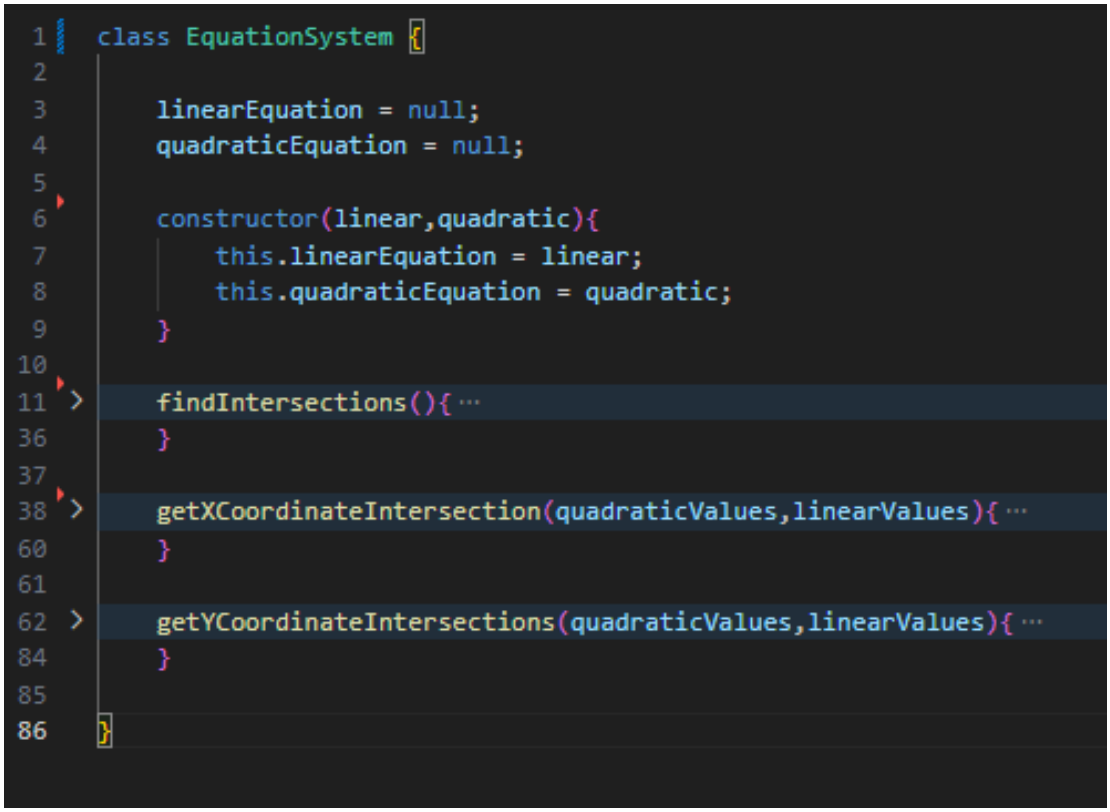
Sa slike 4.8 možemo vidjeti koje metode ima klasa LinearEquation. Metoda getY prima vrijednost x nepoznanice te izračuna kolika je vrijednost y nepoznanice. Metoda getX radi isto kao i getY osim što vraća vrijednost x nepoznanice. Metoda getEquationValues vraća polje equationValues koje je definirano na 3. liniji koda. Metoda setVariables pronalazi parametre

nepoznanica iz varijable equationString. Metoda setVariables podijeli equationString tekst na polje te prolazi po tom polju. Ako je equationString izgledao ovako : “5x + y = 5” tada će polje izgledati ovako : “5x”, “y”, “5”. Metoda pogleda koje slovo ima na određenim adresama polja te sprema brojčanu vrijednost u equaionValues. Predznak koji će se upisati u equationValues ovisi s koje strane jednakosti je pronađen parametar i koji predznak ima u jednadžbi.

```
1   class QuadraticEquation{
2
3       equationString = "";
4       equationValues = [[0,"x2"],[0,"y2"],[0,"x"],[0,"y"],[0,""]];
5
6       constructor(equationString){
7           this.equationString = equationString;
8           this.setVariables();
9       }
10
11  >    getX(y){ ···
24      }
25
26  >    getY(x){ ···
39      }
40
41  >    getEquationValues(){ ···
43      }
44
45  >    setVariables() { ···
81      }
82
83  }
```

Slika 4.9 QuadraticEquation klasa

Sa slike 4.9 možemo vidjeti koje metode ima QuadraticEquation klasa. Ova klasa ima nekoliko metoda sa istom funkcionalnosti kao i LinearEquation, a to su getX, getY getEquationValues i setVariables. Metode setVariables i getEquationValues su iste metode kao i u LinearEquation klasi, a getX i getY se razlikuju jer se koristi različita formula za izračunavanje nepoznanica.



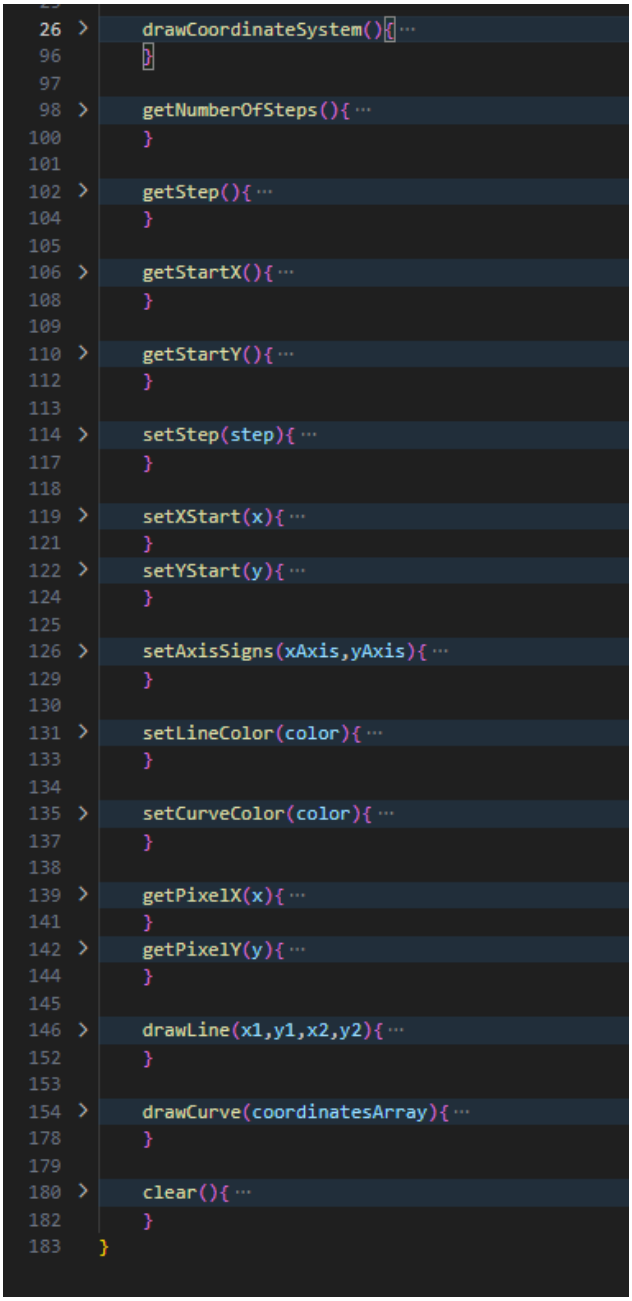
Slika 4.10 EquationSystem klasa

Sa slike 4.10 možemo vidjeti EquationSystem klasu. Metode getXCoordinateIntersection i getYCoordinateIntersection služe nam za dobivanje rješenja sustava linearne i kvadratne jednadžbe. Ove dvije metode razlikuju se po tome što jedna metoda iz linearne jednadžbe izlučuje y a druga metoda izlučuje x nepoznanicu. Metoda findIntersections odlučuje koje od ove dvije metode će se koristiti za rješavanje sustava. Ne možemo izlučiti y iz linearne jednadžbe ako linearna jednadžba nema y nepoznanicu, npr. x = 5 te će se tada koristiti getXCoordinateIntersection.

```
class Graph{

    graph;
    canvas;
    xAxisSign = 'x';
    yAxisSign = 'y';

    step = 1;
    numberOfSteps = 22;
    pixelsPerStep = 0;
    xStart = -10;
    yStart = -10;

    lineColor = "#FF0000";
    curveColor = "#0000FF";


    constructor(canvas){
        this.graph = canvas.getContext('2d');
        this.canvas = canvas;
        this.drawCoordinateSystem();
        this.pixelsPerStep = ((this.canvas.width)/this.numberOfSteps)/this.step;
    }
```

Slika 4.11 Graph klasa

Sa slike 4.11 možemo vidjeti konstruktor i početne vrijednosti klase Graph. Prilikom učitavanja aplikacije crta se koordinatni sustav, tako da trebaju biti postavljene neke početne vrijednosti kao što su vrijednosti koje će biti na x i y osi te slova koja se koriste za označavanje x i y osi.



Slika 4.12 metode Grap klase

Sa slike 4.12 možemo vidjeti koje sve metode ima klasa Graph. Za varijable sa slike 4.11 imamo metode koje vraćaju te vrijednosti i metode koje postavljaju nove vrijednosti. Te metode počinju sa riječi set za postavljanje i get za dobivanje vrijednosti. Metoda drawCoordinateSystem crta koordinatni sustav povlačenjem x i y osi, dodavanjem crta po x i y osi ovisno o tome koliki je broj koraka postavljen (varijabla numberOfSteps), ispisuje nazive x i y osi ovisno o tome što je postavljeno u varijablama xAxisSign i yAxisSign, ispisuje brojeve po x i y osi počevši od varijable

xStart za x os i yStart za y os te svaki korak veći je za varijablu step. Metode getPixelX i getPixelY primaju vrijednosti koje se nalaze na koordinatnom sustavu te vraćaju gdje se nalazi piksel za vrijednost koja je predana u metodu. Te metode se koriste u metodama drawLine i drawCurve. Metoda drawLine prima 4 parametra koje su koordinate od kojih počinjemo povlačiti liniju do kojih završavamo. Metoda drawCurve prima polje koordinata, te će crtati puno malih linija te zbog toga što su te linije toliko male, neće se vidjeti da je to puno malih linije nego će izgledati kao krivulja. Metoda clear briše sve sa elementa canvas. Metoda clear koristi se svaki puta kada se treba crtati novi graf, pomicati po koordinatnom sustavu ili povećati graf.

```
let quadratic = null;
let linear = null;
let graph = new Graph((document.getElementById("canvasId")));
let system = null;
let resultContainer = document.getElementById("result");
let result = null;
function calculate() {

    let linearNonParsed = document.getElementById("linearEquationInput").value;
    let quadraticNonParsed = document.getElementById("quadraticEquationInput").value;

    let linearSigns = findUniqueLetters(linearNonParsed);
    let quadraticSigns = findUniqueLetters(quadraticNonParsed);

    let signs = sortAndMerge(linearSigns,quadraticSigns);

        if(signs.length<=2 && signs.length>0 && linearNonParsed!="" && quadraticNonParsed!=""){
            graph.setAxisSigns(signs[0],signs[1]);

            let linearParsed = linearNonParsed.replaceAll(signs[0],'x');
            linearParsed = linearParsed.replaceAll(signs[1],'y');
            let quadraticParsed = quadraticNonParsed.replaceAll(signs[0],'x');
            quadraticParsed = quadraticParsed.replaceAll(signs[1],'y');


            quadratic = new QuadraticEquation(quadraticParsed);
            linear = new LinearEquation(linearParsed);
            system = new EquationSystem(linear,quadratic);
            result = system.findIntersections();
            if(isQuadraticSolvable()){
                writeResult();
                paint(true);}
            else{
                resultContainer.innerHTML = "Kvadratna jednadžba nije ispravno unešena";
                paint(true);
            }
        }
        else {
            resultContainer.innerHTML = "Unos jednadžbi nije ispravan";
        }
}
```

Slika 4.13 calculate funkcija iz index.js datoteke

Na slici 4.13 prikazana je index.js datoteka i vidimo funkciju calculate koja se pokreće kada se klikne gumb izračunaj. Kada se pokrene funkcija calculate, tekst koji se nalazi u poljima za jednadžbe spremamo u varijable linearNonParsed I quadraticNonParsed. Kroz tekstove prolazimo da vidimo koliko ima različitih slova. Ne smije imati više od 2 slova (najviše dvije nepoznanice). Pronađena slova postavljamo za imena x i y osi. S obzirom na to da QuadraticEquation i LinearEqution klase primaju prilagođene tekstove (nepoznanice moraju biti x i y), zamjenjujemo pronađena slova sa slovima x i y. Riješi se sustav te se ispiše rezultat u polje za rezultat. Nakon ispisivanja rezultata pokrećemo funkciju paint koju možemo vidjeti na slici 4.14.

```
131 function paint(autoPosition){
132     graph.clear();
133     if(!isNaN(result[0][0]) && !isNaN(result[0][1]) && !isNaN(result[1][0]) && !isNaN(result[1][1]) && autoPosition){
134         let step = graph.getStep();
135         let xDiff = Math.abs(result[0][0] - result[1][0]);
136         let yDiff = Math.abs(result[0][1] - result[1][1]);
137         if(xDiff>((graph.getNumberOfSteps()-2)/2)*graph.getStep()*0.9){
138             do{
139                 graph.setStep(graph.getStep()+1);
140             }while(xDiff>((graph.getNumberOfSteps()-2)/2)*graph.getStep()*0.9)
141         }
142         if(yDiff>((graph.getNumberOfSteps()-2)/2)*graph.getStep()*0.9){
143             do{
144                 graph.setStep(graph.getStep()+1);
145             }
146             while(yDiff>((graph.getNumberOfSteps()-2)/2)*graph.getStep()*0.9)
147         }
148         let xCor =  Math.floor((result[0][0] + result[1][0])/2);
149         let yCor =  Math.floor((result[0][1] + result[1][1])/2);
150         graph.setXStart(xCor-((graph.getNumberOfSteps()-2)/2)*graph.getStep());
151         graph.setYStart(yCor-((graph.getNumberOfSteps()-2)/2)*graph.getStep());
152     }
153     graph.drawCoordinateSystem();
154     drawLinear();
155     drawQuadratic();
156 }
157
```

Slika 4.14 paint metoda

Funkcija prvo provjerava ima li sustav rješenja. U slučaju da sustav ima rješenja funkcija će postaviti (prilagoditi će vrijednosti koje se nalaze na x i y osi) sjecište u sredinu koordinatnog sustava ili ako ima dva sjecišta tada će sjecišta biti jednako udaljena od sredine koordinatnog sustava. Crta se koordinatni sustav te onda linearna pa kvadratna jednadžba. Funkciju za crtanje linearne jednadžbe možemo vidjeti sa slike 4.15.

```
158  function drawLinear(){
159
160      let startX = graph.getStartX();
161      let startY = graph.getStartY();
162      let endX = graph.getStartX() + graph.getStep()*(graph.getNumberOfSteps()-2);
163      let endY = graph.getStartY() + graph.getStep()*(graph.getNumberOfSteps()-2);
164
165      if(!isNaN(linear.getY(graph.getStartX()))){
166          graph.drawLine(startX,linear.getY(startX),
167          endX,linear.getY(endX));
168      }
169      else{
170          graph.drawLine(linear.getX(startX), startY,
171          linear.getX(endY),endY);
172      }
173  }
```

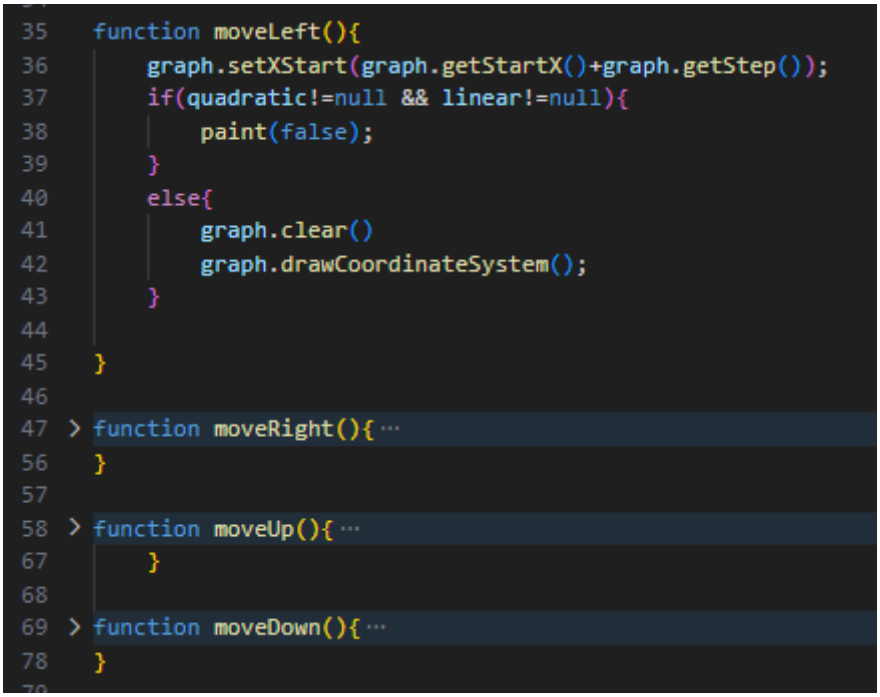
Slika 4.15

Funkcija pronalazi početne i završne točke koordinatnog sustava te pomoću getX i getY metoda klase LinearEquation dobivamo koordinate od kojih se treba započeti crtati do kojih se crta linija. Sa slike 4.16 možemo vidjeti kako radi funkcija za crtanje kvadratne jednadžbe.

```
function drawQuadratic()
{
    let startX = graph.getStartX();
    let startY = graph.getStartY();
    let endX = graph.getStartX() + graph.getStep()*(graph.getNumberOfSteps()-2);
    let endY = graph.getStartY() + graph.getStep()*(graph.getNumberOfSteps()-2);


    let arrayX1 = [];
    let arrayX2 = [];

    let arrayY1 = [];
    let arrayY2 = [];

    let precision = 2000;

    let smallStep = (endX-startX)/precision;

    for(let i = startX; i<endX; i += smallStep){
        let y1 = quadratic.getY(i)[0];
        let y2 = quadratic.getY(i)[1];
        if(y1<endY && y1>startY){
            arrayX1.push([i,y1]);
        }
        if(y2<endY & y2>startY){
            arrayX2.push([i,y2]);
        }
    }
    for(let j = startY; j<endY; j += smallStep){
        let x1 = quadratic.getX(j)[0];
        let x2 = quadratic.getX(j)[1];
        if(x1<endX && x1>startX){
            arrayY1.push([x1,j]);
        }
        if(x2<endX && x2>startX){
            arrayY2.push([x2,j]);
        }
    }

        graph.drawCurve(arrayX1);
        graph.drawCurve(arrayX2);
        graph.drawCurve(arrayY1);
        graph.drawCurve(arrayY2);
}
```

Slika 4.16 drawQuadratic metoda

Funkcija prolazi po x osi koordinatnog sustava i izračunava kolika je y vrijednost. Sakupljene koordinate spremamo u polja. Funkcija također prolazi po y osi i sakuplja x vrijednosti. Nastala polja crtaju se na koordinatni sustav pomoću metoda drawCurve. Rade se prolazi i po x i po y osi jer je ponekad bolje crtati tako da se prolazi po x osi, a ponekad je bolje kada se prolazi po y osi, ovisno o obliku kvadratne jednadžbe.



Slika 4.17

Sa slike 4.17 možemo vidjeti kako rade funkcije za pomicanje po koordinatnom sustavu. Nanovo se postavlja varijabla startX ili startY ovisno da li pomićemo x ili y os i tu varijablu zbrajamo ili oduzimamo sa varijablom step. Nakon postavljanja startX i startY, sve što je nacrtano na elementu canvas se briše i sve se nanovo crta.

```
function zoomIn(){
    if(graph.getStep()>1){
        let newStep = graph.getStep()-1;
        graph.setStep(newStep)
        let numberOfSteps = (graph.getNumberOfSteps()-2)/2;
        graph.setXStart(graph.getStartX()+numberOfSteps);
        graph.setYStart(graph.getStartY()+numberOfSteps);
        if(quadratic!=null && linear!=null){
            paint(false);
        }
        else{
            graph.clear()
            graph.drawCoordinateSystem();
        }
    }
}

function zoomOut(){ ···
    }
```

Slika 4.18

Sa slike 4.18 vidimo kako radi povećavanje i smanjivanje grafa. U metodi zoomIn i zoomOut mijenja se startX i startY i uz to i varijabla step. Najmanji korak može biti 1 a najveći nema ograničenje. Varijable startX i startY će se postaviti tako da u sredini koordinatnog sustava ostanu jednake vrijednosti x i y osi.

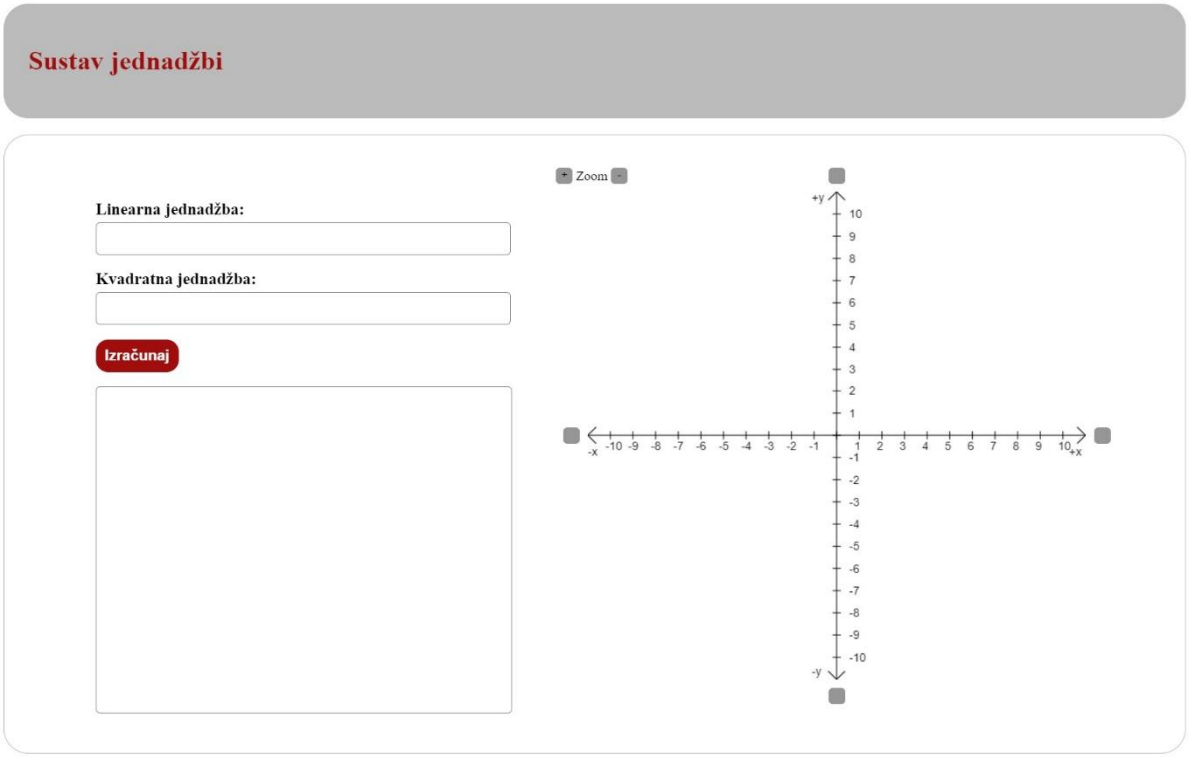
```
function writeResult(){
    if(!isNaN(result[0][0]) && !isNaN(result[0][1]) && !isNaN(result[1][0]) && !isNaN(result[1][1])){
        if(result[0][0]!=result[1][0] || result[0][1] != result[1][1]){
            resultContainer.innerHTML = "Sjeku se u točkama:<br><br>T1("+parseFloat(result[0][0]).toFixed(2)+","
                                        +parseFloat(result[0][1]).toFixed(2)+") i T2("+parseFloat(result[1][0]).toFixed(2)
                                        +","+parseFloat(result[1][1]).toFixed(2)+").";
        }
        else{
            resultContainer.innerHTML = "Sjeku se u točki:<br><br>T("+result[0][0]+","
                                        +result[0][1]+").";
        }
    }
    else
    {
        resultContainer.innerHTML = "Ne sjeku se ni u jednoj točki.";
    }
}
```

Slika 4.19

Sa slike 4.19 vidimo kako radi funkcija za ispisivanje rezultata. Ako su rezultati različiti, funkcija će ispisati da ima dvije točke T1 i T2, ako su rezultati isti, funkcija će ispisati da ima jedno sjecište

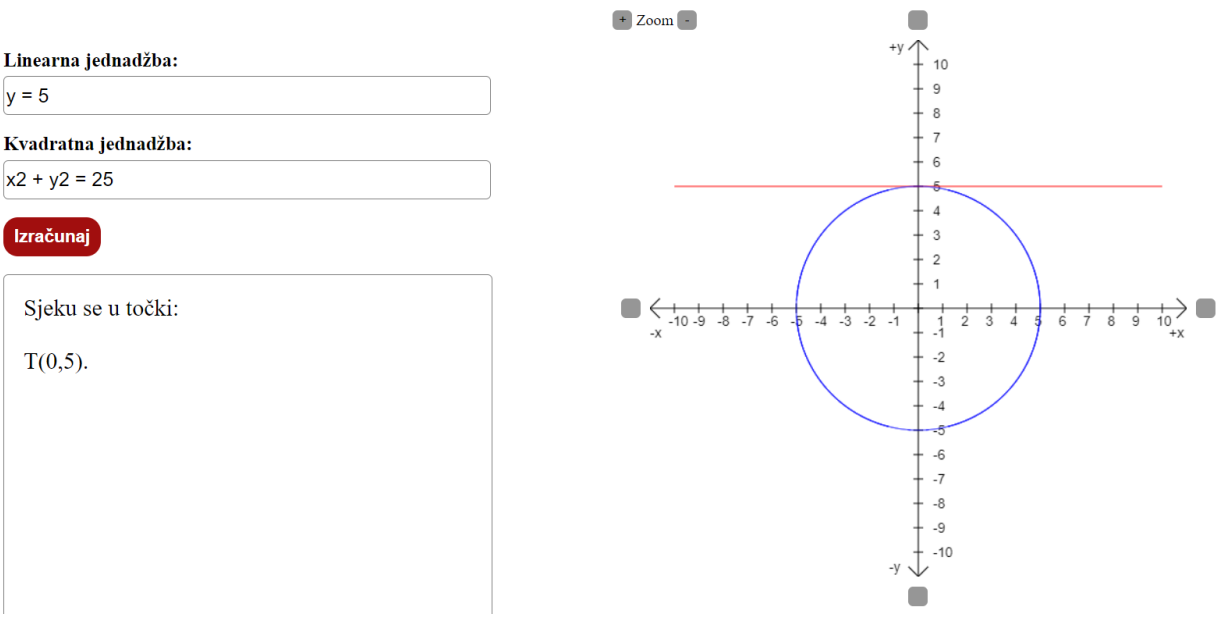
u točki T. Ako nema sjecišta funkcija za rješavanje sustava će ispisati rezultate kao NaN te će funkcija za ispisivanje rezultata otići u else uvjet i ispisati da se ne sijeku ni u jednoj točki.

5. Dizajn i demonstracija aplikacije



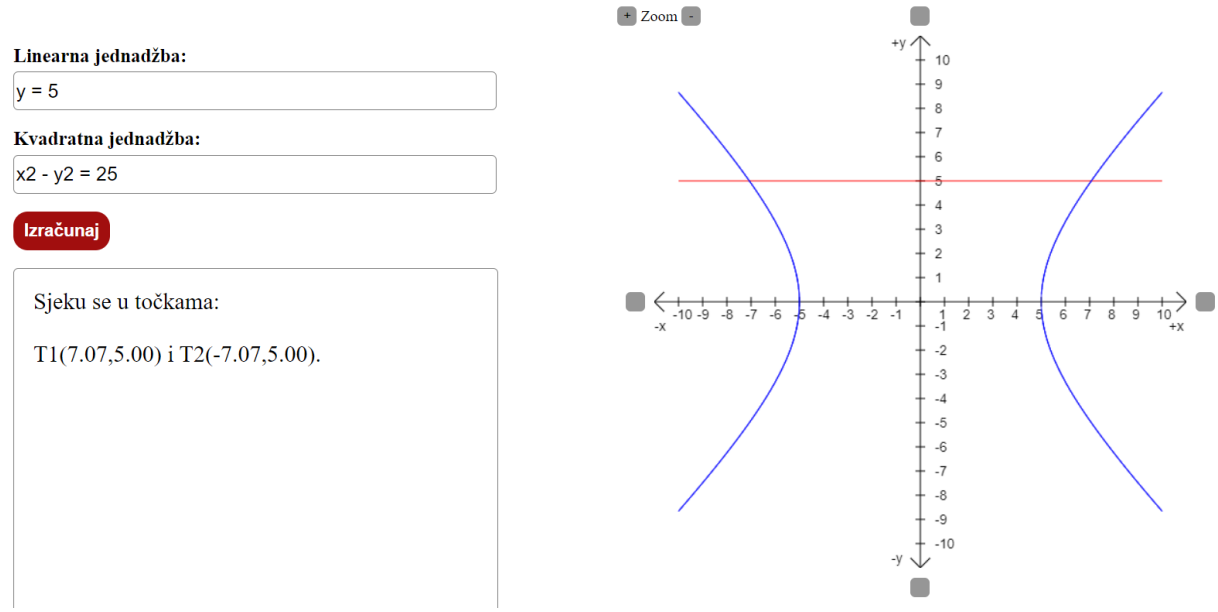
Slika 5.1 dizajn aplikacije

Uzevši u obzir nedostatke prethodno opisanih aplikacija, nastao je dizajn aplikacije ovog rada. Iz slike 5.1 možemo vidjeti kako aplikacija izgleda. Sa lijeve strane su dva polja za upis kvadratne jednadžbe i linearne jednadžbe. U jednadžbama bilo koja dva slova mogu biti nepoznanice. Nepoznanice na kvadrat pišu se tako da se dodaje broj 2 nakon nepoznanice, npr. x2. Nakon pritiska gumba Izračunaj, rješenje sustava se ispisuje u polje ispod gumba izračunaj te se kvadratna i linearna jednadžba prikažu u koordinatnom sustavu. Na svakoj osi koordinatnog ima gumb koji služi za kretanje po koordinatnom sustavu. Postoji i opcija smanjivanja i povećanja prikaza s gumbovima + i – koji stoje u gornjem lijevom kutu koordinatnog sustava.



Slika 5.2

Uz zadanu kvadratnu i linearnu jednadžbu, iz slike 5.2 vidimo da aplikacija daje dobro rješenje sustava te da su jednadžbe nacrtane u koordinatnom sustavu. U ovom slučaju, kvadratna jednadžba se prikazuje grafički kao kružnica. Međutim, ako promijenimo predznak kod člana s kvadratom nepoznanice y, dobit ćemo hiperbolu kao što vidimo iz slike 5.3.



Slika 5.3

6. ZAKLJUČAK

Cilj ovog rada bio je napraviti web aplikaciju za računanje sustava jedne linearne i jedne kvadratne jednadžbe koja osim što daje rješenje sustava, aplikacija daje i grafički prikaz jednadžbi. Pronađene aplikacije imaju nedostatke, tako da je dizajnirana nova aplikacija koja nema te nedostatke. Aplikacija ovog rada ima dva polja za unos jedne linearne i jedne kvadratne jednadžbe. Jednadžbe koje se unose u polja moraju imati ili jednu ili dvije nepoznanice. Računanje sustava pokreće se pritiskom gumba izračunaj. Prije nego li se započne računanje sustava, aplikacija provjera jesu li jednadžbe dobro zapisane te ako jesu, aplikacija rješava sustav. Nakon što je sustav riješen, aplikacija crta jednadžbe u koordinatni sustav. Crtanje jednadžbi u koordinatni sustav najzahtjevniji je dio ovog rada. Aplikacija je brza, točno računa i točno crta. Ograničenje ove aplikacije je to što ne može računati sustav s dvije kvadratne jednadžbe ili sustav sa dvije linearne jednadžbe. Tehnologije koje su korištene za ovaj rad su HTML, CSS i JavaScript. Sustav jedne linearne i jedne kvadratne jednadžbe jednostavan je za rješavanje, ali ljudske pogreške uvijek su moguće. Što je više sustava potrebno izračunati to je veća vjerojatnost pogreške. Iz tog razloga dobro je koristiti ovu aplikaciju za računanje i crtanje grafa ili samo za provjeru rezultata.

LITERATURA

[1] Skupina autora, About WolframAlpha [online], WolframAlpha, dostupno na: https://www.wolframalpha.com/about [2.10.2023.]

[2] D. Bakić, Linearna algebra, Školska knjiga, Zagreb, 2008

[3] D. Bakić, Linearna algebra i primjene, Školska knjiga, Zagreb, 2021.

[4] K. Horvatić, Linearna algebra, Golden marketing, Stručna knjižara, Zagreb, 2004.

[5] N. Elezović, A. Aglić, Linearna algebra, Zbirka zadataka, Element, Zagreb, 1995.

[6] World Wide Web Consortium (W3C) (2017), „HTML5 – A Vocabulary and Associated APIs for HTML and XHTML.“ [online], dostupno na https://www.w3.org/TR/html5/ [2.10.2023.]

[7] J. Duckett, HTML and CSS: Design and Build Websites, John Wiley & Sons, Hoboken, 2011.

[8] Mozilla Developers, "CSS (Cascading Style Sheets) - MDN Web Docs." [Online], dostupno na: https://developer.mozilla.org/en-US/docs/Web/CSS [2.10.2023.]

[9] L. Verou, CSS Secrets: Better Solutions to Everyday Web Design Problems, O'Reilly Media, Sebastopol, 2015.

[10] Mozilla Developers, "JavaScript." [online], dostupno na: https://developer.mozilla.org/en-US/docs/Web/JavaScript [2.10.2023.]

SAŽETAK

Naslov: Web aplikacija za računanje sustava jedne linearne i jedne kvadratne jednadžbe

Cilj ovog rada bio je napraviti web aplikaciju za računanje sustava jedne linearne i jedne kvadratne jednadžbe koja osim što daje rješenje sustava, aplikacija daje i grafički prikaz jednadžbi. Napravljeno je istraživanje te su pronađene dvije aplikacije koje mogu računati sustav jedne linearne i jedne kvadratne jednadžbe. Nedostatci pronađenih aplikacija uklonjeni su u aplikaciji ovog rada. Korišteni alati za izradu ovog radu su Visual Studio Code i Gitlab. Korištene tehnologije za izradu ovog rada su HTML, CSS i JavaScript. Aplikacija je brza, točno računa sustav jedne linearne i jedne kvadratne jednadžbe i grafički prikazuje te jednadžbe na koordinatnom sustavu.

Ključne riječi: funkcije, nepoznanice , sustav, web aplikacija.

ABSTRACT

Title: Web Application for Solving Systems of One Linear and One Quadratic Equation

The aim of this paper was to create a web application for solving systems of one linear and one quadratic equation, which not only provides solutions but also offers a graphical representation of the equations. Research was conducted, and two applications capable of solving systems of one linear and one quadratic equation were identified. The shortcomings of the discovered applications have been removed from application developed in this paper. The tools used for creating this paper were Visual Studio Code and GitLab. The technologies employed for this project include HTML, CSS, and JavaScript. The application is fast, accurately computes systems of one linear and one quadratic equation, and graphically displays these equations on a coordinate system.

Keywords: functions, unknowns, system ,web application.