OLIMPIADA DE INOVARE ȘI CREAȚIE DIGITALĂ - InfoEducație

~Secțiunea Utilitar~

~Etapa Națională~

Joc de table

Candidați:

Colev Thomas-Daniel

Cudla Ioan-Radu

Profesor coordonator

Lăzărescu Antoneta

Coajă Gabriela

Strugariu Ciprian

Cuprins

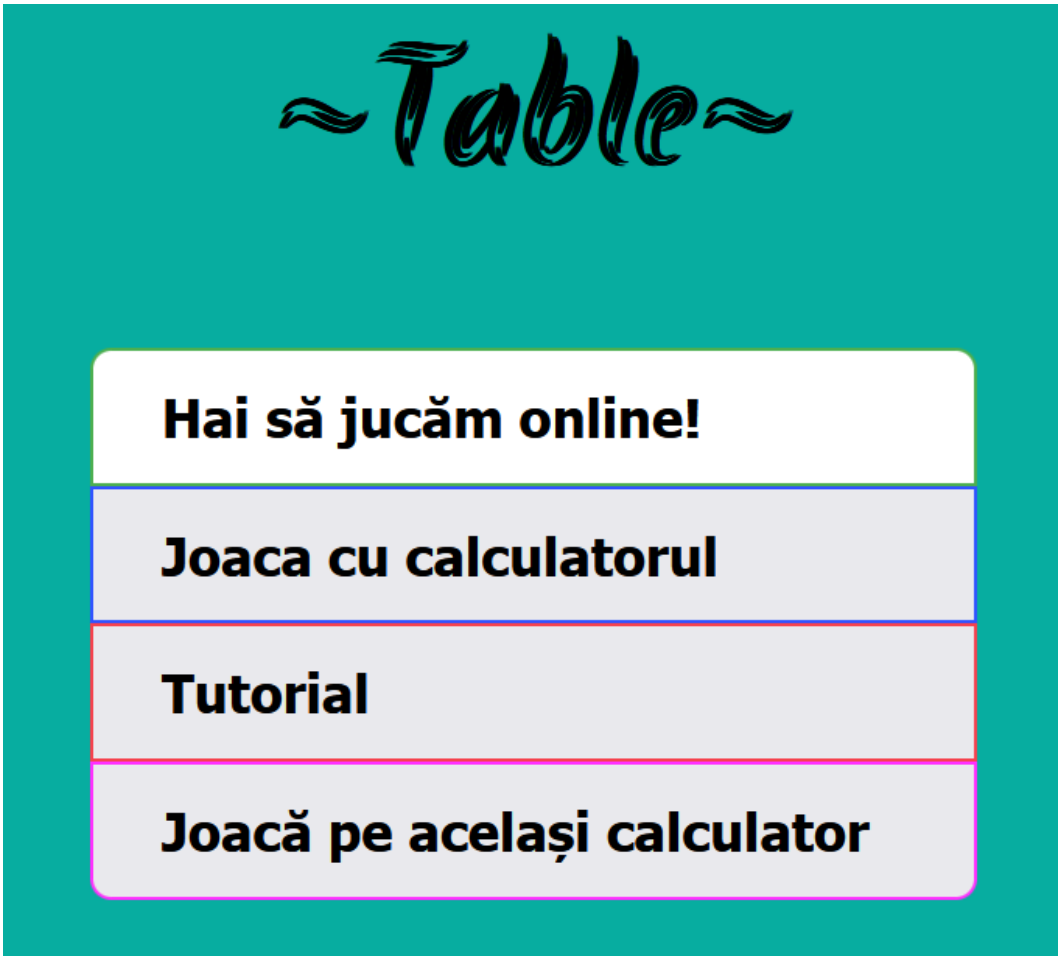
1.Scopul lucrării

Din dorința de a exersa și a îmbunătăți abilitățile de a scrie cod, dar și a implementa un proiect de o complexitate mai mare , candidații au încercat să îmbine utilul cu plăcutul.
Astfel, acest joc arhicunoscut- *jocul de table*- a apărut în calea noastră, echipa realizând că nu există implementări calitative ale acestuia pe internet.
Prin *implementare calitativă¸* echipa înțelege o platformă *„on-site”* (bucuria jocului nu este condiționată de instalarea unei aplicații terțe, experiența putând avea loc în browser) cu design minimalist, intuitiv, demn de secolul 21.
Astfel, proiectul s-a rezumat la două mari probleme ce au necesitat rezolvare: implementarea *mecanicii de joc,* dar și la generarea unui sistem ce să permită atingerea scopului jocului de table, *acela de a crea legături sociale*.
Vom încerca să prezentăm sumar rezultatul, în paginile ce urmează.

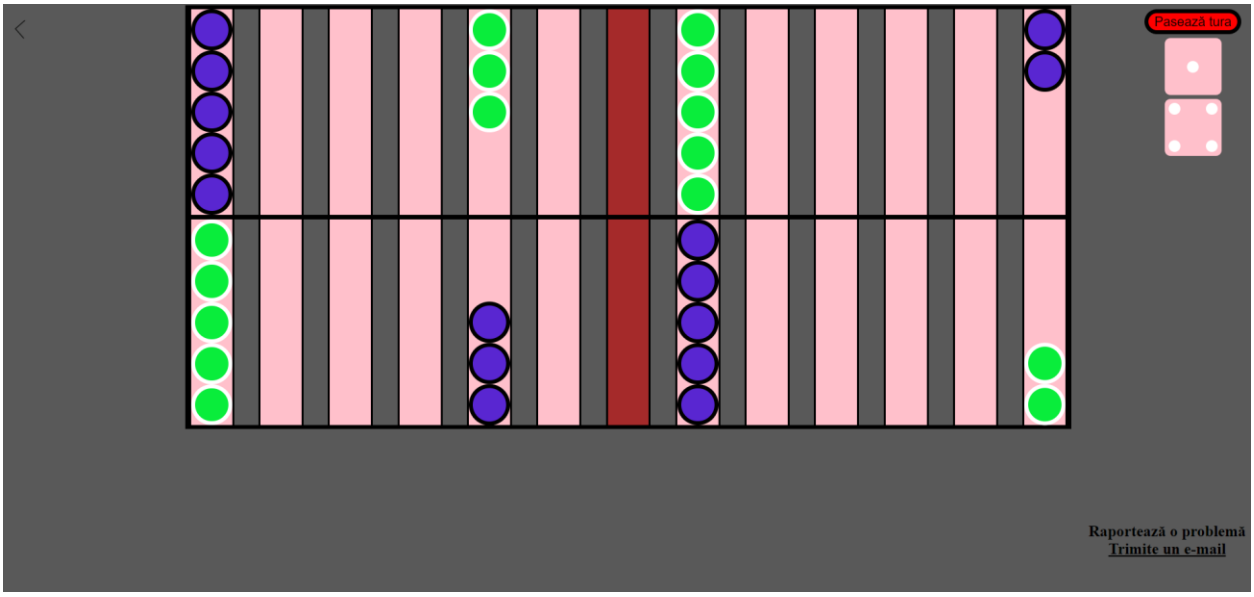
2.Structura site-ului

Utilizatorul este întâmpinat de pagina principală, cea care ne prezintă un buton de afișare a meniului, meniu din care utilizatorul poate alege modul de joc preferat în cadrul acelei sesiuni.

Necondiționat de opțiunea utilizatorului, acesta va fi redirecționat către pagina jocului propriu zis.



Ideea unui design cât mai simplu și minimalist este regăsită și în cadrul acestei pagini. Aceasta având doar elementele strict necesare.



Din încercarea de a evita „*kitsch-ul*”, s-au evitat animațiile și elementele decorative 3D.
În colțul din stânga sus se află butonul de părăsire a sesiunii curente și întoarcere la pagina principală, fiind și singurul animat.
Tabla de joc este centrală, înconjurată de butonul de zaruri, zaruri , dar și unde este prezentată scris desfășurarea jocului.

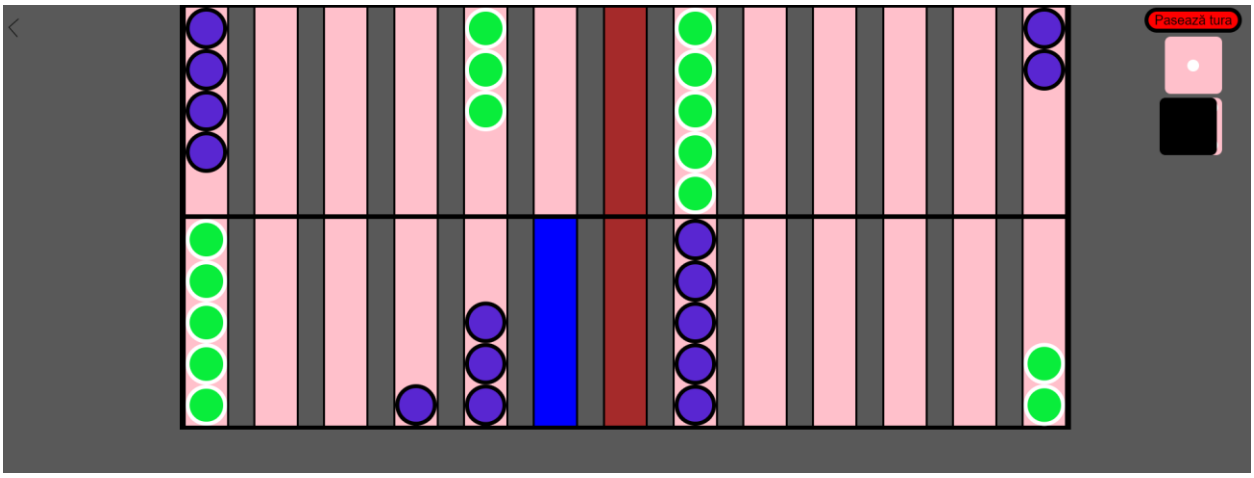
3.Modul de folosire

Obiectivul jocului de table

Poate nu putem considera obiectivul jocului de table o regulă, dar cu siguranță este foarte important de știut că obiectivul jocului este să aduceți toate piesele în “casă”, dupa care să le scoateți afară**.** Primul care scoate toate piesele afară este declarat câștigător.

Mutarea pieselor

Fiecare jucator isi va misca piesele conform celor 2 zaruri aruncate de catre el. El va muta piesele in corcondanta cu numerele indicate de zaruri. Sa presupunem ca zarurile indica 4-2, el va putea muta piesa 6 spatii sau o piesa 4 spatii si urmatoarea 2 spatii. Trebuie sa luati in considerare ca mutarea unei singure piese presupune defapt 2 mutari, fiecare mutare trebuie sa corespunda cu numarul indicat pe zar. Mutarea este sprijinită în mod automat, prin albăstrirea spațiilor unor mutări ce poate avea loc, astfel economisind timp prin numărărea de spații, timp câștigat față de un joc de table clasic, fizic.



Dublele

Dacă pe ambele zaruri este indicat același număr, de exemplu 4-4 sau 5-5,

se va numi *dublă*, iar cel care a aruncat zarurile nimerind dubla va putea face 4 mutări în loc de două. Deci daca va nimeri o dublă de 3, 3-3 va putea muta de 4 ori 3 spații, indiferent dacă mută o piesă sau 4.
Jucătorii aruncă și joacă alternativ în timpul jocului, excepție făcând cazul în care unul dintre jucători nu poate să facă o mișcare legală, însemând că va trebui să renunțe la mutarea sa în favoarea oponentului său.

Marcarea spațiilor

Un jucător poate marca un spațiu și să-l dețină dacă are poziționat în acel spațiu 2 sau mai multe piese, odată ce deține acel spațiu oponentul său nu mai are voie să se oprească în acel spațiu indiferent că face o mutare simplă folosindu-se doar de unul dintre numerele arătate de zaruri, sau se folosește de combinația acestora și indiferent de număr, prima mutare pică pe un spațiu deținut de oponent.

Blocajul

Un blocaj este atunci când un jucător deține 6 spații consecutive, astfel oponentul său este prins dincolo de blocaj și nu poate trece de acesta deoarece cel mai mare număr pe zar este 6 și el deține 6 spații.

Descoperire

O singură piesă pe un spatiu este considerată descoperită, astfel dacă în procesul de mutare al pieselor treceți sau mutarea se termină pe spațiul unde are oponentul piesa descoperită, aceasta va fi înlăturată de pe tablă. Cel căruia i se înlătură piesa va trebui să o readucă înapoi pe tablă. Acesta nu va putea face nicio mișcare până când nu își introduce piesa pe tablă. Piesa poate fi introdusă înapoi pe tablă în casa oponentului pe spațiul corespunzător numărului de pe zar, dacă oponentul deține însă acel spațiu, piesa nu va putea fi reintrodusă în joc.

Închiderea tablei

Un jucător care a reușit să ocupe toate cele 6 spații din propria casa cu câte cel puțin 2 piese pe fiecare spațiu, va putea zice că a închis tabla. Dacă oponentul său are încă piese în afara tablei nu le va putea reintroduce decât după ce va fi eliberat unul dintre spații(va fi cel mult o piesă pe spațiu). Cel care a închis tabla va continua să arunce cu zarurile și să joace practic de unul singur până se va elibera unul dintre spații, astfel openentul său va avea șansa să nimerească numărul spațiului cu unul dintre zaruri. De reținut că oponentul încă mai are dreptul să arunce cu zarurile chiar dacă tabla este închisă doar că va trebui să aștepte eliberarea unui spatiu.

Mutările obligatorii

Mulți începători nu cunosc acesata regulă deși este una dintre cele mai importante reguli ale jocului de table. Un jucător este forțat să facă toate mutările conform zarurilor dacă acest lucru este posibil. Dacă poate mută doar pentru unul din numerele indicate de zaruri, va trebui să mute pentru cel mai mare număr, dacă acest lucru nu este posibil va muta pentru numărul mai mic indicat de zaruri.

Scoaterea pieselor

Odată ce un jucător a adus absolut toate piesele sale în propria casă, va putea să înceapă scoaterea lor de pe tablă. Piesele ce sunt astfel retrase de pe tablă nu vor fi reintroduse înapoi. Jucătorul care reușește să scoată cel mai repede toate piesele de pe tablă conform regulilor, este declarat *câștigător*.

4.Modul de realizare

Daca am clasifica structura proiectului în funcție de limbajele de programare folosite, aceasta ar fi: partea de HTML, partea de CSS și partea de JavaScript.

i. HTML

Partea de HTML reprezintă scheletul jocului. Aceasta conține toate zarurile posibile în urmă rulării și tablă propriu-zisă.Tablă a fost proiectată sub formă a două flexbox-uri. În HTML, *un flexbox* reprezintă un tabel cu o singură dimensiune: fie linii, fie coloane.

```
<!--Flexbox container - the upper part  -->
<div class = "flex-container-up">
    <div class = "column c13"></div>
    <div class = "column c14"></div>
    <div class = "column c15"></div>
    <div class = "column c16"></div>
    <div class = "column c17"></div>
    <div class = "column c18"></div>
    <div class = "column c69"></div> <!--mijlocul-->
    <div class = "column c19"></div>
    <div class = "column c20"></div>
    <div class = "column c21"></div>
    <div class = "column c22"></div>
    <div class = "column c23"></div>
    <div class = "column c24"></div>
</div>
<!-- End Flexbox container - the upper part-->
<!-- Flexbox container - the bottom part-->
<div class = "flex-container-down">
    <div class = "column c12"></div>
    <div class = "column c11"></div>
    <div class = "column c10"></div>
    <div class = "column c9"></div>
    <div class = "column c8"></div>
    <div class = "column c7"></div>
    <div class = "column c-69"></div> <!--mijlocul-->
    <div class = "column c6"></div>
    <div class = "column c5"></div>
    <div class = "column c4"></div>
    <div class = "column c3"></div>
    <div class = "column c2"></div>
    <div class = "column c1"></div>
</div>
<!-- End Flexbox container - the bottom part-->
```

Imagine cu toate coloanele tablei implementate.

ii. CSS

Partea de CSS cuprinde, evident, toate stilizările proiectului: alinieri, contururi, culori deosebite, butoane și piesele jocului de table. Acestea din urmă au fost create folosind Javascript și CSS. Practic, sunt doar niște div-uri în HTML prelucrate cu ajutorul CSS pentru a arăta ca o piesă de joc.

```css
div.flex-container-up{
display : flex;
flex-direction : row;
outline : 4px solid black;
position : relative;
justify-content : center;
margin : 5px auto 5px auto;
left : 50%;
width : 70vw;
margin-left : -35vw;
height:20vw;
}
div.flex-container-down{
    margin-bottom : 5px;
    display : flex;
flex-direction : row;
outline : 4px solid black;
position : relative;
justify-content : center;
margin : 5px auto 5px auto;
left : 50%;
width : 70vw;
margin-left : -35vw;
height:20vw;
}
div.column {
    position : relative;
    width : 4vw;
    height : auto;
    background-color : pink;
    margin-right : 25px;
    border-left : 2px solid black;
    border-right : 2px solid black;
    flex-shrink : 0;
}
```

Stilizarea flexbox-urilor și coloanelor acestora

```css
.piece{
    width : 4vw;
    height : 4vw;
    border-radius : 50%;
    background-color : black;
}
.white{
    background-color : white;
    border : 4px solid black;
    box-sizing : border-box;
}
.black{
    background-color : black;
    border : 4px solid  white;
    box-sizing : border-box;
}
```

Implementarea pieselor albe și negre

iii. JavaScript

Pe departe, cea mai complexă și cea mai interesantă parte a proiectului este cea de JavaScript. Acest limbaj de programare a fost folosit pentru implementarea mecanicilor jocului: toată logica unui joc tradițional de table plus interacțiunile cu utilizatorul ce permit derularea unei partide. Codul sursă este structurat pe numeroase funcții conținând comentarii sugestive pentru a nu pierde firul logic din el,dimensiunea lui fiind de aproape *1000 linii* de cod. Cele mai reprezentative funcții sunt:

- funcția *calculate()*: la momentul apăsării unei piese ce se poate muta (marcată prin cursorul de tip pointer), această colorează coloanele mutărilor valide, în funcție de zaruri și apelează funcția move();

```
function calculate(evt) // column si n sunt numerice
{
    if(die1 + die2 > 0)
    {
        let column = evt.currentTarget.column;
    if(moved == false) // daca nu a mutat si a schimbat coloana
        {if(valid(prev1))
        {

            let aux = document.querySelector(`.c${prev1}`);
            aux.style.backgroundColor = null;
            if(columnpieces(aux) == 0)
            Cursor(aux,0);
            else
            aux.addEventListener('click',calculate);
            aux.removeEventListener('click',move);
        }
        if(valid(prev2))
        {

            let aux = document.querySelector(`.c${prev2}`);
            aux.style.backgroundColor = null;
            if(columnpieces(aux) == 0)
            Cursor(aux,0);
            else
            aux.addEventListener('click',calculate);

            aux.removeEventListener('click',move);
        }
        if(valid(prevsum))
        {

            let aux = document.querySelector(`.c${prevsum}`);
            aux.style.backgroundColor = null;
            if(columnpieces(aux) == 0)
            Cursor(aux,0);
            else
            aux.addEventListener('click',calculate);
            aux.removeEventListener('click',move);
        }
        if(prevcolumn != column && valid(prevcolumn))
        {
            let aux = document.querySelector(`.c${prevcolumn}`);
            aux.removeEventListener('click',deselect);
        }
        }
    moved = false;
    let n = player;
    col1 = null;
    col2 = null;
    colsum = null;
    //in urm 3 linii adaug optiunea de deselectare a coloanei in cazul in care userul se rasgandest
    let col = document.querySelector(`.c${column}`);
    col.c = column;
    col.addEventListener('click',deselect);
```

- funcția *move()*: la momentul selectării unei coloane valide pentru mutarea piesei selectată anterior, această funcție mută piesă deja selectată și resetează coloanele ce au fost colorate.

```
function move(evt) // functia de mutare cand dai click pe highlighted column
{
    let columnn = evt.currentTarget.c;
    let die = evt.currentTarget.d;
    let column = evt.currentTarget.cstart;
    moved = true;
    movePiece(column,columnn); // mutare

    if(die == 1) // anulez zarul care l am folosit pt a muta
        {
            if(die3)
            {
                ascunde(2);//zar02.style.visibility = 'visible';
                die1-=die3;
                if(!die1)
                die3 = 0,ascunde(1);//zar01.style.visibility = 'visible';
            }
            else
            {
                die1 = 0;
                ascunde(1);//zar01.style.visibility = 'visible';
            }
        }
    else
        if(die == 2)
            {
                die2 = 0;
                ascunde(2);//zar02.style.visibility = 'visible';
            }
            else
                {
                    die1 = 0;
                    die2 = 0;
                    ascunde(1);//zar01.style.visibility = 'visible';
                    ascunde(2);//zar02.style.visibility = 'visible';
                }

                //sterg event listenerurile
    if(col1 != null) {
        if(columnpieces(col1) == 0 || Math.abs(c1) == 666)
        Cursor(col1,0);
        col1.style.backgroundColor = null;
        col1.removeEventListener('click',move);
        if(columnpieces(col1) > 1 )
        col1.addEventListener('click',calculate);
        col1.column = c1;
    }
    if(col2 != null) {
        if(columnpieces(col2) == 0 || Math.abs(c2) == 666)
        Cursor(col2,0);
        col2.style.backgroundColor = null;
        col2.removeEventListener('click',move);
        if(columnpieces(col2) > 1 )
        col2.addEventListener('click',calculate);
        col2.column = c2;
    }
```

Pentru că funcțiile prezentate au dimensiuni mari(>200 linii) am inserat doar o mostră din fiecare.

iv. Partea de server

Datorită faptului că ne-am dorit încă din fazele incipiente ca rezultatul să fie un joc funcțional, *multiplayer*, am fost nevoiți să folosim *module Node.js* pentru a face posibil acest lucru.

Node.js este un mediu de execuție JavaScript de fundal multiplatformă cu sursă deschisă, care rulează pe motorul V8 și execută cod JavaScript în afara unui navigator web.

```
const express=require('express');
const favicon = require('serve-favicon');
const app=express();
const path=require('path');
const http=require('http');
const server = http.createServer(app);
const port = 3000;
const { Server} = require("socket.io");
const io= new Server(server);
```

modulele Node.js folosite

index.js

Inițializează produsul, indicând locația jocului, pornind o sesiune:

```
app.use(express.static(path.join(__dirname, "public")));
app.use(favicon(__dirname + '/favicon.ico'));

server.listen(port, ()=>{console.log(`server is running on ${port}`)});
```

Totodată, în acest fișier are loc toată magia sincronizării. Pe de o parte, „*ascultă*”, pe rând, datele de intrare a fiecărui participant din *room*. Pe de altă parte, execută funcțiile și trimite, „*emite*” spre utilizatori („*socket*”-urile conectate) rezultat/ apeluri de funcții din logica jocului, sincronizându-le.

```
    socket.emit('player-number', playerIndex);
    socket.emit('msjS', `Player ${playerIndex} has  just connected.`)
    console.log(`Player ${playerIndex} has connected.`);
      if(playerIndex == -1 )
      console.log(`Too many players!`);

      connections[playerIndex]=false;

      socket.on('disconnect',()=>{
          console.log(`Player ${playerIndex} has disconnected.`);
        connections[playerIndex]=null;
        playerIndex--;
      });


      socket.on('roll',()=>{
  let die1, die2,die3;
    die3 = 0;
    die1 = Math.ceil(Math.random()*6) ;
    die2 = Math.ceil(Math.random()*6) ;
    die1 > 0 ? 0 : die1++;
    die2 > 0 ? 0 : die2++;
    if(die1 > die2)
    [die1,die2] = [die2,die1];
    io.emit('zaruri',die1,die2);
  });
```

5.Bibliografie

https://www.w3schools.com/

https://stackoverflow.com/

HTML & CSS - Certification Course for Beginners – de Kiran Gavali

JavaScript – The Definitive Guide – de David Flanagan

https://nodejs.org/en/docs/

https://socket.io/docs/v4/

https://www.youtube.com/c/programmingwithmosh

6.Puncte forte

Având în vedere scopul lucrării prezentat la începutul documentului,am avut grijă ca implementarea noastră sa fie cât mai originală și să avem elemente distinctive față de concurența noastră.Așadar,prezentăm în continuare punctele forte ale aplicației noastre:

-Opțiunea de a comuta cu ușurință între modurile Singleplayer și Multiplayer : Cele două moduri sunt regăsite la pagina de început,reprezentate de cele trei butoane “Hai să jucăm” ,“Joacă pe același calculator”(Multiplayer local) și “Joacă cu calculatorul”(Singleplayer împotriva calculatorului) ;

- Posibilitatea de a raporta o problemă chiar in momentul depistării acesteia : Aplicația noastră este momentan în versiunea Alpha și suntem conștienți de faptul că pot apărea mici probleme în timpul jocului.De accea,am implementat un buton în colțul din dreapta jos al ecranului care deschide în momentul apăsării o fereastră ce permite trimiterea unui e-mail către noi.

-Opțiunea de a selecta culoarea pieselor fiecărui jucător.Înainte de a începe jocul propriu-zis,fiecare jucător este obligat să își aleagă o culoare pentru piesele sale.

-Baza de date ce stochează conturile înregistrate de utilizatori.

7.Ghid de instalare

Aplicația se inițializează prin pornirea serverului node și accesarea

acestuia pe portul 3000.

8.Răspunsuri la întrebările propuse

-Justificarea folosirii tehnologiilor alese : Am folosit CSS,HTML pentru că ne-am dorit implementarea aplicației pe un browser,JavaScript

pentru mecanica jocului si interfața utilizatorului ,iar Node.js(librărie din JavaScript) pentru a putea realiza sincronizarea jocului între doi jucători.

-Opinia autorului despre ideea ce stă la baza proiectului implementat este menționată la **1.Scopul lucrării** , iar utilitatea acestuia pentru publicul țintă (jucătorii de table de orice vârstă) constă în ușurința pornirii unui joc față de metoda fizică cât și în funcțiile adiționale ce fac jocul mai interesant (alegerea culorii,jocul împotriva calculatorului) .