Izrada aplikacije za vizualno testiranje i unaprijeđenje perceptivnih vještina raspoznavanja boja

Sabol, Damir

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:542362

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-11**

Repository / Repozitorij:

University North Digital Repository

Sveučilište
Sjever

Završni rad br. 830/MM/2023

Izrada aplikacije za vizualno testiranje i unaprijeđenje perceptivnih vještina raspoznavanja boja

Damir Sabol, 0336039761

Varaždin, lipanj 2023. godine

Sveučilište Sjever

Odjel za Multimediju oblikovanje i primjenu

Završni rad br. 830/MM/2023

Izrada aplikacije za vizualno testiranje i unaprijeđenje perceptivnih vještina raspoznavanja boja

Student

Damir Sabol, 0336039761

Mentor

Vladimir Stanisavljević, prof.

Varaždin, lipanj 2023. godine

Predgovor

Tema ovog rada je proces planiranja i implementacije javascript računalne igre "Color Sorting Game" za vježbanje prepoznavanja nijansi boja. Opisan je proces izrade igre, arhitektura, pisanja koda te testiranje i evaluacija rezultata. Igra je inspirirana Farnsworth-Munsell 100 Hue testom. Cilj rada je pružiti pregled tehnologija, istražiti ljudsku vizualnu percepciju boja i testirati osjetljivost na nijanse. Na kraju rada donose se zaključci i mogućnosti daljnjeg razvoja programa.

Sažetak

U ovom završnom radu opisati će se proces izrade računalne igre "Color Sorting Game" za ispitivanje i vježbanje prepoznavanja nijansi boja. Bit će prikazana arhitektura igre, kao i implementacija glavnog koda igre koristeći JavaScript i Phaser.js. Također će se opisati proces testiranja igre i evaluacija dobivenih rezultata. Na kraju rada bit će doneseni zaključci o postignućima i mogućnostima daljnjeg razvoja igre. Prvi dio završnog rada bavit će se pregledom tehnologija koje će biti korištene u izradi igre, kao i ljudskom vizualnom percepcijom boja i testiranjem osjetljivosti na nijanse boja.

Ideja za izradu Color Sorting Game dobivena je istraživanjem Farnsworth-Munsell 100 Hue testa koji se koristi za procjenu vizualnih sposobnosti. Test se sastoji od sortiranja boja prema njihovom spektralnom rasporedu. Inspiriran tom idejom, autor rada je kreirao igru u kojoj korisnici sortiraju boje kako bi razvijali i poboljšavali svoje vizualne sposobnosti. Color Sorting Game omogućuje korisnicima da se zabave dok istovremeno rade na treniranju svojih vizualnih perceptivnih vještina. Kroz kombinaciju igre, izazova i vizualnih elemenata, korisnici mogu testirati i unaprijediti svoju sposobnost razlikovanja boja.

Tijekom izrade igre, posebna pažnja će se posvetiti korisničkom sučelju kako bi se dobila intuitivnost i jednostavnost korištenja. Igra će sadržavati različite razine te će se postepeno povećavati složenost kako bi korisnici mogli kontinuirano napredovati u svojim vještinama prepoznavanja nijansi boja. Također će se istražiti mogućnosti personalizacije igre, kao što su prilagodljivi postavke težine ili dodatni izazovi za napredne korisnike. Kroz izradu ovog završnog rada, cilj je pružiti sveobuhvatan uvid u proces stvaranja računalne igre za testiranje i vježbanje prepoznavanja nijansi boja te istaknuti važnost vizualne percepcije boja.

KLJUČNE RIJEČI:

računalna igra, boje, prepoznavanje nijansi, javascript, phaser.js, php, sql, json, psihologija boja , vizualna percepcija boja

Abstract

This final thesis describes the process of creating the computer game "Color Sorting Game" for testing and practicing color shade recognition. The architecture of the game will be presented, along with the implementation of the main game code using JavaScript and Phaser.js. The process of game testing and the evaluation of the obtained results will also be described. Conclusions about the achievements and possibilities for further game development will be drawn at the end of the thesis. The first part of the thesis will focus on reviewing the technologies that will be used in the game development, as well as human visual color perception and sensitivity testing to color shades.

The idea for creating the Color Sorting Game was inspired by researching the Farnsworth-Munsell 100 Hue test, which is used to assess visual abilities. The test involves sorting colors according to their spectral order. Inspired by this idea, the author of the thesis has created a game in which users sort colors to develop and improve their visual abilities. The Color Sorting Game allows users to have fun while simultaneously training their visual perceptual skills. Through a combination of gameplay, challenges, and visual elements, users can test and enhance their color discrimination abilities.

During the game development, special attention will be given to the user interface to ensure intuitiveness and ease of use. The game will include different levels, gradually increasing in complexity, allowing users to continuously progress in their color shade recognition skills. Additionally, the possibilities for game personalization, such as adjustable difficulty settings or extra challenges for advanced users, will be explored. The goal of this thesis is to provide a comprehensive insight into the process of creating a computer game for testing and practicing color shade recognition while highlighting the importance of visual color perception.

KEYWORDS:

computer game, colors, color shade recognition, JavaScript, Phaser.js, PHP, SQL, JSON, color psychology, visual color perception.

Popis korištenih kratica

HTML	HyperText Markup Language
XML	Extensible Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
PHP	Hypertext Preprocessor
JSON	JavaScript Object Notation
RDBMS	Relational Database Management System
SQL	Structured Query Language
DQL	Data Query Language Commands
DDL	Data Definition Language Commands
DML	Data Manipulation Language Commands
DCL	Data Control Language Commands

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL Odjel za multimediju

STUDIJ preddiplomski stručni studij Multimedija, oblikovanje i primjena

PRISTUPNIK Damir Sabol | JMBAG 0336039761

DATUM 28.06.2023. | KOLEGIJ Programski alati 3

NASLOV RADA Izrada aplikacije za vizualno testiranje i unaprijeđenje perceptivnih vještina raspoznavanja boja

NASLOV RADA NA ENGL. JEZIKU Implementation of an application for visual testing and improvement of color perception skills

MENTOR Vladimir Stanisavljević | ZVANJE viši predavač

ČLANOVI POVJERENSTVA
1. izv.prof.dr.sc Dean Valdec - predsjednik povjerenstva
2. doc.dr. Andrija Bernik - član povjerenstva
3. mr.sc. Vladimir Stanisavljević, v.pred. - mentor
4. pred. Dražen Crčić, dipl.ing. - zamjenski član
5.

Zadatak završnog rada

BROJ 830/MM/2023

OPIS

U procesu učenja ili savladavanja novih vještina današnje generacije spremne su učiti gotovo isključivo uz pomoć računala. Za neke djelatnosti računalna podrška je prirodna i za multimedijalni prikaz sadržaja za učenje i za provjeru znanja, dok je kod nekih djelatnosti moguć samo dio u obliku tekstualnih materijala i verbalnih kvizova. Gemifikacija je proces u kojem se učenje prožima sa zabavnim dijelom igranja kako bi se korisnike još više motiviralo na savladavanje gradiva i vještina. Zahvaljujući dobroj podršci suvremenih računala u prikazu boja i radu sa slikom, primjenom računala i igara moguće je dodatno potpomognuti stjecanje vještina raspoznavanja i rada s bojama.

U radu je potrebno:

* osmisliti aplikaciju u obliku računalne igre za vizualne provjere i učenje perceptivne vještine raspoznavanja boje,
* usporediti metode učenja rada s bojama na računalu i klasičnom radu s fizičkim uzorcima,
* osmisliti i oblikovati web stranicu na kojoj će biti igra i pristup ostvarenim rezultatima,
* razmotriti postupke koji bi omogućili izradu mobilne akcije za rad aplikacije na mobilnim platformama.

Detaljno opisati sve korištene tehnologije i korake u radu potrebne da bi se ostvarilo traženo te detaljno opisati stečena iskustva i postignute rezultate.

ZADATAK URUČEN 06.07.2023. | POTPIS MENTORA Vlado S

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

Sadržaj

1.Uvod....1

2. Obrada zadatka....3

2.1. Korištene tehnologije....3

2.1.1. JAVASCRIPT....3

2.1.2. PHASER JS....5

2.1.3. PHP (Hypertext Preprocessor)....7

2.1.4. JSON (JavaScript Object Notation)....9

2.1.5. SQL (Structured Query Language)....10

2.1.6. MYSQL I PHPMYADMIN....11

2.1.7. CAPACITORJS....13

2.2. Vizualna osjetila čovjeka i prepoznavanje boja....16

2.2.1. Vizualna percepcija boja kod čovjeka....16

2.2.2. Psihologija boja....18

2.2.3. Testiranje osjetljivosti na nijanse boja....20

2.2.4. Ishihara test....21

2.2.5. Farnsworth-Munsell 100 Hue test....22

3. Praktični dio....23

3.1. Proces izrade i implementacije....23

3.1.1. Projekt izrade i arhitektura računalne igre 23
3.1.2. Izrada programskog koda "Color sorting game" računalne igre 25
3.1.3. Opis logike igre "Color Sorting Game" 26
3.2. Datoteke izvornog koda web aplikacije 28
3.2.1. Početni modul za pokretanje igre (index.hml) 28
3.2.2. Implementacija glavne logike igre (game.js) 29
3.2.3. Definiranje razina i njihovih karakteristika (leveli.js) 30
3.2.4. Modul za funkcionalnost glavnog izbornika (menu.js) 31
3.2.5. Modul za prikazivanje elemenata na zaslonu (display.js) 32
3.2.6. Implementacija prikaza uputa za igru (upute.js) 33
3.2.7. Modul za ponovno pokretanje razine (restart.js) 34
3.2.8. Modul za prikazivanje, ažuriranje i spremanje rezultata igrača (scores.js) 35
3.2.9. Prikazivanje uvodne slike i teksta na početku svake razine (splash.js) 36
3.2.10. Definiranje osnovnih postavki phaser.js scena i fizike (config.js) 37
3.2.11. Modul za čitanje rezultata iz baze (readscores.php) 38
3.2.12. Modul za spremanje rezultata u bazu (savescores.php) 39
3.2.13. Modul za ažuriranje podataka u bazi (updatescores.php) 39
4. Analiza rezultata 40
4.1. Metodologija testiranja vizualnih sposobnosti korisnika 40
4.1. Statistika podataka dobivenih u tablici rezultata 41
4.2. Daljnji razvoj i mogućnosti 42
5. Zaključak 43
6. Literatura: 44
7. Popis slika: 45

1.Uvod

Fokus ovog završnog rada je pružiti sveobuhvatan opis koraka u izradi računalne igre. Pokušat ću na principima gemifikacije razviti edukativnu računalnu igru koja će korisnicima pružiti izazovno i zabavno iskustvo za testiranje i poboljšanje njihovih vizualnih sposobnosti prepoznavanja nijansi boja. Kroz detaljan opis arhitekture igre, implementacije koda i evaluacije rezultata, nadam se da će ovaj rad pružiti uvid u važnost vizualne percepcije boja i mogućnosti korištenja računalnih igara u edukativne svrhe.

Ljudska vizualna percepcija boja je fascinantan fenomen koji se temelji na složenom sustavu signala koji putuju od očiju do mozga. Percepcija je pojava koja se odvija u mozgu tj. ovisna je o neurološkom sustavu, pozornosti i psihološkoj osnovi. Znanost orjentirana specificiranju sposobnosti senzora ljudskog vizualnog sustava naziva se psihofizika. Većina vizualnih psihofizikalnih studija usmjerena je prema određivanju "praga" eng. treshold započinjanja reakcije na fizički stimulus (najčešće u razlici intenziteta svijetlosti ili boje). [**1**] Boje su prisutne u svim aspektima našeg svakodnevnog života i ključne su za naše prepoznavanje objekata, orijentaciju u prostoru i komunikaciju. Međutim, sposobnost razlikovanja i kategorizacije boja može značajno varirati među pojedincima. Neki ljudi imaju izuzetno razvijene perceptivne sposobnosti i mogu prepoznati suptilne nijanse boja, dok drugi mogu biti manje osjetljivi na te razlike.

Uzimajući u obzir važnost vizualne percepcije boja i želju za razvojem igre koja bi pomogla korisnicima u poboljšanju svojih vizualnih sposobnosti, odlučio sam implementirati Color Sorting Game. Kroz ovu igru, korisnici će imati priliku testirati svoju sposobnost razlikovanja nijansi boja kroz izazovno i zabavno iskustvo. Arhitektura igre sastoji se od nekoliko ključnih komponenti. Prva komponenta je grafičko sučelje koje korisniku omogućuje interakciju s igrom. Implementirao sam intuitivan dizajn sučelja koji omogućuje jednostavno sortiranje boja. Korisnici će imati pregled nad svim bojama koje se moraju sortirati i bit će im prikazano jednostavno sučelje kako bi lakše prepoznali nijanse i vršili sortiranje.

Računalne igre imaju veliku popularnost i koriste se ne samo za zabavu, već i u različite druge svrhe poput obrazovanja, simulacija, treninga i razvoja kognitivnih vještina. Kako bi obrazovna igra bila zabavna koristi se gemifikacijski pristup. Osnovni elementi kako bi

obrazovna aplikacija bila zabavna, prema knjizi prof.Bernika "Gemifikacija visokoškolskog obrazovanja" su: bodovi, top-lista postignuća, bedževi, razine, vizualni prikaz napretka u sustavu, pokazatelj uspješnosti korisnika, motivirajuća priča, jasni ciljevi, povratna informacija, sustav nagrađivanja, avatar, usputni izazovi i dinamični razvoj korisničkog profila. [**2**]

Druga ključna komponenta je algoritam koji generira niz boja koje korisnik treba sortirati. Ovaj algoritam uzima json inicijalne podatke o rasporedu boja i zatim svaki put slučajno stvarad drugačiji raspored. Kroz svaku razinu igre biti će sve više boja i suptilnijih nijansi, potičući korisnike da unaprijede svoje sposobnosti tijekom igranja. Treća komponenta je glavni kod igre koji je implementiran koristeći JavaScript i Phaser.js. Phaser.js je popularan i moćan framework za razvoj HTML5 igara koji pruža razne funkcionalnosti i alate za stvaranje igara visoke kvalitete. Uz pomoć Phaser.js-a mogao sam učinkovito implementirati logiku igre, upravljanje korisničkim interakcijama i vizualne efekte.

Važan aspekt izrade igre bilo je testiranje. Proveo sam testiranje na nekoliko igrača kako bih evaluirao učinkovitost igre u poboljšanju prepoznavanja nijansi boja. U budućnosti planiram proširiti broj sudionika i testirati ih kako bi se procijenila njihova osjetljivost na nijanse boja prije i poslije igranja igre. Hipoteza je da bi rezultati testiranja pokazali statistički značajno poboljšanje u sposobnosti razlikovanja boja kod sudionika nakon igranja. Uvjeren sam da bi igra mogla biti učinkovit alat za trening i vježbanje prepoznavanja nijansi boja na zabavan način. Daljnji razvoj igre uključivao bi dodavanje novih razina s još izazovnijim zadacima, uvođenje više načina igre i dodatnih funkcionalnosti, kao i poboljšanje vizualnog aspekta igre kako bi se korisnicima pružilo još privlačnije iskustvo. Također planiram proširiti dostupnost igre na različite platforme i uključiti mogućnost natjecanja između igrača putem interneta.

Izrada vlastite računalne igre može biti izazovan, ali također i kreativan proces. U ovom završnom radu fokusirat ću se na izradu računalne igre koja osim zabavne namjene ima i svrhu ispitivanja i vježbanja prepoznavanja nijansi boja. Cilj ove igre je pružiti korisniku mogućnost da vježba svoje vizualne sposobnosti prepoznavanja nijansi boja. Kroz različite razine igre, korisnik će morati identificirati i upariti boje prema njihovim nijansama. Na taj način kroz igru, korisnik će razvijati svoju sposobnost prepoznavanja boja i time poboljšavati svoju vizualnu osjetljivost.

2. Obrada zadatka

U prvom dijelu završnog rada opisati ću tehnologije korištene prilikom izrade aplikacije , a nokon toga ću detaljno opisati cijeli postupak planiranja i izrade aplikacije. Tehnološki okvir i alati igraju ključnu ulogu u uspješnom razvoju igre, a njihov odabir temelji se na funkcionalnostima i zahtjevima projekta. Tema drugog dijela je kako sam koristio ove tehnologije i alate tijekom razvojnog procesa kako bih postigao ciljeve i funkcionalnost aplikacije "Color Sorting Game".

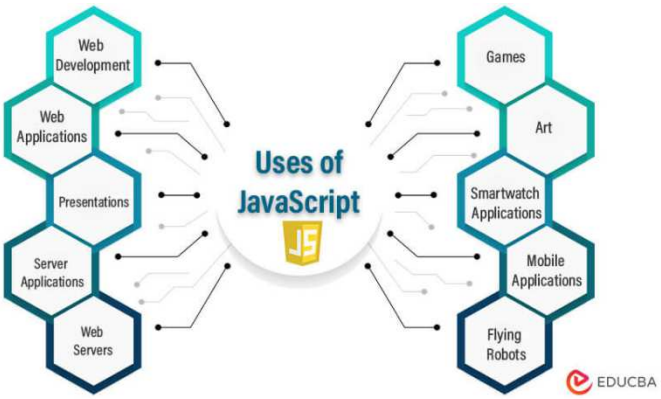
2.1. Korištene tehnologije

2.1.1. JAVASCRIPT

JavaScript je najčešće korišteni programski jezik za razvoj web aplikacija. Nastao je 1995. godine kao jednostavan jezik za manipulaciju HTML dokumentima, ali je s vremenom postao sveobuhvatan jezik koji se može koristiti i za izradu složenih web aplikacija, igara i samostalnih windows, linux ili android aplikacija. Jedna od glavnih karakteristika JavaScripta je da je interpretirani jezik, što znači da se izvorni kod izvršava izravno u pregledniku ili interpreteru, bez potrebe za prethodnom kompilacijom. [**3**] Ovo omogućuje brz razvoj i prototipiranje aplikacija, jer se promjene mogu vidjeti odmah bez potrebe za kompajliranjem. Još jedna važna karakteristika JavaScripta je da je objektno orijentiran jezik. To znači da programeri mogu definirati i koristiti objekte koji imaju svoja svojstva i metode. JavaScript također podržava funkcionalno programiranje. Funkcije se mogu dodijeliti varijablama, proslijediti kao argumenti drugim funkcijama i vratiti kao rezultat iz drugih funkcija. Ova mogućnost omogućava programerima pisanje fleksibilnog i modularnog koda. Još jedna važna značajka JavaScripta je podrška za asinkrono programiranje. To je posebno korisno za izvršavanje operacija koje zahtijevaju vremenski zahtjevne zadatke, poput dohvaćanja podataka s udaljenih servera. Asinkrono programiranje omogućuje izvršavanje drugih operacija dok se čeka na završetak asinkronog zadatka, čime se poboljšava ukupna performansa aplikacije. JavaScript ima veliki broj biblioteka koda(frameworks) koji olakšavaju razvoj web aplikacija. Jedan od najpopularnijih framework-a je React, koji se koristi za izgradnju korisničkih sučelja. Često se koriste i Angular, popularan okvir koji pruža potpunu infrastrukturu za izgradnju web aplikacija, Vue.js, koji kombinira jednostavnost i

performanse, te jQuery, koji olakšava manipulaciju HTML elementima i obradu događaja. JavaScript se također koristi izvan web-a. Node.js je okvir koji omogućuje izvršavanje JavaScripta na serveru izvan preglednika. JavaScript se koristi za razvoj mobilnih aplikacija pomoću frameworka poput Capacitor JS, Cordova, Phone Gap , React Native i Ionic. Kao svaki jezik, JavaScript ima svoje snage i slabosti. Jedna od prednosti je njegova široka dostupnost i podrška, jer je ugrađen u svaki moderni preglednik. Također je vrlo fleksibilan i može se koristiti za različite vrste aplikacija. Međutim, JavaScript također ima neke nedostatke, poput manje čitljivog koda zbog nedostatka strogog definiranja tipova podataka. i mogućnosti grešaka u razvoju aplikacija.

Uz rast web tehnologija, JavaScript postaje sve važniji jezik za svakog programera koji želi razvijati moderne web aplikacije. Njegova prilagodljivost i raznovrsnost čine ga snažnim alatom u razvoju programa. U mom završnom projektu, koji je bio igra sortiranja boja, koristio sam JavaScript kao glavni programski jezik za razvoj i implementaciju igre. JavaScript je bio ključan za stvaranje interaktivnosti, upravljanje korisničkim unosom i manipulaciju elementima na ekranu. Jedan od glavnih elemenata igre je mreža kvadrata različitih boja. Koristio sam JavaScript kako bih generirao tu mrežu dinamički, na temelju postavki igre. Pomoću JavaScripta sam definirao logiku koja generira nasumične boje i raspoređuje kvadrate u dvodimenzionalnoj matrici. Također sam koristio JavaScript za upravljanje korisničkim unosom. Kada korisnik klikne na kvadrat, JavaScript detektira taj događaj i izvršava odgovarajuću logiku. Na primjer, kada korisnik klikne na dva kvadrata iste boje, JavaScript provjerava je li sortiranje ispravno i ako je, dodaje bodove korisniku i onemogućuje da se sortirani kvadrati više mijenjaju. JavaScript je omogućio i praćenje bodova i razina u igri. Kada je korisnik uspješno sortirao kvadrate, JavaScript je povećavao bodove korisnika i prelazio na sljedeću razinu. Također sam koristio JavaScript za pohranjivanje i prikazivanje najboljeg rezultata korisnika. Uz osnovni jezik koristio sam i Phaser JS biblioteku koda kako bih olakšao razvoj. Kroz upotrebu JavaScripta u mom završnom projektu, stekao sam dublje razumijevanje jezika i njegovih mogućnosti u razvoju igara. JavaScript mi je omogućio stvaranje interaktivne i zabavne igre sortiranja boja koja je temeljni dio mog završnog rada. Uz kreativno korištenje JavaScripta, uspio sam ostvariti svoju viziju igre i stvoriti izazovno korisničko iskustvo za igrače.



Slika 1 - Upotreba javascripta (poglavlje 2.1.1.)

2.1.2. PHASER JS

Phaser.js je open-source biblioteka za razvoj igara u JavaScriptu. Phaser.js omogućuje programerima da stvaraju 2D igre koje se mogu pokretati u web preglednicima i na mobilnim uređajima. Ovaj framework pruža sveobuhvatni set alata za razvoj igara, uključujući upravljanje scenama, animacije, fiziku, zvuk i još mnogo toga. Jedna od glavnih prednosti Phaser.js-a je jednostavnost upotrebe. Biblioteka pruža intuitivno sučelje i dobro dokumentiranu API dokumentaciju koja olakšava programerima da započnu s razvojem igara. Sintaksa jezika JavaScript koja se koristi u Phaser.js-u također je čista i čitljiva, što olakšava održavanje i dijeljenje koda među razvojnim timovima. [**4**]

Phaser.js podržava razne vrste igara, poput platformskih igara, arkadnih igara, pucanja, puzzle igara i mnoge druge. Sa svojom podrškom za animaciju, programeri mogu jednostavno stvarati glatke i privlačne vizualne efekte za svoje igre. Također, Phaser.js omogućuje jednostavno upravljanje korisničkim unosom i događajima, što je ključno za interakciju s igrom.

Još jedna važna značajka Phaser.js-a je podrška za fiziku. Framework ima ugrađenu podršku za simulaciju fizikalnih objekara u arkadnim igrama, koja omogućuje simulaciju realističnog ponašanja objekata u igri. Ova značajka olakšava programerima da dodaju gravitaciju, kolizije, detekciju sudara i druge fizikalne efekte u svoje igre. Phaser.js također podržava integraciju s drugim bibliotekama, što omogućuje programerima da koriste postojeće resurse i alate u svojim igrama. Na primjer, okvir ima ugrađenu podršku za integiranje s Pixi.js, popularnom bibliotekom za rad s grafikom. Također se može koristiti s drugim alatima poput Tiled Editora za izradu mapa i Spinea za animaciju likova. Postoje mnogi resursi dostupni programerima, kao što su primjeri koda, tutorijali, forumi i

blogovi, koji olakšavaju učenje i rješavanje problema. Phaser.js je u skladu s najnovijim standardima web tehnologija i podržava moderne značajke poput WebGL-a za hardversko ubrzavanje grafike i Web Audio API za naprednu manipulaciju zvukom. Također podržava mobilne uređaje putem Cordove, što omogućuje programerima da izgrade igre koje se mogu reproducirati na različitim platformama. U mom završnom projektu, igri sortiranja boja, koristio sam Phaser.js kao glavni framework za razvoj igre. Phaser.js je open-source JavaScript framework za razvoj 2D igara koji pruža moćne alate i funkcionalnosti koje su mi bile potrebne za implementaciju igre.

Koristio sam Phaser.js za upravljanje korisničkim unosom, za praćenje događaja poput klika na kvadrat, povlačenja i otpuštanja kvadrata te interakcije s ostalim elementima igre. To mi je omogućilo da implementiram logiku sortiranja kvadrata. Phaser.js također pruža podršku za fiziku, što bi bilo korisno u implementaciji kolizija između kvadrata. Mogao bih definirati fizikalna svojstva kvadrata poput mase, brzine i gravitacije te pratiti kolizije između njih. Na taj način bih mogao dodati elemente realizma igri. Jedna od najvažnijih funkcionalnosti Phaser.js-a koju sam iskoristio bila je mogućnost upravljanja scenama. Scena u Phaser.js predstavlja zaseban dio igre, poput početnog zaslona, razine ili zaslona za prikaz rezultata. Mogao sam lako prelaziti između različitih scena i upravljati logikom igre u svakoj od njih. Korištenje Phaser.js-a u mom završnom projektu donijelo je mnoge prednosti. Osim što je sadrži moćne alate za grafiku, upravljanje korisničkim unosom i fizikom, framework također ima opsežnu i lako dostupnu dokumentaciju. To mi je omogućilo brži razvoj igre i lakše rješavanje problema.

Slika 2 - PhaserJS osnovni podaci (poglavlje 2.1.2.)

2.1.3. PHP (Hypertext Preprocessor)

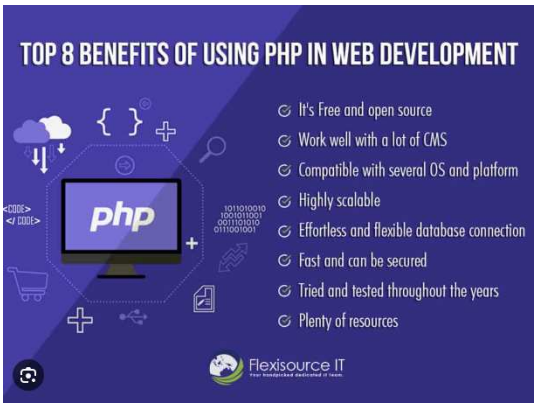
PHP (Hypertext Preprocessor) je popularan skriptni jezik koji se najčešće koristi za razvoj web aplikacija i dinamičkih web stranica. Njegova fleksibilnost, jednostavnost i široka podrška čine ga jednim od najčešće korištenih programskih jezika. U mom završnom radu, PHP je bio ključni alat koji sam koristio za implementaciju backend funkcionalnosti moje web aplikacije. Jedna od ključnih prednosti PHP-a je jednostavnost sintakse i brza izvedba. PHP je interpretirani jezik, što znači da se kôd izvršava na serveru i rezultati se šalju klijentskom pregledniku. To omogućava brzu i dinamičku generaciju sadržaja na temelju korisničkih zahtjeva. PHP također ima bogatu biblioteku funkcija i alata koji olakšavaju rad. [**5**] Ove funkcije uključuju rad s bazama podataka, manipulaciju datotekama, upravljanje sesijama i mnoge druge. Uz to, PHP podržava različite baze podataka, kao što su MySQL, PostgreSQL i SQLite, omogućavajući mi da lako povežem svoju web aplikaciju s backend bazom podataka. Još jedna prednost PHP-a je podrška za objektno orijentirano programiranje (OOP). OOP pruža strukturiraniji i modularniji pristup razvoju aplikacija. Korištenje klasa, objekata, nasljeđivanja i drugih OOP koncepta omogućava da se programski kod organizira na efikasan način. Sigurnost je također važan aspekt PHP-a. PHP pruža razne sigurnosne mehanizme i funkcije zaštite od napada kao što su XSS (Cross-Site Scripting) i SQL injection. Upotreba ovih mehanizama pomaže u zaštiti web aplikacije od potencijalnih sigurnosnih propusta.
Jedna od najbitnijih značajki PHP-a je integracija s HTML-om i JavaScriptom. PHP omogućava izmjenu PHP koda s HTML-om, što olakšava generiranje dinamičkih web stranica. Također se može koristiti PHP za generiranje JavaScript koda i interakciju s korisničkim sučeljem. Razvojna zajednica PHP-a je vrlo aktivna i na webu je ogroman broj resursa, kao što su forumi, blogovi, tutorijali i dokumentacija.

U mom završnom radu, PHP mi je omogućio implementaciju backend logike i povezivanje s bazom podataka, što je ključno za uspješno funkcioniranje moje web aplikacije. U mom završnom projektu "Color Sorting Game" PHP je odigrao ključnu ulogu u razvoju backend funkcionalnosti. Korištenje PHP-a omogućilo mi je obradu podataka, povezivanje s bazom podataka i generiranje dinamičkog sadržaja za korisnike. Jedan od načina na koji sam koristio PHP bio je za spremanje rezultata igrača u bazu podataka. Implementirao sam PHP skriptu "savescores.php" koja je bila odgovorna za primanje podataka kao što su ID igre, ID cookiesa, bodovi, razina, datum, IP adresa i ime igrača putem POST metode. Nakon toga,

koristio sam te podatke za izvršavanje SQL upita kako bih ih spremao u bazu podataka. Korištenje PHP-a za ove zadatke omogućilo mi je sigurno spremanje podataka igrača i omogućilo preglednost i organizaciju podataka u bazi. Također sam koristio PHP za dohvat najboljih rezultata iz baze podataka. Korištenje PHP skripte "readscores.php" omogućilo mi je povezivanje s bazom podataka, izvršavanje SQL upita za dohvat podataka o igračima i pripremu tih podataka za prikaz na korisničkom sučelju. Nakon što sam dohvatio podatke, koristio sam PHP funkcije za obradu i oblikovanje podataka te sam na kraju koristio funkciju json_encode kako bih pretvorio podatke u JSON format koji se zatim šalje klijentskom sučelju.

Kako bih omogućio igračima ažuriranje njihovih imena, koristio sam PHP skriptu "updatescores.php". Ova skripta je prohvatila podatke poput ID-a igre i novog imena igrača koje je korisnik unio putem POST metode. Zatim sam koristio PHP za izvršavanje SQL upita kako bih ažurirao ime igrača u bazi podataka. Korištenje PHP-a za ove zadatke omogućilo mi je fleksibilnost i kontrolu nad ažuriranjem podataka igrača.

Ukratko, PHP je bio ključni alat u razvoju mog završnog projekta "Color Sorting Game". Omogućio mi je sigurno spremanje i dohvaćanje podataka igrača iz baze podataka, kao i generiranje dinamičkog sadržaja za korisnike. PHP je pokazao svoju snagu u obradi podataka, povezivanju s bazom podataka i omogućio mi je da pružim interaktivno iskustvo igračima na mojem projektu.



Slika 3 - Prednosti korištenja PHP-a (poglavlje 2.1.3.)

2.1.4. JSON (JavaScript Object Notation)

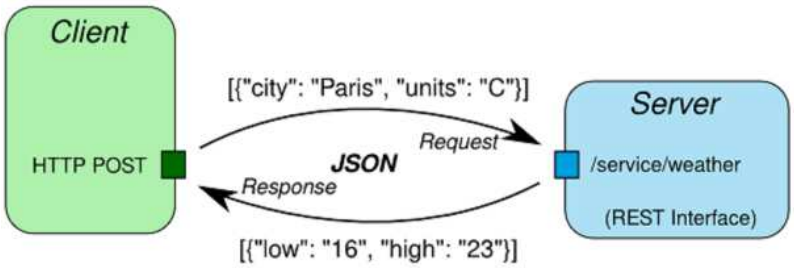
JSON (JavaScript Object Notation) je popularan format za razmjenu podataka između različitih aplikacija. On pruža jednostavnu i laganu sintaksu koja je lako čitljiva i razumljiva ljudima i strojevima. U mom završnom radu "Color Sorting Game" koristio sam JSON format za komunikaciju između frontend i backend dijela aplikacije, kao i za pohranu podataka o rezultatima igrača. JSON je baziran na konceptu objekata koji se sastoje od parova ključ-vrijednost, gdje ključevi moraju biti jedinstveni. Vrijednosti mogu biti različitih tipova, uključujući brojeve, nizove, objekte, boolean vrijednosti i null. Ova fleksibilnost JSON formata omogućava nam da organiziramo podatke na strukturiran način i olakšava manipulaciju s njima. [**6**]

U mom projektu, JSON se koristio za prijenos podataka između backend PHP skripti i frontend JavaScript koda. Kada je igrač završio igru i postigao rezultat, javascript kod je prikupio sve potrebne informacije kao što su ime igrača, bodovi, razina i datum, te je te podatke formatirao u JSON format. Zatim je taj JSON objekt poslan prema backendu kako bi se rezulatati spremili u bazi podataka. Kasnije, kada je potrebno dohvatiti rezultate igrača, PHP skripta je dohvatila podatke iz baze i pretvorila ih u JSON format kako bi se lako mogli prikazati na korisničkom sučelju. Korištenje JSON formata je donijelo nekoliko prednosti. Prvo, JSON je lako čitljiv i razumljiv za čitanje ljudima, što olakšava preglednost podataka. Također, JSON je kompatibilan s većinom programskih jezika i tehnologija, što ga čini vrlo fleksibilnim za integraciju s različitim sustavima. Također, JSON je lagan i nezahtjevan format, što ga čini idealnim za prijenos podataka preko mreže.

U mom završnom projektu "Color Sorting Game" također sam koristio JSON za organizaciju podataka o različitim razinama u igri. U datoteci "leveli.js" kreirao sam JSON objekt koji sadrži informacije o svakoj razini. Svaka razina ima svoj identifikator, broj redova i stupaca, matricu boja te vremensko ograničenje za završetak razine. Koristio sam JSON format kako bih strukturirano pohranio sve potrebne podatke. Svaka razina ima svoj objekt u JSON-u s ključevima poput "level", "rows", "columns", "colors" i "timeLimit". Podaci su organizirani na način da je svaka razina zaseban element u polju.

JSON objekt sam koristio i za prikazivanje uvodnog teksta za svaku razinu. Kada igrač započne određenu razinu, frontend aplikacija dohvaća odgovarajući JSON objekt za tu razinu i prikazuje uvodni tekst igraču. Tekst se prikazuje na temelju identifikatora razine koji je

spremljen u JSON-u. Korištenje JSON-a omogućilo je organizaciju i strukturiranje podataka o razinama na jednostavan način. JSON je lako čitljiv i razumljiv format, što je olakšalo rad s podacima u razvoju igre. Također, JSON format omogućuje jednostavno proširivanje i dodavanje novih razina bez potrebe za izmjenom izvornog koda aplikacije. Ukratko, korištenje JSON-a u mom završnom projektu omogućilo je organizaciju podataka o razinama igre na strukturiran način. JSON format je pružio jednostavnost i fleksibilnost u radu s podacima te omogućio dinamičko prikazivanje uvodnog teksta za svaku razinu u igri.



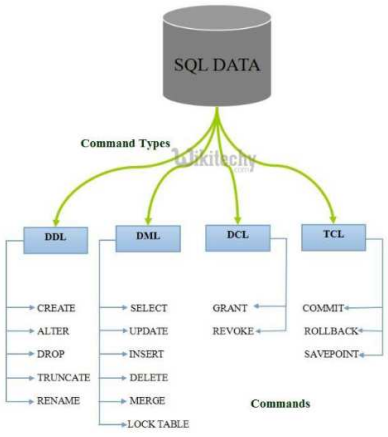
Slika 4 - Primjer upotreba JSON-a (poglavlje 2.1.4.)

2.1.5. SQL (Structured Query Language)

SQL (Structured Query Language) je jezik za upravljanje relacijskim bazama podataka. Koristi se za definiranje, manipuliranje i upravljanje podacima unutar baze podataka. SQL je standardni jezik koji se koristi u većini relacijskih baza podataka i pruža efikasne i fleksibilne mogućnosti za rad s podacima. SQL se sastoji od različitih vrsta naredbi koje se koriste za izvršavanje operacija nad bazom podataka. Naredbe se dijele na četiri osnovne kategorije: DDL (Data Definition Language), DML (Data Manipulation Language), DQL (Data Query Language) i DCL (Data Control Language). DDL naredbe se koriste za definiranje strukture baze podataka. To uključuje stvaranje tablica, definiranje veza između tablica, definiranje indeksa i ograničenja integriteta podataka. Primjeri DDL naredbi su CREATE TABLE, ALTER TABLE i DROP TABLE. DML naredbe se koriste za manipulaciju podacima unutar tablica. To uključuje dodavanje novih zapisa, ažuriranje postojećih zapisa i brisanje zapisa iz tablica. Primjeri DML naredbi su INSERT, UPDATE i DELETE. DQL naredbe se koriste za dohvaćanje podataka iz baze podataka. To uključuje izvršavanje upita nad tablicama kako bi se dobio željeni rezultat. Primjeri DQL naredbi su SELECT, FROM i WHERE. DCL naredbe se koriste za kontrolu pristupa podacima. To uključuje dodjelu i povlačenje

privilegija korisnicima baze podataka te upravljanje transakcijama. Primjeri DCL naredbi su GRANT, REVOKE i COMMIT. SQL nudi napredne mogućnosti za upravljanje podacima kao što su grupiranje, sortiranje, filtriranje, spajanje tablica i izvođenje složenih izračuna. Također podržava indekse koji poboljšavaju performanse upita, transakcije koje osiguravaju dosljednost podataka te pohranjene procedure i funkcije koje olakšavaju ponovno korištenje logike kodiranja. U mom završnom projektu "Color Sorting Game" koristio sam SQL za stvaranje i upravljanje bazom podataka u kojoj sam pohranjivao rezultate igrača. Koristio sam SQL naredbe za stvaranje tablice "rezultati" s odgovarajućim stupcima za pohranu podataka o rezultatima igre. Također sam koristio SQL naredbe za dodavanje, ažuriranje i dohvaćanje podataka iz tablice. [7]

SQL je snažan jezik koji omogućuje efikasno i pouzdano upravljanje podacima. Njegova struktura i sintaksa su intuitivni, što olakšava rad s bazama podataka. SQL je standardiziran jezik, što znači da se može koristiti s različitim bazama podataka bez potrebe za izmjenom koda. To ga čini vrlo fleksibilnim i prilagodljivim za razne projekte. U mom projektu, SQL je korišten za stvaranje i upravljanje bazom podataka u kojoj su pohranjeni rezultati igrača.



Slika 5 - Vrste SQL instrukcija (poglavlje 2.1.5.)

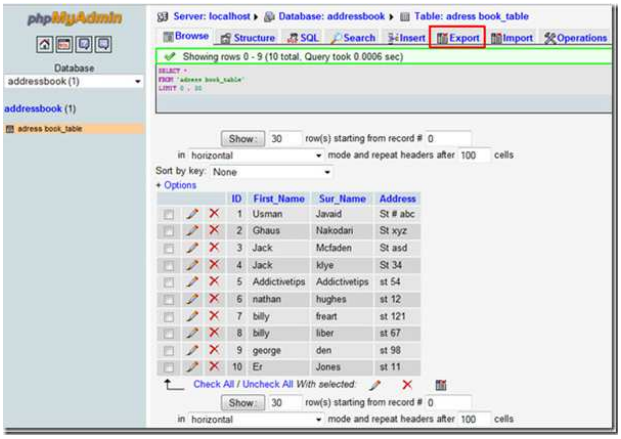
2.1.6. MYSQL I PHPMYADMIN

MySQL i phpMyAdmin su dva ključna alata koji se koriste u svijetu razvoja web aplikacija i upravljanja bazama podataka. MySQL je popularni sustav za upravljanje bazama podataka (RDBMS - Relational Database Management System), dok je phpMyAdmin web aplikacija koja pruža grafičko sučelje za administraciju MySQL baza podataka. MySQL je

jedan od najpoznatijih RDBMS sustava na svijetu. On je izuzetno pouzdan, skalabilan i pruža visoke performanse. MySQL je baziran na jeziku SQL (Structured Query Language) i omogućuje učinkovito organiziranje, pohranjivanje i upravljanje velikim količinama strukturiranih podataka. MySQL se koristi u širokom rasponu aplikacija, uključujući web stranice, e-trgovinu, CRM sustave, CMS sustave, aplikacije za analitiku podataka i još mnogo toga. Jedna od ključnih prednosti MySQL-a je njegova otvorenost i velika zajednica korisnika koja pruža podršku i razvoj dodatnih funkcionalnosti.

phpMyAdmin je popularna web aplikacija koja omogućuje upravljanje MySQL bazama podataka putem preglednika. Ona pruža jednostavno i intuitivno sučelje za administraciju i manipulaciju bazom podataka. Korisnici mogu izvršavati razne zadatke kao što su stvaranje, brisanje i uređivanje baza podataka, tablica, indeksa i korisnika. [**8**] Također podržava izvršavanje SQL upita, uvoz i izvoz podataka, generiranje izvještaja i postavljanje sigurnosnih postavki. phpMyAdmin je posebno koristan za korisnike koji nemaju detaljno znanje SQL-a, jer pruža grafičko sučelje za lakše rukovanje bazom podataka. Korištenje MySQL-a i phpMyAdmin-a donosi brojne prednosti u razvoju web aplikacija. Prvo, MySQL je pouzdana i stabilna baza podataka, podržava transakcije, osigurava dosljednost podataka i ima mehanizme oporavka od kvarova. Također podržava napredne funkcionalnosti kao što su indeksi, ograničenja integriteta podataka, pohrana procedura i pogleda. Drugo, MySQL je visoko skalabilan i može se nositi s velikim opterećenjem i količinom podataka. PhpMyAdmin olakšava upravljanje bazom podataka putem grafičkog sučelja, čime se smanjuje potreba za ručnim pisanjem SQL upita i olakšava administracija. U mom završnom projektu "Color Sorting Game", MySQL i phpMyAdmin su bili ključni za upravljanje podacima vezanim za igru. Koristio sam MySQL za stvaranje i upravljanje bazom podataka koja je sadržavala informacije o korisnicima, njihovim rezultatima igre, postavkama i drugim relevantnim podacima. Pomoću phpMyAdmin-a sam mogao jednostavno upravljati bazom podataka kroz intuitivno sučelje. Kroz phpMyAdmin, mogao sam stvarati, uređivati i brisati tablice, provoditi upite i analizirati podatke. To je znatno olakšalo razvoj i održavanje baze podataka za projekt.

Slika 6 - Logo MySQL baze podataka (poglavlje 2.1.6.)



Slika 7 - phpMyAdmin web sučelje (poglavlje 2.1.6.)

2.1.7. CAPACITORJS

CapacitorJS je open-source framework za izgradnju naprednih web aplikacija i mobilnih aplikacija temeljenih na web tehnologijama kao što su HTML, CSS i JavaScript. On omogućuje razvoj i izgradnju aplikacija koje se mogu izvršavati na različitim platformama, uključujući web preglednike, iOS i Android uređaje. CapacitorJS pruža različite značajke i alate koji olakšavaju razvoj mobilnih aplikacija. Nudi jednostavan API za pristup funkcionalnostima uređaja poput kamere, GPS-a, akcelerometra i drugih senzora. Također podržava integraciju s nativnim SDK-ovima i pluginima kako bi se omogućila dodatna funkcionalnost. CapacitorJS koristi web tehnologije za izgradnju korisničkog sučelja, što olakšava ponovno korištenje postojećeg koda i pruža fleksibilnost pri dizajniranju korisničkog iskustva.

U mom završnom projektu "Color Sorting Game", planiram proširiti projekt kroz korištenje CapacitorJS-a. Trenutno je aplikacija razvijena kao web aplikacija koja se izvršava u web pregledniku. Međutim, želim proširiti funkcionalnost aplikacije i pretvoriti je u Android aplikaciju kako bih omogućio korisnicima da je koriste na svojim mobilnim

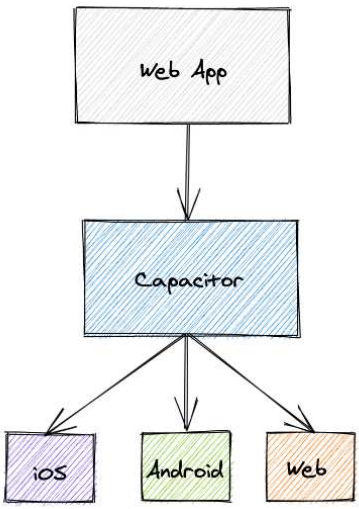
uređajima bez potrebe za pristupom web pregledniku. Korištenje CapacitorJS-a za pretvaranje web aplikacije u Android aplikaciju ima brojne prednosti. Prvo, omogućuje bolje integriranje s mobilnim platformama i pristup funkcionalnostima uređaja. Mogu koristiti funkcije kao što su kamera i senzori uređaja kako bih poboljšao iskustvo korisnika i omogućio napredne značajke. Drugo, aplikacija će biti dostupna korisnicima putem Google Play Storea, što će omogućiti veću dostupnost. Treće, CapacitorJS pruža fleksibilnost u razvoju i održavanju koda, jer koristim već poznate web tehnologije.

Planiram proširiti funkcionalnost aplikacije dodavanjem značajki kao što su mogućnost prijave i registracije korisnika, praćenje rezultata igrača, dijeljenje rezultata putem društvenih mreža i integracija s analitičkim alatima za praćenje performansi aplikacije. Također želim poboljšati korisničko sučelje i iskustvo kroz prilagodbu dizajna, dodavanje animacija i poboljšanje interakcije korisnika. Pretvaranje web aplikacije u Android aplikaciju pomoću CapacitorJS-a je jednostavan proces. CapacitorJS pruža alate za izgradnju, testiranje i pakiranje aplikacije za Android i IOS platforme. Koristit ću ove alate za generiranje APK datoteke koja se može instalirati na Android uređaje. Kroz ovu nadogradnju, ciljam na širu publiku i bolje korisničko iskustvo. Mobilna aplikacija pruža korisnicima jednostavan pristup i mogućnost igranja igre na svojim mobilnim uređajima bilo kada i bilo gdje. Također će omogućiti bolje praćenje i analizu podataka o korisnicima i njihovim rezultatima igre.

Pretvaranje "Color Sorting Game" u mobilnu aplikaciju za Android ili iOS zahtijevalo bi prilagodbu grafičkih i sučelja igre za mobilne uređaje, optimizaciju performansi i prilagođavanje kontrola igre na dodirne zaslone. Također bi bilo važno provesti testiranje na različitim mobilnim uređajima kako bi se osigurala kompatibilnost i stabilnost aplikacije. Pretvaranje "Color Sorting Game" u mobilnu aplikaciju otvara mogućnosti za širenje korisničke baze, veću dostupnost igre i potencijalno komercijaliziranje putem prodaje aplikacije ili uključivanjem reklama ili mikrotransakcija unutar igre.

Tabela 1 - Usporedba korištenih tehnologija

Tehnologija	Prednosti	Nedostaci	Primjena
Javascript	Univerzalnost i široka podrška	Izvršava se na klijentskoj strani što može izložiti sigurnosne propuste	Prog.jezik za razvoj web aplikacija, interaktivni elementi na web stranicama, igre
Phaser JS	Jednostavna upotreba i brza izrada igara	Specifično orijentiran na razvoj igara	Javascript framework za 2D igare, web igre, mobilne igre
PHP (Hypertext Preprocessor)	Jednostavnost i široka podrška za web razvoj	Performanse mogu biti slabije u odnosu na neke druge jezike	Programski jezik za backend web razvoj, dinamičko generiranje web stranica
JSON (JavaScript Object Notation)	Lagan i čitljiv format za razmjenu podataka	Ne podržava složene tipove podataka, poput funkcija	Format za razmjenu podataka između klijentskih i poslužiteljskih aplikacija, konfiguracijske datoteke
SQL (Structured Query Language)	Moćan jezik za upravljanje bazama podataka	Specifična upotreba za rad s relacijskim bazama podataka	Programski jezik za rad sa bazama podataka, dohvaćanje, unos i ažuriranje podataka
MySQL i phpMyAdmin	Jednostavna instalacija i upravljanje bazama podataka	Performanse mogu biti ograničene u određenim uvjetima	Baza i sučelje za pohranu i upravljanje podacima, razvoj web aplikacija koje koriste MySQL
CapacitorJS	Omogućava razvoj mobilnih aplikacija s web tehnologijama	Ne podržava sve platforme	Javascript framework za razvoj mobilnih aplikacija koristeći web tehnologije



Slika 8 - CapacitorJS prikaz (poglavlje 2.1.7.)

Slika 9 - CapacitorJS logo (poglavlje 2.1.7.)

2.2. Vizualna osjetila čovjeka i prepoznavanje boja

2.2.1. Vizualna percepcija boja kod čovjeka

U hrvatskom jeziku pojam "boja" može se interpretirati na tri osnovna značenja. Boja može biti materijalna tvar kao nosilac obojenja (obično naziv pigmenta cink bijela, kobalt plava), fizikalno mjerljiv stimulus (određena valna dužna vidljivog dijela spektra) i osjet u čovjeku izazvan percepcijom svijetlosti emitirane od nekog izvora ili reflektirane od površine nekog tijela. [**1**]

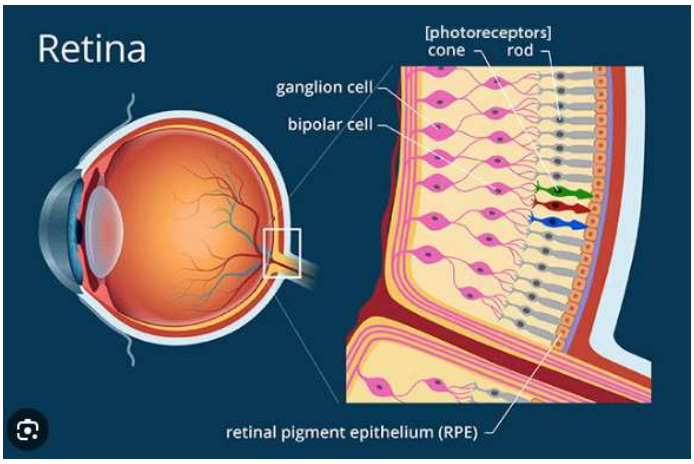
Ljudska vizualna percepcija boja je složen fenomen koji se odnosi na način na koji ljudi opažaju i interpretiraju boje oko sebe. Boje su prisutne u našem svakodnevnom životu i imaju značajan utjecaj na naše emocionalno stanje, percepciju svijeta i interakciju s okolinom.

Naša sposobnost razlikovanja boja temelji se na radu našeg vizualnog sustava koji uključuje oči, optičke živce i mozak. Kada svjetlost prolazi kroz oko, dolazi do aktivacije specijaliziranih stanica nazvanih štapići i čunjići koji se nalaze na mrežnici. Štapići su

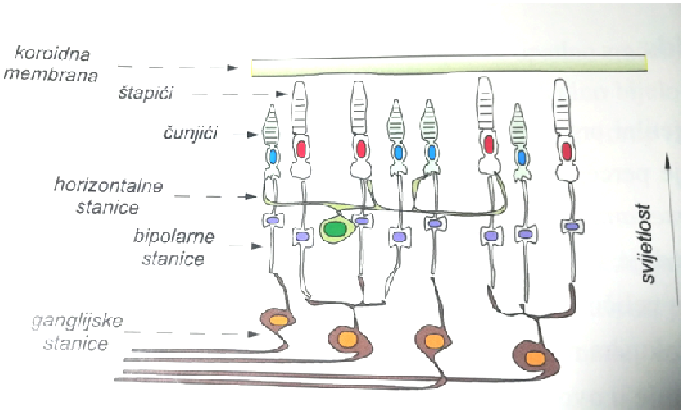
odgovorni za percepciju svjetlosti i tamnih tonova, dok čunjići igraju ključnu ulogu u percepciji boje. Postoje tri vrste čunjića koji su odgovorni za percepciju triju osnovnih boja: crvene, zelene i plave. Kombinacija aktivnosti ovih čunjića omogućuje nam razlikovanje širokog spektra boja. Na primjer, kada su aktivirani crveni i zeleni čunjići, vidimo žutu boju, dok aktivnost svih triju čunjića rezultira percepcijom bijele boje.

Percepcija boje nije apsolutna i može varirati između pojedinaca. Ova varijabilnost može biti posljedica genetskih čimbenika, različitih osjetljivosti čunjića ili individualnih razlika u percepcijskim procesima. Pored fizioloških čimbenika, percepcija boje također je podložna utjecaju konteksta, kulture i iskustva. Na primjer, određene boje mogu imati simboličko značenje u određenim kulturama ili kontekstima. Crvena boja može se povezivati s ljubavlju ili opasnošću, dok zelena boja može se povezivati s prirodom ili svježinom. Ovi kulturni i kontekstualni čimbenici mogu utjecati na našu interpretaciju boja i emocionalni odgovor na njih.

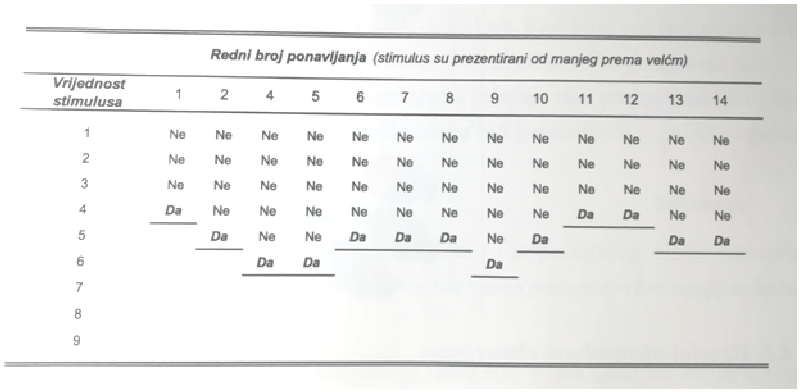
Vizualna percepcija boja ima široku primjenu u različitim područjima, uključujući umjetnost, dizajn, marketing i psihologiju. U umjetnosti i dizajnu, boje se koriste kako bi se postigao određeni estetski učinak i prenesla određena poruka. U marketingu, boje se koriste kako bi se privukla pažnja potrošača i utjecalo na njihove odluke. U psihologiji, boje se proučavaju kako bi se shvatilo kako boje utječu na naše raspoloženje, ponašanje i doživljaj okoline.



Slika 10 - Fotoreceptori u ljudskom oku (poglavlje 2.2.1.)



Slika 11 - Shematski prikaz izgleda i međupovezanosti fotoreceptora mrežnice



Redni broj ponavljanja (stimulus su prezentirani od manjeg prema većim)

Vrijednost stimulusa	1	2	4	5	6	7	8	9	10	11	12	13	14
1	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne
2	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne
3	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne
4	*Da*	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ne	*Da*	*Da*	Ne	Ne
5		*Da*	Ne	Ne	*Da*	*Da*	*Da*	Ne	*Da*			*Da*	*Da*
6			*Da*	*Da*				*Da*					
7													
8													
9													

Slika 12 - Primjer testiranja praga percepcije metodom limita

2.2.2. Psihologija boja

Psihologija boja je grana psihologije koja proučava kako boje utječu na naše raspoloženje, emocije, ponašanje i percepciju okoline. Boje imaju moćan utjecaj na našu psihu i mogu izazvati različite reakcije i doživljaje kod ljudi. Ova disciplina istražuje kako se boje percipiraju, kako ih doživljavamo i kako utječu na naše svakodnevno funkcioniranje.

Svaka boja ima svoju psihološku interpretaciju i može izazvati različite emocionalne reakcije. Evo nekoliko primjera:

1. Crvena boja: Crvena je boja energije, strasti i ljubavi. Ona može pobuditi uzbuđenje, povećati srčanu frekvenciju i adrenalin te potaknuti osjećaje ljubavi, ali i bijesa. Crvena se često koristi u marketinškim materijalima kako bi privukla pažnju i potakla akciju.

2. Plava boja: Plava je boja mira, smirenosti i stabilnosti. Ona može imati umirujući učinak na živčani sustav, smanjiti stres i potaknuti osjećaj ravnoteže. Plava boja se često koristi u prostorima za opuštanje i za meditaciju.

3. Zelena boja: Zelena je boja prirode, obnove i ravnoteže. Ona može djelovati umirujuće i osvježavajuće te potaknuti osjećaj povezanosti s prirodom. Zelena boja se često koristi u terapiji bojama kako bi se smanjio stres.

4. Žuta boja: Žuta je boja sunca, radosti i energije. Ona može izazvati osjećaj sreće, optimizma i vitalnosti. Žuta boja se često koristi u prostorima za poticanje kreativnosti i poboljšanje raspoloženja.

5. Narančasta boja: Narančasta je boja topline, entuzijazma i kreativnosti. Ona može potaknuti osjećaj veselja i uzbuđenja te poboljšati raspoloženje. Narančasta boja se često koristi u marketinškim kampanjama kako bi privukla pažnju i potaknula akciju.

Osim osnovnih boja, postoje i nijanse, tonovi i kombinacije boja koje mogu imati specifične psihološke učinke. Na primjer, ljubičasta boja može asocirati na eleganciju i tajanstvenost, dok smeđa boja može potaknuti osjećaj stabilnosti i prirodnosti.

Psihološki učinci boja mogu biti subjektivni i ovisiti o kontekstu, kulturi, osobnim iskustvima i preferencijama pojedinca. Neki ljudi mogu imati pozitivan doživljaj određene boje, dok drugi mogu imati negativan ili neutralan doživljaj. Psihologija boja ima široku primjenu u različitim područjima, uključujući dizajn interijera, marketing, oglašavanje, terapiju bojama i psihoterapiju. Razumijevanje psiholoških učinaka boja može pomoći u stvaranju željenih doživljaja i utjecaja na ciljanu publiku. Boje imaju moćan emocionalni i mentalni utjecaj na nas, a razumijevanje psiholoških učinaka boja može biti korisno u mnogim područjima našeg života. [**9**]

Slika 13 - Boje i emocije u marketingu (poglavlje 2.2.2.)

2.2.3. Testiranje osjetljivosti na nijanse boja

Testiranje osjetljivosti na nijanse boja je postupak koji se koristi kako bi se procijenila sposobnost pojedinca da razlikuje i identificira različite boje. Ova vrsta testiranja je od velike važnosti u različitim područjima, kao što su dizajn, umjetnost, medicina i industrija, gdje preciznost u percepciji boja može biti ključna. Testiranje osjetljivosti na nijanse boja provodi se uz pomoć posebnih testova koji se temelje na principu ispitivanja sposobnosti osobe da razlikuje različite boje ili identificira određene nijanse. Najčešći test koji se koristi je Ishihara test, koji se sastoji od tablica s različitim nizovima točkica različitih boja. Osoba koja se testira treba identificirati brojeve ili oblike koji su skriveni među točkicama. Na temelju toga može se procijeniti njihova sposobnost razlikovanja boja.

Osim Ishihara testa, postoje i drugi testovi koji se koriste za testiranje osjetljivosti na nijanse boja. Farnsworth-Munsell 100 Hue test je jedan od takvih testova. Ovaj test uključuje sortiranje seta boja u pravilnom redoslijedu. Osoba koja se testira mora postaviti niz boja od

najsvjetlije do najtamnije ili obrnuto. Na temelju rezultata testa može se procijeniti koliko dobro osoba percipira različite nijanse boja. [**10**]

Testiranje osjetljivosti na nijanse boja ima veliku važnost u dizajnu i umjetnosti. Dizajneri i umjetnici moraju biti u mogućnosti precizno razlikovati i koristiti različite nijanse boja kako bi postigli željeni vizualni učinak. Na primjer, u dizajnu web stranica ili grafičkom dizajnu, točna reprodukcija boja ključna je za postizanje vizualne privlačnosti i komunikaciju određenih poruka.

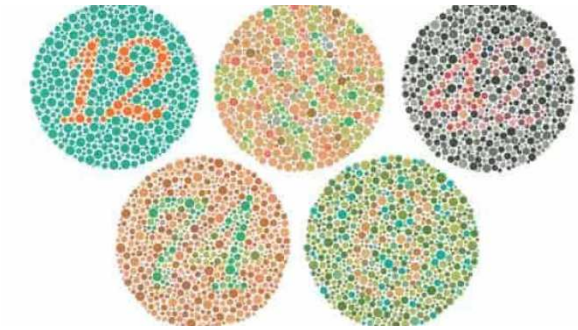
U medicini, testiranje osjetljivosti na nijanse boja može biti važno u dijagnostici određenih bolesti ili stanja. Kod određenih očnih bolesti ili stanja, kao što su glaukom ili makularna degeneracija, može doći do gubitka osjetljivosti na određene nijanse boja. Testiranje osjetljivosti na nijanse boja također je važna u industriji proizvodnje boja i pigmentiranih proizvoda gdje je važno osigurati konzistentnost boja i njihovu preciznu reprodukciju. Testiranje osjetljivosti na nijanse boja omogućuje provjeru kvalitete i usklađenosti boja prije nego što se proizvod plasira na tržište.

2.2.4. Ishihara test

Ishihara test je jedan od najpoznatijih testova za procjenu percepcije boja i otkrivanje kolorne sljepoće kod ljudi. Ovaj test je nazvan po japanskom oftalmologu Dr. Shinobu Ishihari koji ga je razvio 1917. godine. Glavni cilj Ishihara testa je identificirati osobe s određenim oblicima kolorne sljepoće ili smanjenom sposobnošću razlikovanja različitih boja. Kolorna sljepoća je genetski uvjetovano stanje koje utječe na percepciju boje. Osobe koje pate od kolorne sljepoće imaju poteškoća u razlikovanju ili identifikaciji određenih boja. Ishihara test se sastoji od serije tablica s različitim nizovima točkica različitih boja. Svaka tablica prikazuje skriveni broj ili oblik, koji osoba koja se testira treba identificirati. Brojevi ili oblici su prikazani pomoću točkica različitih boja i intenziteta. Ljudi s normalnom percepcijom boje lako mogu prepoznati skrivene brojeve ili oblike, dok osobe s kolornom sljepoćom mogu imati poteškoća u identifikaciji.

Tablice Ishihara testa osmišljene su na način da se testira osjetljivost na različite nijanse boja, kao i sposobnost razlikovanja boja koje su blisko srodne. Test obuhvaća širok spektar boja, uključujući crvenu, zelenu, plavu i njihove varijacije. Ovisno o rezultatima testa, moguće je identificirati vrstu i stupanj kolorne sljepoće ili smanjene percepcije boja. Ishihara

test je široko prihvaćen i koristi se diljem svijeta. Njegova jednostavnost korištenja i pristupačnost čine ga praktičnim alatom za prepoznavanje kolorne sljepoće u različitim kontekstima, uključujući medicinske ustanove, škole i industriju. Test se obično izvodi pod nadzorom stručnjaka za očne bolesti ili obučenih tehničara. [**11**]



Slika 14 - Ishihara test (poglavlje 2.2.4.)

2.2.5. Farnsworth-Munsell 100 Hue test

Farnsworth-Munsell 100 Hue test je standardni test za procjenu percepcije i razlikovanje nijansi boja kod ljudi. Ovaj test se često koristi u područjima kao što su oftalmologija, dizajn, industrija i istraživanje boje. Razvijen je 1943. godine od strane Dr. Dean Farnswortha i Dr. Dorothy Munsell. Cilj Farnsworth-Munsell 100 Hue testa je procijeniti osobu koliko dobro može sortirati niz od 85 obojenih pločica po rastućem ili opadajućem redoslijedu nijansi boja. Pločice se sastoje od različitih nijansi boja iz spektra, uključujući crvenu, žutu, zelenu, plavu i ljubičastu. Testiranje se obično provodi pod kontrolom stručnjaka, a osoba koja se testira mora organizirati pločice prema njihovim nijansama. Farnsworth-Munsell 100 Hue test je koncipiran tako da otkrije nijanse boja koje osoba možda ne može razlikovati ili percepirati. Test se temelji na Munsellovoj sistematskoj boji, koji je standardizirani sustav za kategorizaciju boja. Pločice u testu predstavljaju postupno mijenjajuće nijanse boje, što omogućuje ocjenjivanje osjetljivosti osobe na te nijanse. [**12**]

Rezultati Farnsworth-Munsell 100 Hue testa se analiziraju kako bi se procijenila osjetljivost osobe na različite nijanse boja. Osobe s normalnom percepcijom boje obično mogu precizno sortirati pločice prema njihovim nijansama, dok osobe s određenim oblicima kolorne sljepoće mogu imati poteškoća u sortiranju i mogu napraviti greške u redoslijedu. Farnsworth-Munsell 100 Hue test se koristi u različitim kontekstima. U oftalmologiji, test se

može koristiti za dijagnosticiranje i praćenje kolorne sljepoće i drugih poremećaja percepcije boja. Također se koristi u industriji boja, gdje je točnost percepcije boje od ključne važnosti za proizvodnju i ocjenu kvalitete boja. Dizajneri boja i umjetnici također mogu koristiti test kako bi procijenili svoju sposobnost razlikovanja i korištenja različitih nijansi boja u svojim djelima.

Slika 15 - Farnsworth Munsell 100 Hue test (poglavlje 2.2.5.)

3. Praktični dio

3.1. Proces izrade i implementacije

3.1.1. Projekt izrade i arhitektura računalne igre

Arhitektura igre je ključni koncept u procesu izrade računalne igre. Ona obuhvaća organizaciju, strukturu i dizajniranje komponenti igre kako bi se postigla funkcionalnost,

performanse i korisničko iskustvo. U kontekstu završnog rada "Izrada računalne igre za ispitivanje i vježbanje prepoznavanja nijansi boja " i procesu implementacija računalne igre, proces planiranja ima važnu ulogu.

Arhitektura igre uključuje različite aspekte koji su važni za uspješno izrađivanje i implementaciju igre:

1. Struktura igre: Struktura igre obuhvaća organizaciju različitih komponenti igre kao što su razine, izbornici, sučelje i ostali elementi koji čine igru. Strukturiranje igre olakšava navigaciju igračima i pruža im jasnu strukturu koja ih vodi kroz različite dijelove igre.

2. Game engine: Game engine je temeljni softverski okvir igre koji omogućava izvođenje, upravljanje i interakciju. To je skup alata, biblioteka i funkcionalnosti koji olakšavaju razvoj igre. Game engine pruža mogućnosti za upravljanje grafikom, fizikom, zvukom, umjetnom inteligencijom i drugim aspektima igre.

3. Komponente igre: Komponente igre su ključni elementi koji čine igru. To mogu biti likovi, objekti, oružja, okruženje, interaktivni elementi ili bilo koji drugi elementi s kojima igrači mogu interaktirati. Komponente igre trebaju biti dobro dizajnirane i implementirane kako bi pružile zanimljivo i izazovno iskustvo igračima.

4. Logika igre: Logika igre odnosi se na pravila, mehanike i ponašanje igre. To uključuje upravljanje događajima, bodovima, napretkom igrača, umjetnom inteligencijom neprijatelja i drugim elementima koji utječu na tijek i ishod igre. Logika igre osigurava dosljednost i ravnotežu igre te omogućava igračima da se angažiraju i postignu ciljeve.

5. Umjetnički dizajn: Umjetnički dizajn igre obuhvaća scenarij te vizualni i zvučni aspekt igre. Scenarij računalne igre čini temeljni okvir koji određuje priču, likove, događaje i ciljeve igre, pružajući igračima uzbudljivo i smisleno iskustvo. Uz scenarij je vezan dizajn likova, okruženja, tekstura, animacija, glazbe i zvukova. Umjetnički dizajn igre igra važnu ulogu u stvaranju atmosfere, estetike i emotivnog angažmana igrača.
U kontekstu završnog rada arhitektura igre je pažljivo osmišljena. To je uključivalo jasno definiranu strukturu igre s razinama napredovanja, dobro implementiranu logiku igre koja provjerava ispravnost prepoznavanja nijansi boja, pravilno balansiranje težine igre kako bi bila izazovna, ali ne i preteška, te atraktivan umjetnički dizajn koji potiče igrače da se angažiraju.
Kroz implementaciju i testiranje arhitekture igre "Color Sorting Game", moguće je dalje

proširiti projekt i pretvoriti ga u mobilnu aplikaciju za Android ili iOS. To bi omogućilo širem krugu korisnika da pristupaju igri putem svojih mobilnih uređaja i igraju je u pokretu. Mobilna aplikacija bi također omogućila integraciju dodatnih značajki kao što su praćenje napretka, natjecanja s drugim igračima, dostignuća i mogućnost dijeljenja rezultata na društvenim mrežama.

Slika 16 - Struktura gemifikacijskog pristupa

3.1.2. Izrada programskog koda "Color sorting game" računalne igre

Glavni kod igre uključuje korištenje JavaScript, Phaser.js, PHP, SQL i JSON tehnologije. Ove tehnologije omogućuju izradu dinamične i interaktivne igre s grafičkim prikazom, upravljanjem korisničkih interakcija, komunikacijom s bazom podataka i upravljanjem podacima. Glavni kod igre obuhvaća ključne komponente koje omogućuju funkcioniranje igre i implementaciju njenih funkcionalnosti.

1. Organizacija igre: Kod igre uključuje organizaciju igre u različite scene ili razine. Scene omogućuju prikaz različitih dijelova igre i omogućuju prijelaz između njih.

2. Upravljanje objektima: Kod igre sadrži upravljanje objektima koji se prikazuju na ekranu. To uključujestvaranje, premještanje, rotiranje ili mijenjanje veličine objekata.

3. Detekcija kolizija: Glavni kod igre obično uključuje algoritme za detekciju kolizija između objekata u igri. To omogućuje provjeru kada se dva objekta sudaraju ili kada se događa interakcija između njih.

4. Upravljanje korisničkim akcijama: Kod igre omogućuje prikupljanje korisničkih akcija kao što su pritiskanje tipki, klikovi mišem ili dodir na zaslonu. Ove akcije se koriste za upravljanje igrom i pokretanje različitih funkcionalnosti.

5. Spremanje podataka: Ukoliko je potrebno, kod igre može uključivati komunikaciju s bazom podataka kako bi se spremljeni podaci o igračima, rezultatima ili napretku u igri. Ovo omogućuje praćenje i pohranjivanje korisničkih podataka za daljnju upotrebu.

6. Integracija s vanjskim servisima: Glavni kod igre može sadržavati i integraciju s vanjskim servisima kao što su sustavi za autentifikaciju korisnika, analitika ili plaćanja.

Ove komponente glavnog koda igre su ključne za pravilno funkcioniranje i implementaciju igre koja je opisana u završnom radu.

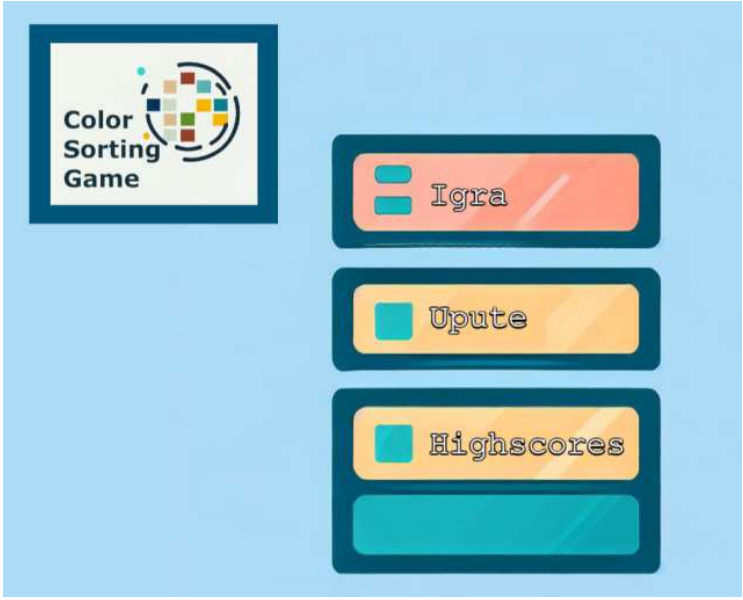
3.1.3. Opis logike igre "Color Sorting Game"

Igra sortiranja boja je zanimljiva i izazovna igra koja testira igračevu sposobnost prepoznavanja i sortiranja boja. Cilj igre je premjestiti boje na odgovarajuće mjesto na zaslonu kako bi se postigao pravilan niz nijansi boja. Igra se sastoji od više razina te se povećava složenost igre kako se napreduje kroz nivoe. Boje su predstavljene kao pravokutnici različitih boja koji se mogu povlačiti po zaslonu. Korisnik može odabrati jednu boju i premjestiti je na željeno mjesto tako da odabere dva obojena polja. Kada se boja povuče, igra provjerava je li boja na ispravnom mjestu prema zadanim pravilima sortiranja boja. Ako je boja ispravno postavljena, igrač osvaja bodove i prelazi na sljedeću boju. U suprotnom, boja ostaje na pogrešnom mjestu, a igrač mora pokušati ponovno. Korisničko sučelje igre je jednostavno i intuitivno. Na vrhu zaslona prikazuje se trenutni bodovi koje je igrač osvojio. Ispod bodova nalazi se pokazivač napretka. Boje su prikazane kao pravokutnici, a svaki pravokutnik ima jedinstvenu boju i oznaku koja je korisna za razlikovanje boja. Ispod pravokutnika boje nalaze se gumbovi za resetiranje i poništavanje poteza. Za sortiranje boja koristi se algoritam koji provjerava je li boja na ispravnom mjestu. Algoritam provjerava usporedbu trenutne boje s ciljnom bojom i provjerava je li ispravno sortirana. Ako je boja

ispravno sortirana, dodaju se bodovi i igra prelazi na sljedeću boju. Ako boja nije ispravno sortirana, korisnik može pokušati ponovno ili koristiti gumb za poništavanje poteza ("undo").

Igra ima više različitih nivoa, pri čemu svaki nivo predstavlja povećanje složenosti igre. U ranijim nivoima postoji manji broj boja koje treba sortirati, dok se u kasnijim nivoima povećava broj boja i složenost sortiranja. Nivoi se mogu postaviti pomoću varijable levelIndex koja se povećava kako igrač napreduje kroz nivoe.

Slika 17 - Glavni ekran aplikacije (poglavlje 3.1.3.)



Slika 18 - Izbornik aplikacije (poglavlje 3.1.3.)

3.2. Datoteke izvornog koda web aplikacije

3.2.1. Početni modul za pokretanje igre (index.hml)

`index.html` je osnovna HTML datoteka koja predstavlja početnu točku igre "Color Sorting Game". Ova datoteka definira strukturu dokumenta i uključuje potrebne skripte i stilove za pokretanje igre u web pregledniku.

Kada se otvori `index.html` u web pregledniku, prva linija `<!DOCTYPE html>` označava da je dokument tipa HTML5. Sljedeći redak `<html>` označava početak HTML dokumenta, dok se `</html>` koristi za označavanje kraja dokumenta. Unutar `<head>` elementa nalaze se informacije o dokumentu i uključivanje stilova. Redak `<meta charset="UTF-8">` specificira da se koristi UTF-8 kodiranje znakova, što osigurava pravilan prikaz različitih znakova na različitim jezicima. Redak `<title>Color Sorting Game</title>` definira naslov igre koji će se prikazati u naslovnoj traci preglednika.

Unutar `<style>` elementa definiramo stilove za dokument. U ovom slučaju, postavljamo marginu na 0 za tijelo dokumenta kako bi se uklonili prostori oko igre. Unutar `<body>` elementa nalazi se sadržaj igre. `<div id="game-container"></div>` predstavlja kontejner u koji će se smjestiti igra.

U nastavku, uključujemo skripte koje su potrebne za funkcioniranje igre. Linije koje počinju s `<script src="..."></script>` uključuju vanjske JavaScript datoteke potrebne za različite dijelove igre. Evo kratkog opisa tih skripti:

- `phaser-master/dist/phaser.min.js`: Ova skripta uključuje Phaser.js biblioteku, framework za izradu HTML5 igara.

- `leveli.js`: Skripta koja sadrži podatke o različitim razinama igre.

- `menu.js`: Skripta koja upravlja izbornikom igre.

- `display.js`: Skripta koja upravlja prikazom igre na ekranu.

- `game.js`: Skripta koja sadrži glavnu logiku igre.

- `upute.js`: Skripta koja prikazuje upute za igru.

- `restart.js`: Skripta koja upravlja ponovnim pokretanjem igre nakon što završi.

- `scores.js`: Skripta koja upravlja prikazom rezultata igre.

- `splash.js`: Skripta koja prikazuje uvodni zaslon razina igre.

- `config.js`: Skripta koja sadrži konfiguracijske postavke igre.

Korištenje ovih skripti omogućuje pokretanje igre "Color Sorting Game" i pruža sve potrebne funkcionalnosti za igrače.

3.2.2. Implementacija glavne logike igre (game.js)

U datoteci "game.js" nalazi se implementacija igre "Color Sorting Game". Igra je implementirana koristeći Phaser biblioteku za izradu igara u JavaScriptu. Cilj igre je sortirati boje na odgovarajuća mjesta na slikarskoj paleti.

U nastavku su opisani ključne dijelove implementacije igre u game.js:

- Kreiranje scene i globalnih varijabli: Inicijaliziraju se globalne varijable za upravljanje elementima igre kao što su boje, pločice, izbornik boje, preostalo vrijeme, bodovi itd. Također

se inicijalizira scena igre. Scena je osnovni grafički element u Phaseru koji omogućuje prikaz i upravljanje različitim dijelovima igre.

- Učitavanje grafičkih elemenata: Unutar funkcije `preload` učitavaju se slike koje se koriste u igri, kao što su slike pločica, slikarska paleta, overlay, gumbi itd.

- Kreiranje scene: Temeljni element prikaza phaser.js frameworka su scene. U funkciji `create` postavljaju se elementi igre i interaktivnosti. Postavlja se pozadina scene, stvaraju se pločice boja i dodjeljuje im se interaktivnost. Kreiraju se gumbi za poništavanje posljednje zamjene boja. Također se priprema overlay za označavanje odabrane pločice boje.

- Glavne funkcije iz datoteke game.js:

- `resetLevel`: Resetira trenutni level igre.

- `createSwatches`: Stvara pločice boja na slikarskoj paleti.

- `createSwatch`: Stvara pojedinačnu pločicu boje na određenoj poziciji.

- `selectSwatch`: Odabire pločicu boje na klik i omogućuje zamjenu s drugom odabranom pločicom.

- `swapSwatches`: Zamjenjuje dvije odabrane pločice boja.

- `undoSwatchMovement`: Poništava posljednju zamjenu boja.

- `updateGame`: Ažurira stanje igre.

- `gameOver`: Prikazuje poruku o završetku igre.

3.2.3. Definiranje razina i njihovih karakteristika (leveli.js)

"leveli.js" je JavaScript datoteka koja sadrži podatke o različitim levelima u igri sortiranja boja. Ova datoteka je ključna za definiranje svakog nivoa, uključujući broj redova i stupaca, boje koje se koriste, te vremensko ograničenje za svaki nivo. Podaci o levelima su organizirani u polju (levelPolje) koje sadrži objekte za svaki level. Svaki objekt predstavlja jedan level i ima sljedeće ključeve:

- level: označava broj levela.

- rows: označava broj redova u mreži boja za level.

- columns: označava broj stupaca u mreži boja za level.

- colors: predstavlja matricu boja za svaki red i stupac u matrici boja. Boje se definiraju u heksadecimalnom formatu.

- timeLimit: označava vremensko ograničenje za level, tj. koliko vremena igrač ima za sortiranje boje.

Svaka razina ima drugačiju konfiguraciju matrice boja, uključujući različit broj redova i stupaca te različite kombinacije boja. Također, datoteka sadrži i varijablu "tekstUvod" koja sadrži niz tekstova koji se koriste za prikazivanje uvoda za svaku razinu. Svaki tekst u nizu opisuje temu ili okruženje odgovarajućeg levela i daje igraču kontekst za taj nivo.

3.2.4. Modul za funkcionalnost glavnog izbornika (menu.js)

JavaScript datoteka "menu.js" implementira glavni izbornik igre. Ova datoteka je odgovorna za prikazivanje glavnog izbornika igre, kao i za reagiranje na korisničke interakcije s tipkama izbornika.

Unutar funkcije "preload" učitavaju potrebne slike za pozadinu izbornika i logotip u objekat "assets". Pretpostavlja se da se slike nalaze u odgovarajućem direktoriju na serveru.

Funkciji create() definira scenu glavnog izbornika. Dodaje se slika pozadine na scenu koristeći add.image() metodu. Slika se postavlja na poziciju (0, 0) i postavlja se da ima porijeklo (origin) u gornjem lijevom kutu.

Nadalje, kreiraju se tipke za glavni izbornik:

- Tipka "Igra" (gameButton): Prikazuje se tekstualna tipka "Igra" na određenoj poziciji i veličini. Kada se klikne na tipku pokreće se scena "GameScene" koja predstavlja igru.

- Tipka "Upute" (uputeButton): Prikazuje se tekstualna tipka "Upute" na određenoj poziciji i veličini. Kada se klikne na tipku pokreće se scena "uputeScene" koja prikazuje upute za igru.

- Tipka "Highscores" (scoresButton): Prikazuje se tekstualna tipka "Highscores" na određenoj poziciji i veličini. Kada se klikne na tipku pokreće se scena "scoresScene" koja prikazuje rezultate igre.

Svaka od ovih tipki je interaktivna i reagira na događaj "pointerup" (kada se klikne na tipku). Ovisno o tipki koja se klikne, odgovarajuća scena se pokreće, a scena glavnog izbornika postaje nevidljiva. Modul "menu.js" omogućava igračima da navigiraju kroz glavni izbornik igre i odaberu željenu opciju, bilo da je to pokretanje igre, pregled uputa ili provjera najboljih rezultata. Pruža intuitivan sučelje i interaktivnost kako bi korisnici mogli lako upravljati igrom.

3.2.5. Modul za prikazivanje elemenata na zaslonu (display.js)

Datoteka "display.js" implementira funkcionalnosti za prikaz i ažuriranje grafičkih elemenata na zaslonu u igri. Ova datoteka sadrži različite funkcije koje se koriste za stvaranje i ažuriranje elemenata poput trake napretka, pješčanog sata, prikaza života i teksta na zaslonu. U datoteci su definirane varijable koje sadrže boje, pozicije i dimenzije elemenata na zaslonu. Na primjer, varijable poput "pbBgColor" (boja pozadine trake napretka), "pb_x" i "pb_y" (pozicija trake napretka) određuju izgled i položaj elementa na zaslonu. Funkcija "progressBarKreiraj()" koristi ove varijable za stvaranje trake napretka. U ovoj funkciji se dodaje pozadinski pravokutnik i dinamički pravokutnik koji predstavlja pokazivač napretka. Također se dodaje slika koja se prikazuje iznad trake napretka. Funkcija "progressBarUpdate()" se koristi za ažuriranje trake napretka. U ovoj funkciji se mijenja duljina dinamičkog pravokutnika ovisno o postotku napretka, a također se mijenja i prozirnost pozadinske slike.

Slično tome, funkcije "pjescaniKreiraj()" i "pjescaniUpdate()" se koriste za stvaranje i ažuriranje pješčanog sata na zaslonu. Pješčani sat se sastoji od pozadine i "sand fill"-a koji se postupno smanjuje kako vrijeme prolazi.

Funkcije "livesCreate()" i "livesUpdate()" se koriste za stvaranje i ažuriranje prikaza života na zaslonu. Ove funkcije koriste petlje za postavljanje i mijenjanje boje ikona života ovisno o broju izgubljenih života.

Funkcije "dispTekstKreiraj()" i "dispTekstUpdate()" se koriste za stvaranje i ažuriranje teksta na zaslonu. Ove funkcije dodaju tekstualne elemente poput bodova, razine, preostalog vremena i broja ispravno složenih elemenata.

Za pokretanje i ažuriranje odbrojavanja vremena koriste se funkcije "startCountdown()" i "timerCall()". One smanjuju preostalo vrijeme i ažuriraju prikazanu vrijednost na zaslonu.

Kada vrijeme istekne, izvršava se funkcija "istekloVrijeme()" koja provjerava broj izgubljenih života i ovisno o tome donosi odluku o završetku igre ili nastavku razine.

Funkcija "spremiRezultat()" se koristi za spremanje rezultata igre u bazu podataka. Generira se jedinstveni identifikator igre i korisnika te se rezultat šalje na poslužitelj radi spremanja. Ako igrač ne unese svoje ime, rezultat se privremeno sprema bez imena.

Za ponovno pokretanje razine nakon gubitka života koristi se funkcija "restartLevelPritisnuto()". Ova funkcija sakriva dijalog za ponovno pokretanje i omogućuje igraču da nastavi igru s preostalim životima. Navedene funkcije omogućuju stvaranje i ažuriranje različitih grafičkih elemenata na zaslonu tijekom igre.

3.2.6. Implementacija prikaza uputa za igru (upute.js)

Za implementaciju prikaza uputa za korištenje igre koristi se javascript datoteka "upute.js". Na početku se definira nova scenu pod nazivom "uputeScene". Unutar funkcije "preload()", koja se izvršava prije pokretanja scene, učitavaju se potrebne slike i tekstualni sadržaj. U ovom slučaju učitava se tekstualna datoteka "upute.txt" koja sadrži detaljne upute za igru. U funkciji "create()", koja se izvršava prilikom stvaranja scene, postavljaju se grafički elementi i manipulacije koji čine prikaz uputa. Prvo se dodaje pozadinska slika na scenu s postavkom da se prikaže na poziciji (0, 0) i da bude poravnata u gornjem lijevom kutu. Zatim se stvara pravokutnik koji će služiti kao pozadina za tekst uputa. Pravokutnik se stvara pomoću objekta "pozadina" koji koristi metodu "fillStyle" i postavlja boju na crnu (0x000000). Nakon toga, poziva se metoda "fillRect" koja stvara pravokutnik na poziciji (110, 20) sa širinom od 800 piksela i visinom od 700 piksela. Da bi pravokutnik izgledao poluprovidno, postavlja se njegova dubina (depth) na 0 i transparentnost (alpha) na 0.8. Sljedeći korak je stvaranje teksta koji će prikazivati upute. Pomoću metode "add.text" dodaje se tekstualni element na scenu. Postavljaju se različiti parametri teksta kao što su font, veličina, boja i postavka omotača teksta. Tekst se prikazuje na poziciji (140, 50) i koristi se Arial font veličine 18 piksela. Boja teksta postavljena je na bijelu (#ffffff). Wrapping teksta je postavljen na širinu od 750 piksela kako bi se osiguralo da se tekst pravilno prikaže unutar pravokutnika.

Sljedeći korak je učitavanje sadržaja uputa iz tekstualne datoteke "upute.txt". Sadržaj se dohvaća iz cache-a pomoću metode "this.cache.text.get('upute')" i postavlja se kao tekst sadržaj teksta koji se prikazuje. Kako bi se omogućilo skrolanje teksta ako je prevelik za

prikaz unutar pravokutnika, tekst se postavlja da koristi masku. Masku se stvara pomoću objekta "maskGraphics" koji koristi metodu "createGeometryMask" i postavlja je na tekstualni element pomoću metode "setMask".

Dodana je i funkcionalnost skrolanja teksta pomoću kotačića miša. Kada korisnik pomiče kotačić miša prema gore, tekst se pomjeri prema dolje kako bi se prikazao viši dio teksta. Kada korisnik pomiče kotačić miša prema dolje, tekst se pomjeri prema gore kako bi se prikazao niži dio teksta. To je postignuto dodavanjem događaja "wheel" na ulazni sustav (input) i postavljanjem odgovarajuće logike pomjeranja teksta ovisno o smjeru pomicanja kotačića miša.

Također, u funkciji "create()" se poziva i funkcija "kreirajTipkaPovratakS()" koja je odgovorna za stvaranje tipke za povratak na glavni izbornik. Tipka se stvara pomoću grafičkog elementa "izlazButton" koji predstavlja pravokutnik s crvenom bojom. Tekst "Glavni izbornik" se dodaje na tipku, a postavlja se i interaktivnost na tipku kako bi reagirala na klik korisnika. Kada korisnik klikne tipku, poziva se funkcija "povratakGlavniIzbornikU()" koja se odnosi na povratak na glavni izbornik igre.

Funkcija "povratakGlavniIzbornikU()" jednostavno ponovno učitava stranicu, što rezultira povratkom na glavni izbornik igre.

Da bi se sve ove funkcionalnosti uspješno izvele, potrebno je da datoteka "upute.js" bude povezana s drugim dijelovima igre i da se izvršava u kontekstu odgovarajućeg okruženja.

3.2.7. Modul za ponovno pokretanje razine (restart.js)

Kako bi se nakon isteka vremena preko postojeće "game" scene jednostavno prikazala tipka za "restart" levela preko postojeće scene kreirao sam posebnu scenu "restart.js". Jedini zadatak ove scene je prikaz tipke i ponovno pokretanje razine nakon pritiska na tipku

Prvo se postavlja scena na neaktivnu i nevidljivu pomoću metoda "this.scene.setActive(false)" i "this.scene.setVisible(false)". Ovo osigurava da se scena ne prikazuje i ne reagira na događaje dok nije aktivirana. Također, omogućava se onemogućavanje ulaznog sustava pomoću "this.input.enabled = false" kako bi se spriječilo bilo kakvo interakciju korisnika s igrom dok je ponovno pokretanje razine prikazano.

Zatim se stvara maska koja definira vidljivo područje na ekranu. Maska se stvara pomoću objekta "maskGraphics" koji koristi metodu "fillStyle" za postavljanje boje maske na crvenu (0xdd0000). Nakon toga, metodom "fillRect" stvara se pravokutnik na poziciji (550, 350) sa širinom od 250 piksela i visinom od 50 piksela. Ova maska definira područje na kojem će se prikazivati ponovno pokretanje razine. Da bi se primijenila maska na cijelu "restart overlay" scenu, koristi se metoda "this.cameras.main.setMask(mask)" koja postavlja masku na glavnu kameru scene. Nakon toga, stvara se tekstualni element "restartButton" koji predstavlja tipku za ponovno pokretanje razine. Tekst se prikazuje na poziciji (670, 370) s tekstom "Restart level" veličine fonta 26 piksela i bijele boje. Tipka je centrirana postavkom "setOrigin(0.5)". Također, postavlja se interaktivnost na tipku pomoću "restartButton.setInteractive()" kako bi reagirala na klik korisnika.

3.2.8. Modul za prikazivanje, ažuriranje i spremanje rezultata igrača (scores.js)

Ova datoteka je ključna za praćenje postignuća igrača i omogućava im da vide svoje bodove, razinu i datum rezultata. Na početku se definira nova scena pod nazivom "scoresScene" pomoću objekta "Phaser.Scene". Ova scena će biti odgovorna za prikazivanje rezultata igrača. U funkciji "preload()", koja se izvršava prije nego što se scena kreira, učitavaju se potrebni resursi. Koristi se metoda "this.load.json()" za učitavanje JSON datoteke "textData" pomoću "readscores.php" koji sadrži podatke o rezultatima igrača. Također se učitava slika "backgroundhs" koja predstavlja grafičku pozadinu.

U funkciji "create()", koja se izvršava prilikom stvaranja scene, postavljaju se elementi za prikazivanje rezultata igrača. Prvo se dodaje slika pozadine na poziciji (0, 0) pomoću "this.add.image()". Zatim se stvara pravokutnik pozadine pomoću objekta "pozadina" i metode "fillStyle" koja postavlja boju na crnu (0x000000). Pravokutnik se crta na poziciji (110, 20) sa širinom od 800 piksela i visinom od 700 piksela. Pravokutnik se postavlja na dubinu 0 i transparentnost se postavlja na 0.8 pomoću "pozadina.setDepth(0).setAlpha(0.8)".

Nakon postavljanja pozadine, poziva se funkcija "kreirajTipkaPovratakS()" koja je odgovorna za stvaranje tipke za povratak na glavni izbornik. Ova funkcija će kasnije biti objašnjena detaljnije. Nakon toga, stvaraju se tekstualni elementi za prikaz naslova "HighScores" i zaglavlja tablice rezultata. Naslov se prikazuje na poziciji (510, 50) s fontom

veličine 36 piksela i bijelom bojom. Zaglavlje se prikazuje na poziciji (510, 100) s fontom veličine 24 piksela i bijelom bojom.

Zatim se koristi objekt "XMLHttpRequest" za izvršavanje AJAX zahtjeva prema PHP skripti "readscores.php" kako bi se dohvatili podaci o rezultatima iz baze podataka. Kada se dobiju podaci, koristi se metoda "JSON.parse()" za parsiranje JSON odgovora i dobivanje podataka u obliku JavaScript objekta. Nakon toga, petljom se prolazi kroz dobivene podatke i za svaki red podataka se kreira tekstualni objekt koji prikazuje redni broj, ime igrača, bodove, razinu i datum postignuća. Tekstualni objekti se postavljaju na odgovarajuće pozicije na temelju rednog broja koristeći "this.add.text()" metodu.

Datoteka također sadrži funkcije koje se koriste za slanje podataka na server. Funkcija "saveScore()" koristi se za spremanje rezultata igrača u bazu podataka. Ona priprema podatke, kao što su ID igre, ID kolačića, bodovi, razina i trenutni datum, i šalje ih na PHP skriptu "savescores.php" pomoću POST zahtjeva. Funkcija "updateScore()" koristi se za ažuriranje rezultata igrača u bazi podataka. Ona prima ID igre i ime igrača te šalje te podatke na PHP skriptu "updatescores.php" pomoću POST zahtjeva. Na taj način implementirao sam spremanje rezultat igre i u slučaju da korisnik ne upiše ime.

3.2.9. Prikazivanje uvodne slike i teksta na početku svake razine (splash.js)

"splash.js" je JavaScript datoteka koja se koristi u igri Color Sorting za prikazivanje uvodne slike i teksta prije početka svake razine. Ova datoteka je odgovorna za stvaranje i postavljanje elemenata za prikazivanje uvodne slike, teksta i pozadine na početku svake razine. Na početku datoteke, stvara se nova scena pod nazivom "splashScene" pomoću objekta "Phaser.Scene". Ova scena će biti odgovorna za prikazivanje uvodne slike i teksta prije početka svake razine. U funkciji "preload()", koja se izvršava prije nego što se scena kreira, učitava se slika pozadine razine "level5_bg.png" pomoću metode "this.load.image()". U funkciji "init(dataReceived)", koja se izvršava prilikom inicijalizacije scene, prihvača se primljeni podatak "indeksLevela" koji označava trenutni indeks razine. Ovaj podatak će se koristiti kasnije za odabir odgovarajuće slike pozadine.

U funkciji "create()", koja se izvršava prilikom stvaranja scene, postavljaju se elementi za prikazivanje uvodne slike i teksta. Ovisno o vrijednosti "indeksLevela", odabire se odgovarajuća slika pozadine i stvara se objekt "splashImage" pomoću "this.add.image()" metode. Slika se postavlja na poziciju (0, 0) i postavlja se da ima porijeklo (origin) u gornjem lijevom kutu.

Zatim se stvara tekstualni objekt "splashText" koji prikazuje informaciju o trenutnom nivou. Tekst se postavlja na poziciju (535, 384) s fontom veličine 90 piksela, bijelom bojom teksta, crnom bojom obruba teksta i debljinom obruba teksta postavljenom na 4 piksela. Tekst se također postavlja da ima porijeklo (origin) na sredini. Nakon toga, stvara se pravokutnik "splashTextPozadina" koji će poslužiti kao pozadina za tekst. Pravokutnik se crta pomoću "splashTextPozadina.fillStyle()" metode i postavlja se boja na crnu (0x000000) s transparentnošću od 0.9. Pravokutnik se crta na poziciji (350, 500) s širinom od 700 piksela i visinom od 250 piksela. Sljedeći korak je stvaranje tekstualnog objekta "splashTextArea" koji prikazuje uvodni tekst za određeni nivo. Tekst se postavlja na poziciju (384, 520) s fontom "Arial", veličinom 24 piksela, crnom bojom obruba teksta, debljinom obruba teksta postavljenom na 5 piksela i word wrapper koji ograničava širinu teksta na 650 piksela. Nakon stvaranja elemenata za prikaz, postavljaju se dubine elemenata pomoću metode "setDepth()" kako bi se osiguralo da se elementi prikazuju na najgornjem sloju. Sve elemente postavljamo na vrijednost "Number.MAX_SAFE_INTEGER" kako bi bili prikazani iznad ostalih elemenata na sceni. Konačno, pomoću "this.time.delayedCall()" funkcije, nakon četiri sekunde, svi elementi uvodne slike i teksta se uništavaju pomoću metode "destroy()" kako bi se oslobodili resursi i pripremili scenu za početak razine.

3.2.10. Definiranje osnovnih postavki phaser.js scena i fizike (config.js)

"config.js" je datoteka koja se koristi za konfiguriranje osnovnih postavki igre. Ova datoteka sadrži objekt "config" koji definira različite parametre igre kako bi se postigli željeni rezultati. U nastavku su navedene ključne postavke prisutne u "config.js" datoteci:

- "type": Ova postavka određuje tip igre koji će se koristiti. U ovom slučaju, postavka je postavljena na "Phaser.AUTO". To znači da će Phaser automatski odabrati najprikladniji način za prikazivanje igre na različitim platformama i uređajima.
- "width" i "height": Ove postavke određuju širinu i visinu prozora igre. U primjeru su postavljene na 1024 piksela širine i 768 piksela visine.

- "physics": Ova postavka definira fiziku igre koja će se koristiti. U konkretnom primjeru, koristi se "arcade" fizika. Arcade fizika pruža jednostavne i brze kolizije te omogućuje postavljanje gravitacije. Gravitacija je postavljena na nulu (y: 0), što znači da nema gravitacije u igri. Opcija "debug" je postavljena na "false", ali ako se postavi na "true", prikazivat će se korisne informacije o detekciji sudara za potrebe ispravljanja grešaka i ispitivanja igre.
- "scene": Ova postavka određuje koje scene će biti uključene u igru. U primjeru su navedene različite scene kao popis objekata: mainMenuScene, gameScene, uputeScene, restartScene, scoresScene i splashScene. Svaka od ovih scena obavlja specifičnu funkcionalnost u igri, poput glavnog izbornika, igre, uputa, ponovnog pokretanja i prikaza splash ekrana.

- "scale": Ova postavka određuje skaliranje igre kako bi se prilagodila veličini prozora ili uređaja na kojem se igra izvodi. U primjeru se koristi "Phaser.Scale.RESIZE" mod, što znači da će se igra automatski prilagoditi veličini prozora. "parent" postavka određuje HTML element koji će biti roditeljski element igre, dok "autoCenter" postavka centriranje igre na zaslonu.

Nakon definiranja objekta "config", stvara se nova instanca Phaser igre koristeći prethodnu konfiguraciju. Kroz ovaj objekt, Phaser primjenjuje sve navedene postavke i omogućuje sa se igra prilagodi specifičnim zahtjevima i potrebama.

3.2.11. Modul za čitanje rezultata iz baze (readscores.php)

"readscores.php" je PHP skripta koja se koristi u igri Color Sorting za čitanje rezultata iz baze podataka i vraćanje podataka u formatu JSON. Ova skripta omogućuje prikazivanje najboljih rezultata igrača na rang-listi. Skripta započinje definiranjem parametara za uspostavljanje veze s bazom podataka. Varijable poput "$servername", "$username", "$password" i "$dbname" koriste se za povezivanje s odgovarajućom bazom podataka. U primjeru se koristi lokalni poslužitelj i odgovarajući pristupni podaci. Zatim se kreira nova instanca mysqli objekta za uspostavljanje veze s bazom podataka. Ako veza nije uspješna, skripta završava izvršavanje prikazivanjem poruke o pogrešci. Nakon uspješne uspostave veze, izvršava se SQL upit koji dohvaća podatke iz tablice "rezultati". Upit izvlači podatke kao što su "playername" (ime igrača), "bodovi" (osvojeni bodovi), "level" (postignuti level) i "datum" (datum postizanja rezultata). Podaci se sortiraju prema osvojenim bodovima, s najvećim rezultatima prvo.

Ako se u rezultatima upita nalaze zapisi, rezultati se spremaju u polje "tableData" kao asocijativni niz. Svaki redak rezultata se dodaje u polje. Nakon toga, skripta koristi funkciju "json_encode" kako bi pretvorila podatke iz polja u JSON format. JSON format omogućuje lako razumijevanje podataka i njihovu daljnju obradu u JavaScriptu. Ako u rezultatima upita nema zapisa, skripta ispisuje poruku "No data found". Na kraju se veza s bazom podataka zatvara pozivom metode "close" na objektu veze. Ova skripta se koristi u kombinaciji s JavaScriptom na klijentskoj strani kako bi se prikazali najbolji rezultati igrača na rang-listi.

3.2.12. Modul za spremanje rezultata u bazu (savescores.php)

"savescores.php" je PHP skripta koja se koristi za spremanje rezultata igrača u bazu podataka. Ova skripta omogućuje bilježenje rezultata, uključujući bodove, level, datum, IP adresu i ostale informacije koje se koriste za praćenje i analizu rezultata.

Skripta također započinje definiranjem parametara za uspostavljanje veze s bazom podataka, kao što su "$servername", "$username", "$password" i "$dbname". U primjeru se koristi lokalni poslužitelj i odgovarajući pristupni podaci. Nakon toga, skripta prima podatke putem POST metode iz igre, kao što su "gameid" (identifikator igre), "cookieid" (identifikator kolačića), "bodovi" (osvojeni bodovi) i "level" (postignuti level). Također, skripta generira trenutni datum i vrijeme koristeći funkciju "date" kako bi bilježila vrijeme postizanja rezultata. Također, zabilježena je i IP adresa igrača koristeći varijablu "$_SERVER['REMOTE_ADDR']".

Varijabla "$playername" postavlja se na vrijednost "-" kako bi se izbjeglo pohranjivanje praznih vrijednosti u bazu podataka. Nakon prikupljanja podataka, skripta uspostavlja vezu s bazom podataka i provodi upit za unos novog zapisa u tablicu "rezultati". Podaci se umetaju u odgovarajuće stupce, a varijable se interpoliraju u SQL upitu. Nakon što je upit izvršen, veza s bazom podataka se zatvara pozivom funkcije "mysqli_close".

3.2.13. Modul za ažuriranje podataka u bazi (updatescores.php)

"updatescores.php" je PHP skripta koja se koristi u igri Color Sorting za ažuriranje imena igrača u bazi podataka. Ova skripta omogućuje upis imena igrača povezanog s određenim rezultatom. Skripta također započinje definiranjem parametara za uspostavljanje veze s bazom podataka, kao što su "$servername", "$username", "$password" i "$dbname". U primjeru se

koristi lokalni poslužitelj i odgovarajući pristupni podaci. Nakon toga, skripta prima podatke putem POST metode iz igre, kao što su "gameid" (identifikator igre) i "playername" (novo ime igrača). Ovi podaci se koriste za ažuriranje odgovarajućeg zapisa u tablici "rezultati". Skripta uspostavlja vezu s bazom podataka i provodi upit za ažuriranje podataka. Koristi se SQL naredba "UPDATE" kako bi se postavilo novo ime igrača ("playername") za određeni "gameid". Varijable se interpoliraju u SQL upitu. Nakon što je upit izvršen, veza s bazom podataka se zatvara pozivom funkcije "mysqli_close".

4. Analiza rezultata

4.1. Metodologija testiranja vizualnih sposobnosti korisnika

Kroz rezultate iz igre Color Sorting Game omogućuje se prikupljanje relevantnih podataka koji se mogu koristiti za analizu i procjenu korisničkih vizualnih sposobnosti. Kroz rezultate iz SQL tablice "rezultati", moguće je dobiti sljedeće informacije:

1. Bodovi: Rezultati igre sadrže informacije o osvojenim bodovima za svakog korisnika. Ovi bodovi mogu biti indikativni za korisnikovu sposobnost brzog i točnog sortiranja boja. Analiziranjem bodova možemo utvrditi kojim korisnicima igra najbolje ide i tko postiže visoke rezultate.

2. Razina (Level): Informacija o razini na kojoj je korisnik završio igru pruža uvid u korisnikovu sposobnost napredovanja kroz igru. Može se pratiti koliko je korisnika uspješno završilo određene razine ili postiglo visoke razine.

3. Datum: Rezultati igre također sadrže informaciju o datumu kada je rezultat postignut. Ova informacija omogućuje praćenje vremenskih trendova i usporedbu performansi korisnika tijekom određenog vremenskog razdoblja.

4. Korisničko ime (Playername): Kroz praćenje cookieid-a ili ukoliko korisnik unese svoje ime prilikom spremanja rezultata, može se pratiti performanse svakog korisnika pojedinačno.

Kombiniranjem ovih informacija, moguće je izvršiti detaljnu analizu korisničkih sposobnosti u igri Color Sorting Game. Primjerice, moguće je identificirati korisnike s najvišim bodovima ili najvišim razinama, pratiti napredak korisnika tijekom vremena i usporediti rezultate među korisnicima. Ovi podaci mogu poslužiti za donošenje odluka o

prilagodbi dizajna igre kako bi se poboljšala korisnička interakcija i optimizirao korisnički doživljaj. Također, rezultati mogu poslužiti kao povratna informacija za korisnike kako bi se potaknula natjecateljska atmosfera i motivirali korisnici na daljnje igranje igre.

4.1. Statistika podataka dobivenih u tablici rezultata

Analizirajući podatke, možemo dobiti dublje razumijevanje korisničkih navika, trendova i uspješnosti, te ih možemo koristiti za poboljšanje dizajna igre i korisničkog iskustva. U nastavku su opisane neke statističke metode koje se mogu primijeniti na podatke iz tablice rezultata:

1. Ukupan broj igrača: Analizirajući broj redaka u tablici rezultata, možemo saznati koliko je ukupno igrača sudjelovalo u igri. Ova informacija pruža uvid u popularnost igre i njezin doseg među korisnicima.

2. Prosječni bodovi: Izračunavanje aritmetičke sredine bodova svih igrača omogućuje nam se razumijevanje općeg uspjeha korisnika u igri. Možemo saznati koliko su korisnici uspješni u postizanju visokih bodova i procijeniti razinu izazova koju igra pruža. Također bismo mogli odrediti i druge srednje vrijednosti kao što su mod, medijan, kvartili...

3. Najviši bodovi: Identifikacija najviših bodova postignutih od strane pojedinih igrača omogućuje nam prepoznavanje najuspješnijih igrača. To nam omogućuje uspostavljanje ljestvice najboljih igrača i natjecateljske atmosfere koja može potaknuti korisnike na daljnje igranje i postizanje visokih rezultata.

4. Raspon bodova: Proračunavanje raspona bodova, odnosno razlike između najviših i najnižih bodova, pruža nam informaciju o varijaciji uspjeha među igračima. Veći raspon ukazuje na veću raznolikost performansi korisnika i različitu razinu vještina među igračima.

5. Vremenska analiza: Analiziranje datuma rezultata omogućuje nam praćenje promjena performansi tijekom vremena. Možemo identificirati trendove u korisničkim performansama, kao što su poboljšanja ili pogoršanja rezultata s vremenom.

6. Distribucija rezultata: Prikazivanje distribucije rezultata u obliku histograma ili grafikona omogućuje nam uvid u frekvenciju postizanja određenih bodova ili razina. Ovo nam pomaže razumjeti kako su korisnici raspoređeni po različitim kategorijama.

7. Povezanost bodova i razina: Analiza povezanosti između bodova i razina omogućuje nam razumijevanje koliko je postizanje visokih bodova povezano s napredovanjem kroz razine igre. Možemo utvrditi ima li korisnička uspješnost veze s napredovanjem kroz igru i jesu li korisnici s višim razinama skloniji postizanju većih bodova.

Ove statističke metode omogućuju bolje razumijevanje korisničkih performansi u igri Color Sorting Game i pružaju uvide koji mogu biti korisni za daljnje prilagođavanje i poboljšanje igre. Analiziranjem statističkih podataka, možemo identificirati obrasce, tendencije i slabosti u korisničkim performansama te donijeti odluke o optimizaciji korisničkog iskustva.

4.2. Daljnji razvoj i mogućnosti

Daljnji razvoj i mogućnosti Color Sorting Game nude niz perspektiva za unapređenje igre i proširenje korisničkog iskustva. U nastavku bih istaknuo samo neke:

1. Dodavanje novih razina i mehanike igre: Jedan od načina za daljnji razvoj igre je dodavanje novih razina s različitim izazovima. Mogao bih dodati nove boje, složenije uzorake ili posebne bonuse koji povećavaju bodove ili pružaju dodatne pogodnosti igračima. Ovo bi pružalo novi izazov i držalo igrače zainteresiranima dulje vrijeme.

2. Poboljšanje vizualne prezentacije: U razvoju igre dalje bih mogao unaprijediti vizualnu prezentacijui poboljšanjem grafike, animacija i dodavanjem zvučnih efekata kako bi se stvorila još atraktivnija i uzbudljivija igra.

3. Uvođenje više načina igre: Trenutna verzija Color Sorting Game fokusira se na sortiranje boja, ali postoji mogućnost proširenja načina igre. Na primjer, moguće je dodati načine poput brzinske igre, preživljavanja ili multiplayer opcija koje omogućuju korisnicima da se natječu ili surađuju s drugim igračima.

4. Implementacija sustava prijave korisnika: Dodavanje sustava prijave za praćenje postignuća i rangiranja igrača potaknulo bi korisnike da se vrate i nastave igrati kako bi postigli bolje rezultate i time bi se stvorio natjecateljski duh.

5. Integracija društvenih mreža: Ako bih omoćio korisnicima da dijele svoje rezultate i postignuća na društvenim mrežama to bi pomoglo u promociji igre i privlačenju novih korisnika. Integracija s društvenim mrežama omogućila bi korisnicima da se povežu s prijateljima, podijele svoje uspjehe i pozovu druge da se pridruže igri.

6. Mobilna verzija igre: Color Sorting Game ima potencijal za razvoj mobilne verzije, koja omogućuje korisnicima da igraju igru na svojim pametnim telefonima i tabletima. Mobilna verzija omogućuje veću dostupnost i prilagođenost korisnicima koji preferiraju mobilno igranje.

5. Zaključak

U konačnici mogu istaknuti nekoliko postignuća i sažeti glavne zaključke koji proizlaze iz istraživanja, pripreme i razvoja ove igre. Uspješno sam razvio jednostavnu logičku računalnu igru koja kombinira zabavno iskustvo igranja s ciljem poticanja vizualnih sposobnosti korisnika. Kroz implementaciju različitih mehanika igre, kao što su sortiranje boja i napredak kroz razine, stvorio smo izazovno okruženje koje korisnike potiče na koncentraciju, brzu reakciju i analitičke vještine. Primijenio sam metodologiju testiranja vizualnih sposobnosti korisnika kako bih prikupio relevantne podatke o njihovim rezultatima. Kroz rezultate iz SQL tablice "rezultati" dobili smo vrijedne informacije o bodovima, razinama, vremenima i ostalim parametrima koji su nam pomogli u analizi. Analizom tih podataka mogao sam identificirati trendove, prosjeke, najviše bodove i druge statističke informacije koje su mi omogućile razumijevanje uspješnosti korisnika u igri. Također smo analizirali raspodjelu bodova, povezanost bodova i razina te vremenske promjene performansi kako bismo stekli dublji uvid u korisničko iskustvo.

U konačnici, Color Sorting Game predstavlja uspješan projekt koji kombinira zabavu i razvoj vizualnih sposobnosti korisnika. Rad na ovom projektu pružio mi je uvid u proces razvoja igre, primjenu metodologije testiranja i analizu rezultata. Nadamo se da će daljnje istraživanje i implementacija ovih spoznaja doprinijeti daljnjem napretku u području razvoja igara i pružanju korisničkog iskustva

6. Literatura:

[1] dr.sc.Mrvac Nikola, dipl.ing.Vusić Damir, dr.sc.Milković Marin, *Vizualna psihofizika i dizajn*. Varaždin: Veleučilište u Varaždinu, 2009.

[2] dr.sc. Andrija Bernik, *Gemifikacija visokoškolskog obrazovanja*. Varaždin, Hrvatska: Sveučilište Sjever, 2019.

[3] David Flanagan, *JavaScript: The Definitive Guide*.: O'Reilly Media, 2020.

[4] Palef Thomas, Davey Richard, *Phaser.js Game Development Essentials*.: Packt Publishing, 2015.

[5] Thomson Laura Welling Luke, *PHP and MySQL Web Development*.: Addison-Wesley Professional, 2016.

[6] Sriparasa Sai Srinivas, *JavaScript and JSON Essentials*.: Packt Publishing, 2015.

[7] Alan Beaulieu, *Learning SQL: Generate, Manipulate, and Retrieve Data*.: O'Reilly Media, 2020.

[8] Marc Delisle, *Mastering phpMyAdmin 4.9 for Effective MySQL Management*.: Packt Publishing, 2020.

[9] Faber Birren, *Color Psychology and Color Therapy: A Factual Study of the Influence of Color on Human Life*.: Citadel Press, 1984.

[10] Daniel Malacara, *Color Vision and Colorimetry: Theory and Applications*.: SPIE--The International Society for Optical Engineering, 2011.

[11] (2023, June) Wikipedia - Ishihara test. [Online]. https://en.wikipedia.org/wiki/Ishihara_test

[12] Dinesh Shrey, Pradeep Venkatesh, Twinkle Parmar, Sourabh Sharma Supriyo Ghose. (2014, June) A simple modification of the Farnsworth-Munsell 100-Hue test for much faster assessment of color vision. [Online]. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4131328/

7. Popis slika:

Slika 1 - Upotreba javascripta (poglavlje 2.1.1.) 5
Slika 2 - PhaserJS osnovni podaci (poglavlje 2.1.2.) 6
Slika 3 - Prednosti korištenja PHP-a (poglavlje 2.1.3.) 8
Slika 4 - Primjer upotreba JSON-a (poglavlje 2.1.4.) 10
Slika 5 - Vrste SQL instrukcija (poglavlje 2.1.5.) 11
Slika 6 - Logo MySQL baze podataka (poglavlje 2.1.6.) 13
Slika 7 - phpMyAdmin web sučelje (poglavlje 2.1.6.) 13
Slika 8 - CapacitorJS prikaz (poglavlje 2.1.7.) 16
Slika 9 - CapacitorJS logo (poglavlje 2.1.7.) 16
Slika 10 - Fotoreceptori u ljudskom oku (poglavlje 2.2.1.) 17
Slika 11 - Shematski prikaz izgleda i međupovezanosti fotoreceptora mrežnice 18
Slika 12 - Primjer testiranja praga percepcije metodom limita 18
Slika 13 - Boje i emocije u marketingu (poglavlje 2.2.2.) 20
Slika 14 - Ishihara test (poglavlje 2.2.4.) 22
Slika 15 - Farnsworth Munsell 100 Hue test (poglavlje 2.2.5.) 23
Slika 16 - Struktura gemifikacijskog pristupa 25
Slika 17 - Glavni ekran aplikacije (poglavlje 3.1.3.) 27
Slika 18 - Izbornik aplikacije (poglavlje 3.1.3.) 28

Sveučilište
Sjever

UNIVERSITY NORTH

VŽ KC

IMM

SVEUČILIŠTE SJEVER

IZJAVA O AUTORSTVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, DAMIR SABOL *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom IZRADA APLIKACIJE ZA VIZUALNO TESTIRANJE I UNAPRIJEĐENJE PERCEP VJEŠTINA RAZPOZN. BOJA *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
DAMIR SABOL
Damir Sabol
(vlastoručni potpis)

Sukladno čl. 83. Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Sukladno čl. 111. Zakona o autorskom pravu i srodnim pravima student se ne može protiviti da se njegov završni rad stvoren na bilo kojem studiju na visokom učilištu učini dostupnim javnosti na odgovarajućoj javnoj mrežnoj bazi sveučilišne knjižnice, knjižnice sastavnice sveučilišta, knjižnice veleučilišta ili visoke škole i/ili na javnoj mrežnoj bazi završnih radova Nacionalne i sveučilišne knjižnice, sukladno zakonu kojim se uređuje znanstvena i umjetnička djelatnost i visoko obrazovanje.