

**PROIECT**

**Proiectare Interfeţe Utilizator**

**Cadru didactic:**

**Titulatura.Şef lucr.dr.ing.Luminiţa Popa**

**Student:**

**Chelaru Denisa-Camelia**

**BRAȘOV, 2024**

**Departamentul Automatică și Tehnologia Informației**

**Programul de studii: Tehnologia Informației**

***CHELARU Denisa-Camelia***

Joc de memorie

„Match the cards”

- React

Brașov, 2024

**Cuprins**

1. [Tema proiectului............................................................................................................................4](#_Tema_proiectului)
2. [Descrierea jocului...........................................................................................................................4](#_Descrierea_jocului)
3. [Tehnologii și instrumente utilizate.................................................................................................4](#_Tehnologii_și_instrumente)
4. [Arhitectura și structura proiectului................................................................................................5](#_Arhitectura_și_structura)
5. [Componenta SingleCard.................................................................................................................6](#_Componenta_singleCard)
6. [Fișierul App.js..................................................................................................................................8](#_Fișierul_App.js)
7. [Fișierul App.css...............................................................................................................................10](#_Fișierul_App.css)
8. [Fișierul Alert.js................................................................................................................................11](#_Fișierul_alert.js)
9. [Capturi de ecran.............................................................................................................................12](#_Capturi_de_ecran)
10. [Concluzii..........................................................................................................................................14](#_Concluzii)
    1. [Direcții de dezvoltare..........................................................................................................14](#_Direcții_de_dezvoltare)
    2. [Impresii personale..............................................................................................................14](#_Impresii_personale)
11. [Bibliografie…………………………………………………………………………………………………………………………………14](#_Bibliografie)

# Tema proiectului

Acest joc de memorie este unul interactiv care testează și dezvoltă abilitățile de concentrare și memoria vizuală a utilizatorilor.

Scopul jocului este de a descoperi toate perechile de cărți identice în cel mai mic număr posibil de mutări și de a finaliza jocul.

Jocul oferă o experiență distractivă și educațională, facilitând îmbunătățirea capacităților cognitive ale utilizatorului, cum ar fi memoria, concentrarea și rezolvarea problemelor.

# Descrierea jocului

Jocul este unul captivant și interactiv, creat pentru a testa și dezvolta abilitățile de memorie vizuală, concentrare și strategie a jucătorilor. Prezentând o interfață simplă și intuitivă, jocul implică întoarcerea cărților pentru a găsi și potrivi perechi identice, stimulând astfel procesele cognitive ale utilizatorului.

Jucătorul începe partida cu un set de cărți răsturnate cu fața în jos, fiecare carte ascunzând o imagine specifică. În fiecare tur, utilizatorul poate selecta două cărți pentru a le întoarce și a dezvălui simbolurile ascunse. Fiecare selectare a utilizatorului va fi contorizată și afișată pe ecran. Dacă cele două cărți întoarse sunt identice, acestea rămân în continuare cu fața în sus și sunt considerate potrivite, contribuind la progresul jocului. În caz contrar, cărțile vor fi întoarse înapoi cu fața în jos.

Interfața grafică simplă și atrăgătoare, adaptată pentru a fi ușor de utilizat de către

utilizatorii de toate vârstele.

Opțiunea de resetare pentru a permite utilizatorilor să înceapă un joc nou în orice moment, eliminând progresul anterior și reinițializând starea jocului.

# Tehnologii și instrumente utilizate

Am ales să dezvolt și să gestionez proiectul în **Visual Studio Code (VSC),** un mediu de dezvoltare integrat (IDE) extrem de versatil și popular în comunitatea de dezvoltare software.

În dezvoltarea aplicației, am folosit **React** pentru construirea interfeței. Structura de bază a proiectului este compusă din componente și fișiere de stil, conform practicilor recomandate pentru dezvoltarea în React.

React este o bibliotecă JavaScript modernă, cunoscută pentru eficiența și flexibilitatea sa în construirea interfețelor de utilizator. Principalele sale caracteristici și beneficii includ:

* Componente Reutilizabile: React se bazează pe conceptul de componente, permițând dezvoltatorilor să împartă interfața în bucăți independente și reutilizabile. Aceste componente pot fi compuse și utilizate în diverse contexte, facilitând dezvoltarea modulară.
* Gestiunea Stării cu Hook-uri (precum useState și useEffect) oferă un mod elegant de a gestiona starea și efectele laterale în componentele funcționale. Acest lucru permite dezvoltatorilor să creeze componente inteligente și dinamice.

**JavaScript (JS)** este un limbaj de programare fundamental în dezvoltarea aplicațiilor web și este o componentă esențială în crearea jocurilor și a interfețelor interactive. Este un limbaj versatil, cu o sintaxă ușor de înțeles, care permite dezvoltatorilor să creeze aplicații web complexe și interfețe de utilizator interactive.

Pe de altă parte, am folosit **CSS** pentru a adăuga stil și aspect vizual paginilor. Am definit reguli de stil în fișierele CSS pentru a personaliza culorile, fonturile, dimensiunile și alte proprietăți vizuale ale elementelor.

Aplicația rulează la adresa URL: http://localhost:3000/.

# Arhitectura și structura proiectului

O imagine care conține text, captură de ecran, Font, software

Descriere generată automat

O imagine care conține text, captură de ecran, Font

Descriere generată automat

Figura 1 Figura 2

După cum se poate observa în figurile de mai sus, proiectul este structurat în următoarele :

* O componentă SingleCard – reprezintă fiecare carte individuală din joc și gestionează aspectul vizual al acesteia și interacțiunile dintre ele;
* Fișierul alert.js – conține codul necesar afișării mesajului de la finalul jocului;
* Fișierele comune aplicațiilor React: App.css, App.js, index.css, index.js – definesc structura și logica de bază a aplicației sau conțin stilurile și formatarea vizuală utilizate în aplicație;
* Fișierul cu imaginile cărților - aceste imagini sunt încărcate și utilizate în componenta SingleCard pentru a oferi elementele vizuale și identitatea fiecărei cărți.

# Componenta singleCard

Componenta SingleCard este esențială în această aplicație deoarece reprezintă fiecare element vizual și interactiv al unei cărți. Printre funcționalitățile aplicației se numără:

* Reprezentarea vizuală a cărților: Această componentă se ocupă de afișarea cărților în joc, fiecare având o imagine specifică pentru fața cărții și o imagine pentru spatele cărții. Este responsabilă pentru prezentarea și stilizarea aspectului vizual al fiecărei cărți.
* Integrarea funcționalității de joc: SingleCard lucrează împreună cu alte componente și funcționalități ale aplicației pentru a gestiona logica jocului. Ea reprezintă o parte esențială a interfeței vizuale a jocului și este interconectată cu alte componente pentru a asigura funcționalitatea corespunzătoare a jocului.

import './SingleCard.css'

export default function SingleCard({ card, handleChoice, flipped, disabled }) {

  const handleClick = () => {

    if(!disabled) {

    handleChoice(card);

    }

  };

  return (

    <div className="card">

      <div className={`card-inner ${flipped ? "flipped" : ""}`}>

        <img className="front" src={card.src} alt="card front" />

        <img

          className="back"

          src="/img/cover.jpg"

          onClick={handleClick}

          alt="card back"

        />

      </div>

    </div>

  );

}

Codul 1 – SingleCard.js

1. Funcția SingleCard: primește patru proprietăți: card (reprezintă datele specifice unei cărți), handleChoice (funcție responsabilă pentru gestionarea alegerii cărților), flipped (indică dacă o carte este răsturnată sau nu), disabled (este un indicator dacă componenta este activă sau inactivă în funcție de etapa jocului).
2. Funcția handleClick: este apelată atunci când utilizatorul dă clic pe cartea din joc. Aceasta verifică dacă această componentă este activă (!disabled) și, în caz afirmativ, apelează funcția handleChoice, pasând informații despre cartea curentă (card).
3. Structura JSX a componentei: returnează un element <div> care reprezintă cartea. În cadrul acestui div, se află un alt div cu clasa card-inner, care conține două imagini: una pentru fața cărții (front) și una pentru spatele cărții (back).

Codul de mai jos reprezintă un set de reguli CSS destinat stilizării cărților în cadrul aplicației React. Codul CSS are rolul de a oferi aspectul și comportamentul vizual al cărților când acestea sunt afișate și când sunt întoarse în timpul jocului.

.card{

    position: relative;

}

.card img{

    width: 50%;

    display: block;

    border: 5px solid #4b9e03;

    border-radius: 6px;

}

.card .front{

    transform: rotateY(90deg);

    position: absolute;

}

.flipped .front{

    transform: rotateY(0deg);

    transition-delay: 0.2s;

}

.card .back {

    transition: all ease-in 0.2s;

    transition-delay: 0.2;

}

.flipped .back{

    transform: rotateY(90deg);

    transition-delay: 0s;

}

Codul 2 – SingleCard.css

Se definește un container pentru cărți (div.card) cu un aspect general și proprietăți de bază pentru imagini (img) care reprezintă fața și spatele cărților.

Se folosesc transformări CSS pentru a simula efectul de întoarcere a cărților. În momentul în care o carte este întoarsă (flipped), aceasta este rotită la 90 de grade pentru a afișa partea opusă a cărții.

Se utilizează tranzacții CSS pentru a oferi o trecere lină între cele două stări ale cărților (întoarsă și neîntoarsă). Acest lucru creează o animație plăcută pentru utilizator atunci când cărțile sunt răsucite.

# Fișierul App.js

Fișierul prezentat reprezintă componenta principală a unei aplicații dezvoltate în React pentru jocul de memorie. Acesta gestionează logica esențială a jocului, inclusiv afișarea interfeței grafice și manipularea stării aplicației în timpul rulării.

Folosind funcționalități precum Hook-urile useState și useEffect din React, acest fișier controlează și actualizează starea jocului pe măsură ce utilizatorul interacționează cu interfața. De

exemplu, monitorizează cărțile alese de utilizator și verifică dacă acestea se potrivesc, gestionând afișarea lor sau întoarcerea în funcție de rezultat.

De asemenea, conține funcții esențiale precum shuffleCards, responsabilă de amestecarea și pregătirea cărților pentru o nouă rundă, și renderMessage, care afișează un mesaj de felicitare atunci când jocul este finalizat cu succes.

Prin structura și logica sa, acest fișier contribuie esențial la funcționalitatea jocului, oferind o experiență interactivă și captivantă utilizatorilor în cadrul aplicației.

import logo from './logo.svg';

import './App.css';

import React, { useState, useEffect } from 'react';

import SingleCard from './components/SingleCard';

import Alert from './alert';

// Definirea imaginilor pentru cărțile din joc

const cardImages = [

  { src: "/img/img1.jpg", matched: false },

  { src: "/img/img2.jpg", matched: false },

  { src: "/img/img3.jpg", matched: false },

  { src: "/img/img4.jpg", matched: false },

  { src: "/img/img5.jpg", matched: false },

  { src: "/img/img6.jpg", matched: false },

  { src: "/img/img7.jpg", matched: false },

  { src: "/img/img8.jpg", matched: false } ];

function App() {

// Starea jocului și a interacțiunii utilizatorului

  const [cards, setCards] = useState([]);

  const [turns, setTurns] = useState(0);

  const [choiceOne, setChoiceOne] = useState(null);

  const [choiceTwo, setChoiceTwo] = useState(null);

  const [disabled, setDisabled] = useState(false);

  const [matchedPairs, setMatchedPairs] = useState(0);

// Funcție pentru amestecarea cărților la începutul jocului

  const shuffleCards = () => {

    const shuffledCards = [...cardImages, ...cardImages]

      .sort(() => Math.random() - 0.5)

      .map((card) => ({ ...card, id: Math.random() }));

    setChoiceOne(null);

    setChoiceTwo(null);

    setCards(shuffledCards);

    setTurns(0);

    setMatchedPairs(0);

    setDisabled(false);  };

// Funcția care gestionează alegerile utilizatorului

  const handleChoice = (card) => {

    if (!disabled && (choiceOne === null || choiceTwo === null)) {

      choiceOne ? setChoiceTwo(card) : setChoiceOne(card);

    } };

// Efect secundar pentru verificarea potrivirii între cărți alese

  useEffect(() => {

    if (choiceOne && choiceTwo) {

      setDisabled(true);

      if (choiceOne.src === choiceTwo.src) {

// Actualizarea stării cărților și a perechilor potrivite

        setCards(prevCards => {

          const updatedCards = prevCards.map(card => {

            if (card.src === choiceOne.src) {

              return { ...card, matched: true };

            } else {

              return card;

            }  });

          setMatchedPairs(prevMatchedPairs => prevMatchedPairs + 1);

          return updatedCards;  });

        resetTurn();

      } else {

// Resetarea cărților în caz de nepotrivire

        setTimeout(() => resetTurn(), 1000);    } }

  }, [choiceOne, choiceTwo]);

  const resetTurn = () => {

    setChoiceOne(null);

    setChoiceTwo(null);

    setTurns(prevTurns => prevTurns + 1);

    setDisabled(false); };

// Funcție pentru afișarea mesajului de finalizare a jocului

  const renderMessage = () => {

    if (cards.every((card) => card.matched || card.flipped)) {

      return (

        <Alert

          message="Felicitări! Ai terminat jocul!"

          variant="success"

        /> );  }

    return null; };

// Efect secundar pentru amestecarea cărților la începutul jocului

  useEffect(() => {

    shuffleCards();

  }, []);

  return (

    <div className="App">

      <h1>Match the cards</h1>

      <button onClick={shuffleCards}>New Game</button>

    { }

      {renderMessage()}

      <div className="card-grid">

        {cards.map((card) => (

          <SingleCard

            key={card.id}

            card={card}

            handleChoice={handleChoice}

            flipped={card === choiceOne || card === choiceTwo || card.matched}

            disabled={disabled}

          />    ))}

      </div>

      <p>Turns: {turns}</p>

    </div>

  );

}

export default App;

Codul 3 – App.js

O imagine care conține text, captură de ecran, Font, număr

Descriere generată automat

Figura 3 – fișierul cu imagini utilizate în aplicație

# Fișierul App.css

Conține stilurile CSS utilizate pentru a formata și a da un aspect plăcut și coeziv aplicației. Este responsabil pentru aspectul vizual al diferitelor elemente din cadrul jocului de memorie, cum ar fi titlul, butoanele, containerul pentru cărți și dimensiunile lor.

.App {

  max-width: 860px;

  margin: 40px auto

}

h1, p{

  color: #144b01;

  font-family: Georgia, 'Times New Roman', Times, serif;

}

button{

  background: none;

  border: 2px solid #144b01;

  padding: 6px 12px;

  border-radius: 4px;

  color: #144b01;

  font-weight: bold;

  cursor: pointer;

  font-size: 1em;

  font-family: Georgia, 'Times New Roman', Times, serif;

}

button:hover{

  background: #307602;

  color: #fff;

}

.card-grid{

  margin-top: 40px;

  display: grid;

  grid-template-columns: 1fr 1fr 1fr 1fr;

  grid-gap: 20px;

}

Codul 4 – App.css

App este clasa principală pentru containerul principal al aplicației, care îi definește lățimea maximă și marginea pentru a fi plasată în centrul paginii.

Stilurile pentru h1 și p se aplică titlului și paragrafelor, stabilind culoarea textului și fonturile utilizate.

Butonul este stilizat cu proprietăți precum culoare de fundal, grosimea și tipul border-ului, padding-ul, fontul și culorile la interacțiunea cu mouse-ul (hover).

Card-grid este un container pentru cărți și este configurat să aibă un anumit aspect în ceea ce privește poziționarea cărților, dimensiunile și spațiile dintre ele.

# Fișierul alert.js

Această componentă reprezintă un element vizual utilizat pentru a afișa mesajul de succes în cadrul aplicației React. Ea primește trei proprietăți (title, message și variant) care permit personalizarea mesajului și a aspectului vizual al alertei.

Componenta Alert este utilă pentru a oferi feedback utilizatorului la sfârșitul jocului.

import React from 'react';

const Alert = ({ title, message, variant }) => (

  <div className={`alert alert-${variant}`}>

    <h4>{title}</h4>

    <p>{message}</p>

  </div>

);

export default Alert;

Codul 5 – Alert.js

# Capturi de ecran

O imagine care conține text, captură de ecran, Dreptunghi, diagramă

Descriere generată automat

Figura 4 – aspectul paginii la deschiderea aplicației

O imagine care conține text, captură de ecran, diagramă

Descriere generată automat

Figura 5 – imagine din timpul jocului cu 2 cărți diferite

O imagine care conține text, captură de ecran, software, Pictogramă computer

Descriere generată automat

Figura 6 – imagine din timpul jocului cu 3 perechi de cărți găsite

O imagine care conține text, captură de ecran, Dreptunghi, diagramă

Descriere generată automatFigura 7 – aplicația după apăsarea butonului de resetare (New Game)

O imagine care conține text, captură de ecran, Sistem de operare, Pictogramă computer

Descriere generată automat

Figura 8 – aspectul paginii la sfârșitul jocului

# Concluzii

## Direcții de dezvoltare în viitor

Consider că acest proiect poate fi îmbunătățit prin introducerea de noi funcționalități, cum ar fi:

* Adăugarea de nivele de dificultate, cum ar fi numărul de cărți care să crească odată cu progresul jocului sau limite de timp pentru a finaliza un nivel.
* Introducerea de animații și efecte vizuale mai avansate pentru întoarcerea și potrivirea cărților, astfel încât jocul să devină mai interactiv și mai atractiv.

## Impresii personale

Ducerea la bun sfârșit a acestui proiect mi s-a părut interesantă. Pe parcursul realizării acestui proiect mi-am dezvoltat spiritul autodidact, dar în același timp am putut pune în practică toate cunoștințele despre modul de lucru cu React*.*

# Bibliografie

[1] Luminița Popa - Curs și Îndrumar de laborator

[2] “Quick Start.” *React.dev*, react.dev/learn.

[3] <https://www.youtube.com/watch?v=XcHNBMG1GQI&t=135s>