

Univerzitet Crne Gore
ELEKTROTEHNIČKI FAKULTET
Studije primijenjenog računarstva

Softversko inženjerstvo

-dokumentacija mini-projekta-

Tema: Hangman

Projekat radili:

1. Lejla Martinović 81/21
2. Amer Biso 91/21

Podgorica, 13.06.2025. godine

Kratak opis projekta:

Naš projekat jeste kreiranje igrice Hangman. Igrica je veoma jednostavna, i cilj je pogoditi skrivenu riječ. Modifikacija same igrice jeste mogućnost biranja teme iz koje dolaze riječi, a takođe će se pamti i najbolji skor.

Opis zadataka i doprinosa članova projektnog tima

Lejla Martinović:

Rad na kodu i dokumentaciji.

Amer Biso:

Rad na kodu i dokumentaciji.

Opis opštih funkcionalnosti softvera

Funkcionalni zahtjevi

Req1- Izbor kategorije riječi

Opis:Sistem mora omogućiti korisniku da iz glavnog menija izabere kategoriju riječi (nivo igre) ili da odabere opciju „Sve kategorije“ za slučajan izbor riječi iz bilo koje kategorije.

Req2- Unos imena igrača

Opis:Sistem mora omogućiti korisniku da unese svoje ime prije početka igre. Ukoliko ime nije uneseno, sistem automatski postavlja ime na „anonimno“

Req3-Prikaz riječi i vješala

Opis: Sistem mora omogućiti prikaz skrivene riječi koja se pogađa tako da su nepogađena slova maskirana (sa "_"), te mora prikazivati vješala koja se postepeno dopunjavaju nakon svake greške.

Req4-Pogadjanje slova

Opis: Sistem mora omogućiti korisniku unos slova putem tastature. Nakon unosa, sistem mora provjeriti da li se slovo nalazi u riječi i prikazati rezultat (pogodak/nepogodak). Pogrešna slova se posebno prikazuju.

Req5-Prikaz pogrešnih slova

Opis: Sistem mora voditi evidenciju i prikazati sva pogrešna slova koja je korisnik unio tokom igre.

Req6-Zapis rezultata u fajl

Opis: Sistem mora po završetku igre zabilježiti datum, ime igrača, mod igre i niz tačnih pogodaka u fajl rezultati.txt

Req7-Prikaz najvećeg rekorda

Opis: Sistem mora analizirati podatke iz fajla sa rezultatima i prikazati najveći niz tačnih pogodaka za izabrani mod/kategoriju tokom igre

Req8-Detekcija kraja igre I upravljanje nizom pogodaka

Opis: Sistem mora detektovati kraj igre – bilo pobjedu (kada su pogođena sva slova), ili poraz (kada ponestane pokušaja). Nakon završetka igre, sistem mora povećati ili resetovati niz pogodaka i zapisati rezultat u fajl.

Acceptance test

ACT1 – Izbor kategorije riječi (*REQ1*)

Opis: Testira se izbor određene kategorije ili opcije „Sve kategorije“.

- Igrač izabrao kategoriju iz liste, riječ je uspješno odabrana iz te kategorije.
- Igrač izabrao „Sve kategorije“, sistem je nasumično izabrao riječ iz svih kategorija.
- Igrač nije izabrao ništa – igra se nije pokrenula ili prikazana greška.

ACT2 – Unos imena igrača (*REQ2*)

Opis: Testira se unos korisničkog imena ili fallback na "anonimno".

- Igrač unio ime "Marko", ime je ispravno sačuvano i prikazano tokom igre.
- Igrač nije unio ime, sistem automatski postavio ime "anonimno".
- Igrač unio prazan string, sistem nije reagovao ispravno (nije dodijelio „anonimno“).

ACT3 – Prikaz riječi i vješala (*REQ3*)

Opis: Testira se prikaz riječi (sa maskiranim slovima) i vješala.

- Igrač pokrenuo igru – prikazana riječ (" _ _ _ ") i početna struktura vješala.
- Igrač pokrenuo igru – riječ nije prikazana ili je prazna.
- Igrač pokrenuo igru – vješala nisu prikazana.

ACT4 – Pogadjanje slova (*REQ4*)

Opis: Testira se unos i provjera slova.

- Igrač unio tačno slovo – slovo se otkrilo u riječi.
- Igrač unio pogrešno slovo – dodat je dio vješala.
- Igrač unio bilo koje slovo – rezultat je bio tačan ili pogrešan, sistem reagovao.
- Igrač unio broj ili više slova – sistem prikazuje poruku.

ACT5 – Prikaz pogrešnih slova (*REQ5*)

Opis: Testira se lista pogrešnih slova.

- Igrač unio 3 pogrešna slova, lista je prikazala ta 3 slova.
- Lista se automatski ažurira nakon svakog novog pogrešnog slova.
- Igrač unio pogrešno slovo, ali nije prikazano na listi.

ACT6 – Zapis rezultata u fajl (*REQ6*)

Opis: Testira se zapis rezultata po završetku igre.

- Igra završena, fajl rezultati.txt sadrži novi red sa datumom, igračem, modom i nizom.
- Igrač nije unio ime – fajl sadrži „igrac: anonimno“.
- Fajl nije kreiran ili linija nije upisana nakon završetka igre.

ACT7 – Prikaz najvećeg rekorda (*REQ7*)

Opis: Testira se prikaz maksimalnog niza pogodaka iz fajla.

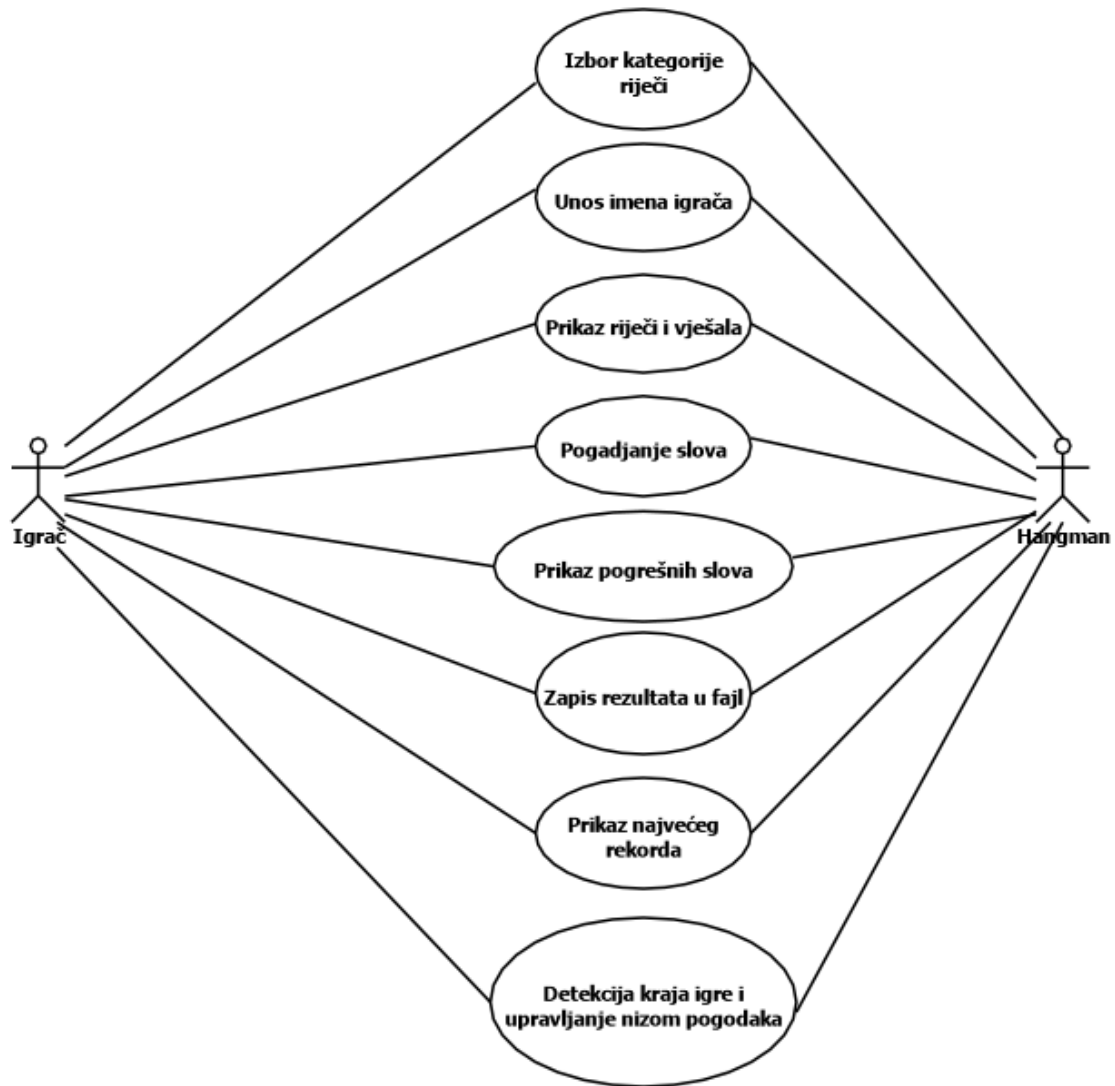
- Na početku igre prikazan najveći niz za izabranu kategoriju.
- Fajl rezultati.txt analiziran, prikazana tačna vrijednost.
- Nema rekorda za tu kategoriju – sistem nije prikazao nulu (default vrijednost).

ACT8 – Detekcija kraja igre i upravljanje nizom pogodaka (*REQ8*)

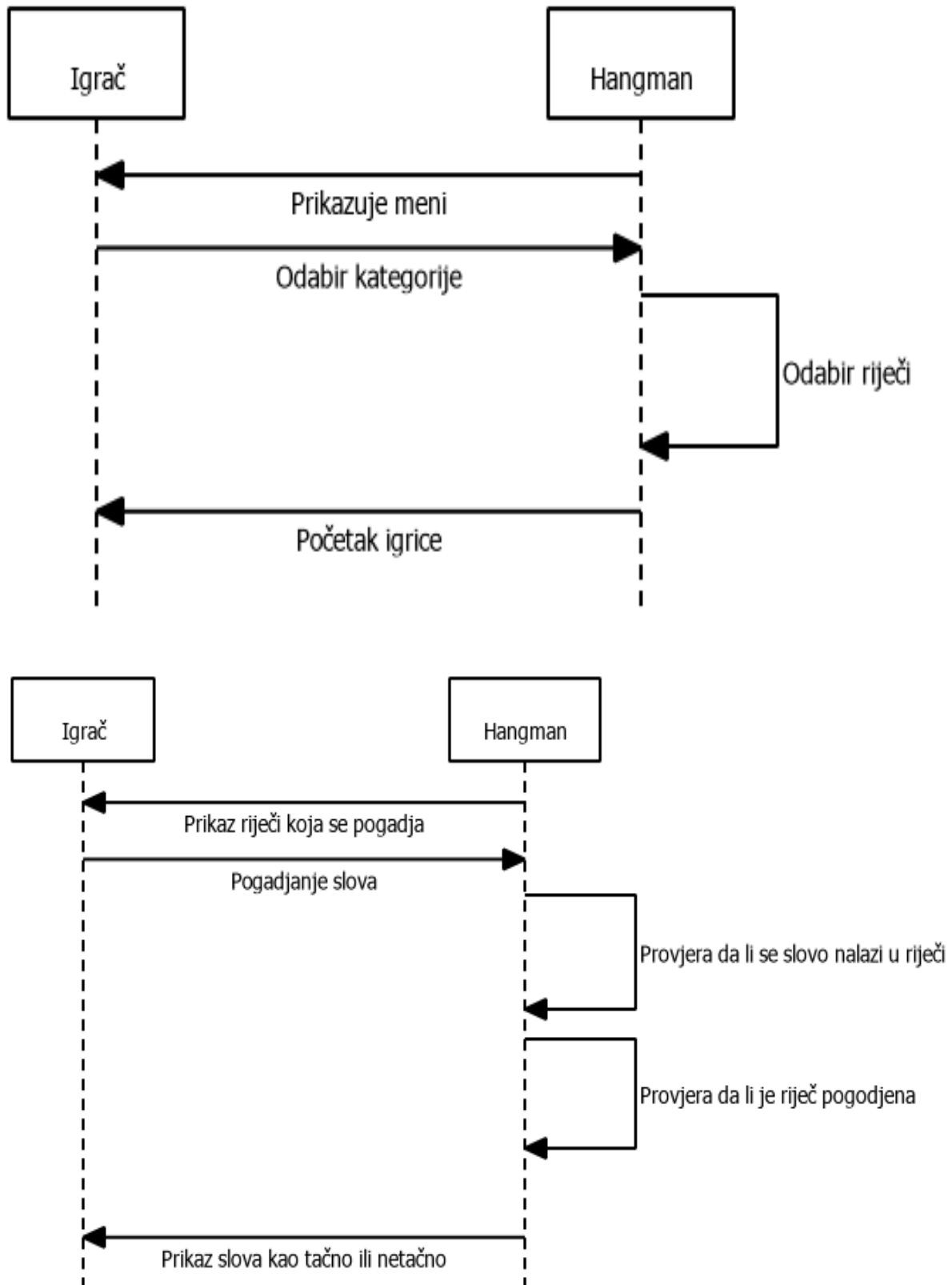
Opis: Testira se logika pobjede/poraza i upravljanje nizom.

- Igrač pogodio sva slova – prikazana poruka pobjede, niz povećan.
- Igrač ostao bez pokušaja – prikazana poruka poraza, niz resetovan.
- Nakon pobjede/poraza – rezultat upisan u fajl.
- Niz nije ažuriran nakon završetka igre.

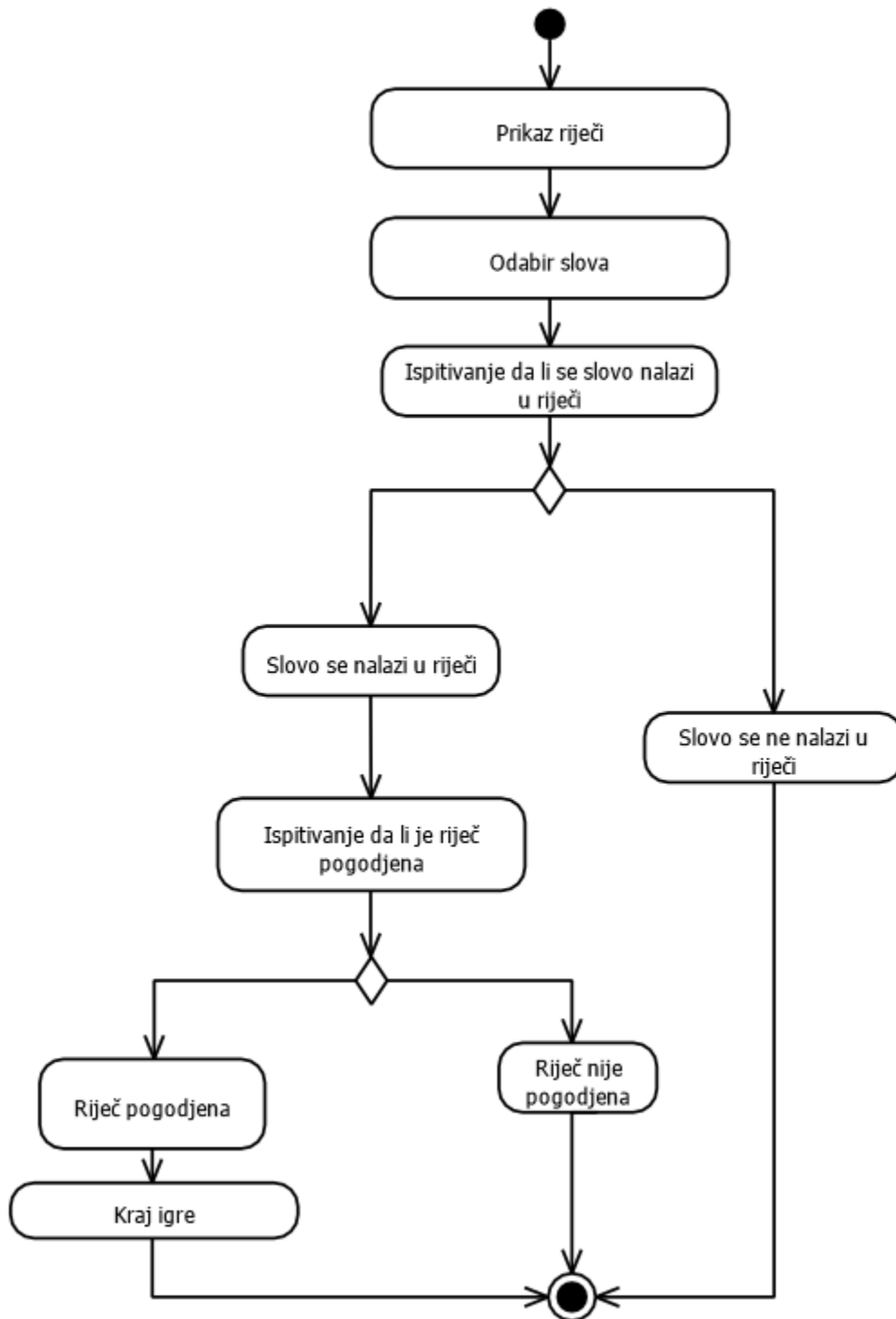
Use Case diagram



Dijagram sekvence



Dijagram aktivnosti

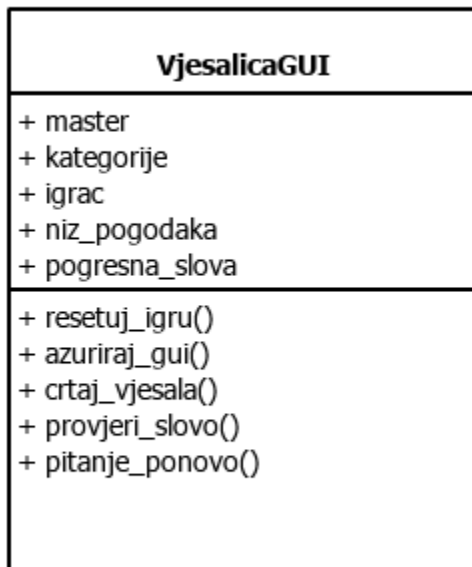


Dijagram klase

Klasa MainMenu




Klasa VjesalicaGUI



Dizajn

Glavni meni

 Vješalica - Meni—□×

Izaberi kategoriju:

Životinje

Unesi ime igrača (opcionalno):

anonimno

Počni igru

Odabir kategorije

Životinje

Životinje

Tehnologija

Sport

Sve kategorije

Tok igre

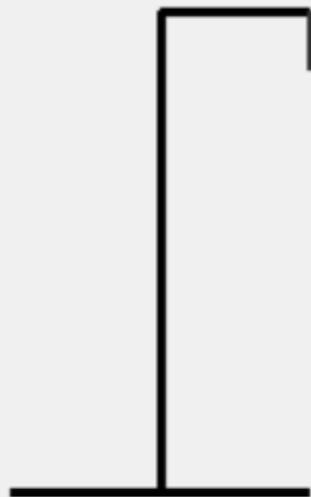


Vješalica - Igra



Kategorija: Životinje

— — —



Preostalo pokušaja: 6

Trenutni niz pogodaka: 0

Najveći rekord za ovaj mod: 5

Pogrešna slova: Nema pogrešnih slova.

Unešeno više slova



Greška



Unesi jedno slovo!

OK

Izgubljeno



Vješalica - Igra



Kategorija: Životinje

_ a _

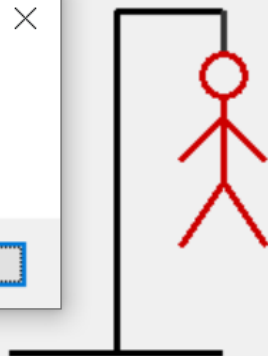


Izgubio si



Riječ je bila: lav

OK



Preostalo pokušaja: 0

Trenutni niz pogodaka: 0

Najveći rekord za ovaj mod: 5

Pogrešna slova: d, e, r, t, u, z

Kod

Ova funkcija čita riječi iz fajla rijeci.txt.

```
def ucitaj_rijeci(ime_fajla):  
    kategorije = {}  
    with open(ime_fajla, 'r', encoding='utf-8') as f:  
        for linija in f:  
            if ':' in linija:  
                kat, rijeci_str = linija.strip().split(':')  
                rijeci = [r.strip() for r in rijeci_str.split(',')]  
                kategorije[kat.strip()] = rijeci  
    return kategorije
```

Bira se riječ za igru:

- Ako je igrač izabrao kategoriju – bira se riječ iz te kategorije.
- Ako je izabrana opcija „Sve kategorije“ – bira se riječ nasumično iz svih kategorija.
- Funkcija vraća par: (rijec, kategorija naziv).

```
def odaberi_rijec(kategorije, kategorija=None):  
    if kategorija and kategorija in kategorije:  
        rijeci = kategorije[kategorija]  
    else:  
        sve_rijeci = sum(kategorije.values(), [])  
        return random.choice(sve_rijeci), "random"  
    return random.choice(kategorije[kategorija]), kategorija
```

Po završetku igre (pobjeda ili poraz), upisuje se rezultat:

- Ime igrača
- Kategorija (mod)
- Broj uzastopnih pogodaka (niz)
- Datum i vrijeme

def zapisi_rezultat(igrac, mod, niz):

```
with open("rezultati.txt", "a", encoding="utf-8") as f:
    datum = datetime.now().strftime("%Y-%m-%d %H:%M")
    f.write(f"{datum} | igrac: {igrac} | mod: {mod} | niz: {niz}\n")
```

Crta se vješalo i lik na osnovu broja preostalih pokušaja:

- Svaki promašaj dodaje novi dio tijela (glava, tijelo, ruke, noge).
- Na početku se crta samo osnovna struktura.

def crtaj_vjesala(self):

```
c = self.canvas
c.delete("all")
konopac_boje = "#333"
figura_boje = "#cc0000"
debljina = 3
c.create_line(50, 180, 150, 180, width=debljina)
c.create_line(100, 180, 100, 20, width=debljina)
c.create_line(100, 20, 150, 20, width=debljina)
c.create_line(150, 20, 150, 40, width=debljina, fill=konopac_boje)
if self.broj_pokusaja <= 5:
    c.create_oval(140, 40, 160, 60, width=debljina, outline=figura_boje)
if self.broj_pokusaja <= 4:
    c.create_line(150, 60, 150, 100, width=debljina, fill=figura_boje)
```

```
if self.broj_pokusaja <= 3:
    c.create_line(150, 70, 130, 90, width=debljina, fill=figura_boje)
if self.broj_pokusaja <= 2:
    c.create_line(150, 70, 170, 90, width=debljina, fill=figura_boje)
if self.broj_pokusaja <= 1:
    c.create_line(150, 100, 130, 130, width=debljina, fill=figura_boje)
if self.broj_pokusaja <= 0:
    c.create_line(150, 100, 170, 130, width=debljina, fill=figura_boje)
```

Uzimamo slovo koje je igrač unio.

Ako je to slovo u riječi: dodaje se u listu pogođenih slova.

Ako nije: broj pokušaja se smanjuje, a slovo se dodaje u pogrešna slova.

```
def provjeri_slovo(self, event):
    unos = self.entry.get().lower().strip()
    self.entry.delete(0, tk.END)
    if not unos or len(unos) != 1 or not unos.isalpha():
        messagebox.showwarning("Greška", "Unesi jedno slovo!")
        return
    if unos in self.slova_pogodjena or unos in self.pogresna_slova:
        return
    if unos in self.trenutna_rijec:
        self.slova_pogodjena.append(unos)
    else:
        self.pogresna_slova.add(unos)
        self.broj_pokusaja -= 1
    if self.broj_pokusaja == 0:
```



```
self.azuriraj_gui()

messagebox.showinfo("Izgubio si", f"Riječ je bila: {self.trenutna_rijec}")

zapisi_rezultat(self.igrac, self.mod, self.niz_pogodaka)

self.niz_pogodaka = 0

self.pitanje_ponovo()

elif all(slovo in self.slova_pogodjena for slovo in self.trenutna_rijec):

    self.niz_pogodaka += 1

    messagebox.showinfo("Bravo!", f"Tačno! Riječ je: {self.trenutna_rijec}")

    self.resetuj_igru()

else:

    self.azuriraj_gui()
```