**VISOKA TEHNIČKA ŠKOLA**

**STRUKOVNIH STUDIJA**

**S** **U B O T I C A**

****

**PROJEKAT**

**iz predmeta Web programiranje**

**KANDIDAT MENTOR**

**Petrik Jelena**

**12117115 dr Zlatko Čović**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**SUBOTICA 2019.god.**

**Table of Contents**

[OPIS ZADATKA](#_3znysh7) **3**

[REALIZACIJA ZADATKA](#_2et92p0) **3**

[TETRIS.JS](#_tyjcwt) 3

[TETROMINOES.JS](#_uyox62w3g346) 9

[INDEX.HTML](#_bq6ygftfbyx5) 10

[UBACIVANJE KORISNIKA U BAZU PODATAKA](#_4d34og8) 12

[UNOŠENJE PODATAKA U BAZU](#_2s8eyo1) 12

[PRIKAZIVANJE PODATAKA IZ BAZE](#_17dp8vu) 14

[STRUKTURA BAZE PODATAKA](#_3rdcrjn) **15**

[OPIS FUNKCIONALNOSTI PROJEKTA](#_26in1rg) **16**

[POČETNA STRANA](#_yxlpait4xgaw) 16

[PRITISAK START BUTTON-a, IGRA U PROCESU](#_a5lgka3bvch) 17

[PROMPT ZA KORISNIKOVO IME](#_44sinio) 18

[POTVRDA PODATAKA](#_l2t8gps6qyii) 19

[POTVRDA PODATAKA - CLICK NA OK](#_z337ya) 20

[CLICK NA REMEMBER ME BUTTON, CLICK NA SHOW KOD RECENT PLAYERS](#_3j2qqm3) 20

[CLICK NA SHOW KOD TOP 5 HIGHSCORES](#_5lsyhngjtlr7) 21

[KORIŠĆENA LITERATURA](#_21l6v241ilbe) **21**

# OPIS ZADATKA

U ovom projektu je realizovan sajt na kojem korisnici mogu igrati popularnu igru Tetris, uz to mogu i ubaciti svoj rezultat u bazu podataka.

U ovom projektu korišteni su sledeći programski jezici:

* HTML5
* CSS
* JavaScript
* PHP
* MySQL

Igra se započinje klikom na START dugme, i korisnik treba da pomoću strelica upravlja tetromino oblike kako bi se redovi “očistili” i sa tim povećao svoj rezultat i nivo do kog je stigao. Ukoliko se redovi naslažu jedni na druge i udare gornju granicu kanvasa, igra se završava i korisnikov krajnji rezultat se čuva.

Korisnik dobija opciju da sačuva svoje podatke ukoliko unese svoje ime u iskačući prozor nakon gubitka i pritisne "Remember me!" dugme (koje se pojavi ukoliko je korisnik potvrdio svoje ime i rezultat). Rezultate svih igrača možemo videti pritiskom na jednu od dugmadi ispod tela igre. Prvo dugme prikazuje highscore-ove tj. 5 igrača sa najvećim ostvarenim rezultatom. Drugo dugme prikazuje poslednjih 5 korisnika koji su odlučili da unesu svoje podatke u bazu.

# REALIZACIJA ZADATKA

# 

## TETRIS.JS

Sama igra je realizovana preko HTML5 canvas elementa i preko JavaScript funkcija. U sledećem kodu se nalaze funkcije koje se koriste pri realizaciji igre (uključeni su i komentari putem kojih se može pratiti kod.)

**const *cvs***=**document**.getElementById(**"tetris"**);

**const *ctx***=***cvs***.getContext(**"2d"**);

**const *scoreElement*** = **document**.getElementById(**"score"**);

**const *levelElement*** = **document**.getElementById(**"level"**);

**const *ROW***=20;

**const *COL***=**COLUMN**=10;

**const *SQ***=**squareSize**=20;

**const *VACANT***=**"WHITE"**;

*//draw a square*

**function** *drawSquare*(x,y,color){

***ctx***.**fillStyle**=color;

***ctx***.fillRect(x\****SQ***,y\****SQ***,***SQ***,***SQ***);

***ctx***.**strokeStyle**=**"BLACK"**;

***ctx***.strokeRect(x\****SQ***,y\****SQ***,***SQ***,***SQ***);

}

*//create board*

**var *board***=[];

**for**(**var *r***=0;***r***<***ROW***;***r***++){

***board***[***r***]=[];

**for**(**var *c***=0;***c***<***COL***;***c***++){

***board***[***r***][***c***]=***VACANT***;

}

}

*//draw the board*

**function** *drawBoard*(){

**for**(**var** r=0;r<***ROW***;r++){

**for**(**var** c=0;c<***COL***;c++){

*drawSquare*(c,r,***board***[r][c]);

}

}

}

*drawBoard*();

*// the pieces and their colors*

**const *PIECES***= [

[***Z***,**"red"**],

[***S***,**"green"**],

[***T***,**"purple"**],

[***O***,**"yellow"**],

[***L***,**"orange"**],

[***I***,**"cyan"**],

[***J***,**"blue"**]

];

*// generate random pieces*

**function** *randomPiece*(){

**let** r = **randomN** = **Math**.floor(**Math**.random() \* ***PIECES***.**length**) *// 0 -> 6*

**return new** *Piece*( ***PIECES***[r][0],***PIECES***[r][1]);

}

**p** = *randomPiece*();

*// The Object Piece*

**function** *Piece*(tetromino,color){

**this**.**tetromino**=tetromino;

**this**.**color**=color; *//the color of every piece*

**this**.**tetrominoN**=0; *//the first pattern, it goes from 0 to 3*

**this**.**activeTetromino**=**this**.**tetromino**[**this**.**tetrominoN**]; *//the tetronimo we are playing with*

*//to control the pieces, with this we see where the piece is on the board*

**this**.**x**=3;

**this**.**y**=-2;

}

*//fill function*

*Piece*.**prototype**.fill = **function**(color){

**for**(**var** r=0;r<**this**.**activeTetromino**.**length**;r++){

**for**(**var** c=0;c<**this**.**activeTetromino**.**length**;c++){

**if**(**this**.**activeTetromino**[r][c]){ *//we only need to draw the occupied squares*

*drawSquare*(**this**.**x** + c,**this**.**y** + r,color);

}

}

}

};

*// draw a piece to the board*

*Piece*.**prototype**.draw = **function**(){

**this**.fill(**this**.**color**);

};

*//undraw the piece (because the piece is keeping its shape in every block as it falls)*

*Piece*.**prototype**.unDraw = **function**(){

**this**.fill(***VACANT***);

};

*//move Down the piece*

*Piece*.**prototype**.moveDown = **function**(){

**if**(!**this**.collision(0,1,**this**.**activeTetromino**)){

**this**.unDraw();

**this**.**y**++;

**this**.draw();

}**else**{

**this**.lock(); *// we lock the piece and generate a new one*

**p** = *randomPiece*();

}

};

*//move the piece right*

*Piece*.**prototype**.moveRight = **function**(){

**if**(!**this**.collision(1,0,**this**.**activeTetromino**)) {

**this**.unDraw();

**this**.**x**++;

**this**.draw()

}

};

*//move the piece left*

*Piece*.**prototype**.moveLeft = **function**(){

**if**(!**this**.collision(-1,0,**this**.**activeTetromino**)) {

**this**.unDraw();

**this**.**x**--;

**this**.draw();

}

};

*//rotate the piece*

*Piece*.**prototype**.rotate = **function**(){

**var** nextPattern = **this**.**tetromino**[(**this**.**tetrominoN** + 1)%**this**.**tetromino**.**length**];

**var** kick = 0;

**if**(**this**.collision(0,0,nextPattern)){

**if**(**this**.**x** > ***COL***/2){

*// it's the right wall*

kick = -1; *// we need to move the piece to the left*

}**else**{

*// it's the left wall*

kick = 1; *// we need to move the piece to the right*

}

}

**if**(!**this**.collision(kick,0,nextPattern)){

**this**.unDraw();

**this**.**x** += kick;

**this**.**tetrominoN** = (**this**.**tetrominoN** + 1)%**this**.**tetromino**.**length**; *// (0+1)%4 => 1*

**this**.**activeTetromino** = **this**.**tetromino**[**this**.**tetrominoN**];

**this**.draw();

}

};

**var *score*** = 0;

**var *level*** = 1;

*Piece*.**prototype**.lock = **function**(){

**for**( ***r*** = 0; ***r*** < **this**.**activeTetromino**.**length**; ***r***++){

**for**(***c*** = 0; ***c*** < **this**.**activeTetromino**.**length**; ***c***++){

*// we skip the vacant squares*

**if**( !**this**.**activeTetromino**[***r***][***c***]){

**continue**;

}

*// pieces to lock on top = game over*

**if**(**this**.**y** + ***r*** < 0){

alert(**"Game Over!"**);

*//stop request animation frame*

**window**.**name** = prompt (**'Please enter your username: '**);

**if**(**name**==**""**){

**name**=prompt (**'This field cant be empty. Please enter your username: '**);

}

**else**{

**window**.**confirmName**=confirm(**"This is your username: "**+**name**+

**"\n This is your score: "** +***score***);

}

**window**.**getScore**=***score***;

**document**.**cookie**=**getScore**;

**if**(**confirmName**==**true**)

{

**document**.getElementById(**"hiddenbutton"**).**style**.**visibility**=**'visible'**;

}

***gameOver*** = **true**;

**return**;

}

*// we lock the piece*

***board***[**this**.**y**+***r***][**this**.**x**+***c***] = **this**.**color**;

}

}

*// remove full rows*

**for**(***r*** = 0; ***r*** < ***ROW***; ***r***++){

**var** isRowFull = **true**;

**for**( ***c*** = 0; ***c*** < ***COL***; ***c***++){

isRowFull = isRowFull && (***board***[***r***][***c***] != ***VACANT***);

}

**if**(isRowFull){

*// if the row is full*

*// we move down all the rows above it*

**for**( **y** = ***r***; **y** > 1; **y**--){

**for**( ***c*** = 0; ***c*** < ***COL***; ***c***++){

***board***[**y**][***c***] = ***board***[**y**-1][***c***];

}

}

*// the top row board[0][..] has no row above it*

**for**( ***c*** = 0; ***c*** < ***COL***; ***c***++){

***board***[0][***c***] = ***VACANT***;

}

*// increment the score*

***score*** += 10;

***level*** += 1;

**if**(***time***>300){

***time*** -= 50;

}**else**{

***time*** -= 5;

}

}

}

*// update the board*

*drawBoard*();

*// update the score*

***scoreElement***.**innerHTML** = ***score***;

***levelElement***.**innerHTML** = ***level***;

};

*//collision function*

*Piece*.**prototype**.collision = **function**(x,y,piece){

**for**(**var** r=0; r < piece.**length**;r++){

**for**(**var** c=0; c < piece.**length**;c++){

**if**(!piece[r][c]){ *//if the square is empty, we skip it*

**continue**;

}

**var** newX = **this**.**x** + c + x; *//coordinates of the piece after movement, the +x and +y at the end represent the*

**var** newY = **this**.**y** + r + y; *//new positions of the coordinates from this: Piece.prototype.collision = function(x,y,piece)*

*/\*conditions for the collision detection:*

*the first rule is for the left wall,*

*the second rule is for the right wall*

*the third rule is for the bottom wall\*/*

**if**(newX < 0 || newX >=***COL*** || newY >= ***ROW***){

**return true**;

}

*//skip newY < 0; eg. board[-1] will crash the game!*

**if**(newY < 0){

**continue**;

}

*//check if there is a locked piece on the board*

**if**(***board***[newY][newX] != ***VACANT***){

**return true**;

}

}

}

**return false**;

};

*//control the piece*

**document**.addEventListener(**'keydown'**,*CONTROL*);

**function** *CONTROL*(event){

**if**(event.**keyCode** == 37){

**p**.moveLeft();

***dropStart*** = Date.now();

}**else if**(event.**keyCode** == 38){

**p**.rotate();

***dropStart*** = Date.now();

}**else if**(event.**keyCode** == 39){

**p**.moveRight();

***dropStart*** = Date.now();

}**else if**(event.**keyCode** == 40){

**p**.moveDown();

}

}

*//drop the piece every 1 second*

**var *dropStart*** = Date.now();

**var *gameOver*** = **false**;

**var *time*** = 1000; *//seconds for falling*

**function** *drop*(){

**var** now=Date.now();

**var** delta = now - ***dropStart***;

**if**(delta > ***time***){

**p**.moveDown();

***dropStart*** = Date.now();

}

**if**( !***gameOver***){

requestAnimationFrame(*drop*);

}

}

## TETROMINOES.JS

U ovom fajlu se nalaze tetromino oblici, tj. struktura svakog od 7 oblika u zavisnosti rotacije.

Primer jednog od 7 oblika:

**const *I*** = [

[

[0, 0, 0, 0],

[1, 1, 1, 1],

[0, 0, 0, 0],

[0, 0, 0, 0]

],

[

[0, 0, 1, 0],

[0, 0, 1, 0],

[0, 0, 1, 0],

[0, 0, 1, 0]

],

[

[0, 0, 0, 0],

[0, 0, 0, 0],

[1, 1, 1, 1],

[0, 0, 0, 0]

],

[

[0, 1, 0, 0],

[0, 1, 0, 0],

[0, 1, 0, 0],

[0, 1, 0, 0]

]

];

## INDEX.HTML

**<html>**

**<head>**

**<title>Tetris Game JS</title>**

**<style>**

**#score,#level{**

**display: inline-block;**

**}**

**.scoreLook{**

**font-size: 25px;**

**font-weight: bold;**

**font-family: monospace;**

**text-align: center;**

**}**

**canvas{**

**display: block;**

**margin:0 auto;**

**}**

**button,#startbutton{**

**background-color: black;**

**color: white;**

**border: none;**

**bottom: 0;**

**height: 40px;**

**font-size: 12pt;**

**width: 90px;**

**alignment: center;**

**}**

**button:hover{**

**background-color: #0000ff;**

**}**

**#startbutton:hover{**

**background-color: #f00;**

**}**

**#dataLook{**

**text-align: center;**

**}**

**#hiddenbutton**

**{**

**visibility: hidden;**

**display: block;**

**}**

**</style>**

**</head>**

**<body>**

**<div class="scoreLook">Tetris</div><br>**

**<canvas id="tetris" width="200" height="400"></canvas>**

**<div class="scoreLook">**

**Score : <div id="score">0</div>**

**<br>**

**Level: <div id="level">1</div>**

**<br>**

**<input type="button" name="startbtn" id="startbutton" value="START" onclick="*drop*()">**

**</div>**

**<div id="dataLook">**

**<br>**

**<label>Show top 5 scores:<br><br></label>**

**<button id="showbutton">Show</button>**

**<button id="clearbutton1" style="margin-left: 20px" onclick="*clearData1*()">Hide</button>**

**<br><br>**

**<div id="showScores"></div>**

**<br>**

**<label>Recent players:<br><br></label>**

**<button id="showbuttonlast">Show</button>**

**<button id="clearbutton2" style="margin-left: 20px" onclick="*clearData2*()">Hide</button>**

**<br><br>**

**<div id="showScoresLast"></div>**

**<br>**

**<div id="hiddenbutton">**

**<button id="rememberbutton" onclick="*insertData*()">Remember me!</button>**

**</div>**

**<div id="showAllScores"></div>**

**</div>**

**<script src="tetrominoes.js"></script>**

**<script src="tetris.js"></script>**

**<script src="ajax.js"></script>**

**</body>**

**</html>**

## UBACIVANJE KORISNIKA U BAZU PODATAKA

Kada se popuni kanvas, izbaci se promt za ime. Transformišemo što su do sada bile lokalne varijable u globalne preko 2 različita načina. Ime napravimo globalnim dodavanjem window. na varijablu name a rezultat ubacujemo u cookie. Ukoliko korisnik potvrdi svoje ime i rezultat, do sada skriveni div sa button-om koji služi za ubacivanje u bazu se pojavljuje.

**if**(**this**.**y** + ***r*** < 0){

alert(**"Game Over!"**);

*//stop request animation frame*

**window**.**name** = prompt (**'Please enter your username: '**);

**if**(**name**==**""**){

**name**=prompt (**'This field cant be empty. Please enter your username: '**);

}

**else**{

**window**.**confirmName**=confirm(**"This is your username: "**+**name**+

**"\n This is your score: "** +***score***);

}

**window**.**getScore**=***score***;

**document**.**cookie**=**getScore**;

**if**(**confirmName**==**true**)

{

**document**.getElementById(**"hiddenbutton"**).**style**.**visibility**=**'visible'**;

}

***gameOver*** = **true**;

**return**;

}

## UNOŠENJE PODATAKA U BAZU

U ovom projektu koristimo ajax kako bi korisnik mogao videti i uneti podatke u bazu podataka bez osvežavanja stranice.

**ajax**

**function** *insertData*() *// this is used for inserting data*

{

**var** xmlhttp;

**if** (**window**.XMLHttpRequest)

{*// code for IE7+, Firefox, Chrome, Opera, Safari*

xmlhttp=**new** XMLHttpRequest();

}

**else**

{*// code for IE6, IE5*

xmlhttp=**new** ActiveXObject(**"Microsoft.XMLHTTP"**);

}

xmlhttp.onreadystatechange=**function**()

{

**if** (xmlhttp.**readyState**==4 && xmlhttp.**status**==200)

{

**document**.getElementById(**"showAllScores"**).**innerHTML**=xmlhttp.**responseText**;

}

};

**var** uname=**name**;

**var** currentscore=**document**.**cookie**;

xmlhttp.open(**"POST"**,**"insert\_scores.php"**,**true**);

xmlhttp.setRequestHeader(**"Content-type"**, **"application/x-www-form-urlencoded"**);

xmlhttp.send(**"uname="**+uname+**"&currentscore="**+currentscore);

}

Pozivamo prethodno definisane podatke, preko window. i cookie. Smeštamo ih u varijable koje ajax šalje na insert scores.php. Funkcija insertData() je vezana za dugme u skrivenom div-u.

**php - insert\_data**

**<?php**

**include**(**"db\_config.php"**);

$name = **""**;

$score = **""**;

**if**(**isset**($\_POST[**'uname'**]))

$name=*mysqli\_real\_escape\_string*($connection,$\_POST[**'uname'**]);

**if**(**isset**($\_POST[**'currentscore'**]))

$score=*mysqli\_real\_escape\_string*($connection,$\_POST[**'currentscore'**]);

**if**(!**empty**($name))

{

$sql = **"INSERT INTO userdata(username,score,datetime) VALUES ('**$name**','**$score**',***CURRENT\_TIME***)"**;

$result= *mysqli\_query*($connection,$sql) **or die**(*mysqli\_error*($connection));

**echo "<br>Your score and level have been remembered!"**;

}

**else**{

**echo "empty!<br>"**;

}

**?>**

## PRIKAZIVANJE PODATAKA IZ BAZE

**ajax**

document.getElementById(**"showbutton"**).addEventListener(**'click'**,**function**(){*loadData*(1,**'showScores'**)});

document.getElementById(**"showbuttonlast"**).addEventListener(**'click'**,**function**(){*loadData*(2,**'showScoresLast'**)});

Prvo dodajemo event listener-e za dugmad na početnoj strani. Takođe smo definisali da se rezultati iz baze pojave u određenim div-ovima.

**function** *loadData*(sql,div) *//function for displaying data*

{

**var** xmlhttp;

**if** (**window**.XMLHttpRequest)

{*// code for IE7+, Firefox, Chrome, Opera, Safari*

xmlhttp=**new** XMLHttpRequest();

}

**else**

{*// code for IE6, IE5*

xmlhttp=**new** ActiveXObject(**"Microsoft.XMLHTTP"**);

}

xmlhttp.onreadystatechange=**function**()

{

**document**.getElementById(div).**innerHTML**=**'<img src="ajax\_loader.gif"'**;

**if** (xmlhttp.**readyState**==4 && xmlhttp.**status**==200)

{

**document**.getElementById(div).**innerHTML**=xmlhttp.**responseText**;

}

};

xmlhttp.open(**"GET"**,**"get\_scores.php?sql="**+sql,**true**);

xmlhttp.send();

}

**php-get\_scores**

**<?php**

**include**(**"db\_config.php"**);

$sql = **""**;

$sqls[1] = **"SELECT \* FROM userdata ORDER BY score DESC LIMIT 5"**;

$sqls[2] = **"SELECT \* FROM userdata ORDER BY id DESC LIMIT 5"**;

**if**(**isset**($\_GET[**'sql'**]))

$sql=*mysqli\_real\_escape\_string*($connection,$\_GET[**'sql'**]);

$result= *mysqli\_query*($connection,$sqls[$sql]) **or die**(*mysqli\_error*($connection));

**if** (*mysqli\_num\_rows*($result)>0)

{

**while** ($record = *mysqli\_fetch\_array*($result,***MYSQLI\_BOTH***))

{

**echo "user:** $record[username]**,<br>score:** $record[score]**<br><hr style='width: 10%'>"**;

}

}

**?>**

Definisali smo da želimo da nam dugmad izbacuju iz baze podatke o najboljih 5 igrača i poslednjih 5 igrača.

Pored ovoga takođe su definisane funkcije da se ovi podaci sakriju kako ne bi zauzimali mesto.

**function** *clearData1*() *//functions for clearing out the divs*

{

**document**.getElementById(**'showScores'**).**innerHTML** = **""**;

}

**function** *clearData2*()

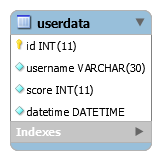
{

**document**.getElementById(**'showScoresLast'**).**innerHTML** = **""**;

}

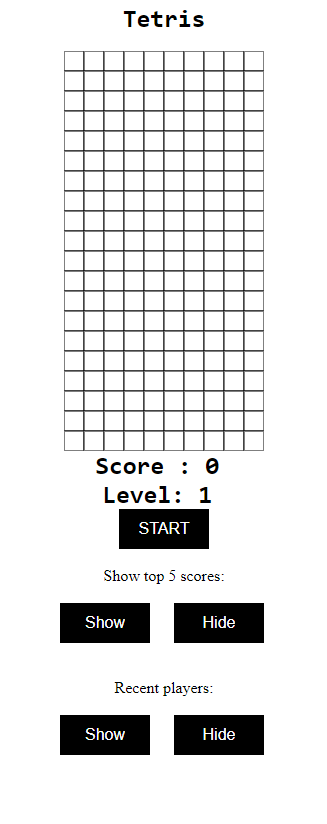
# STRUKTURA BAZE PODATAKA

Baza podataka u ovom projektu je veoma jednostavna. Treba nam samo jedna tabela preko koje ili ispisujemo ili upisujemo osnovne podatke o korisniku.



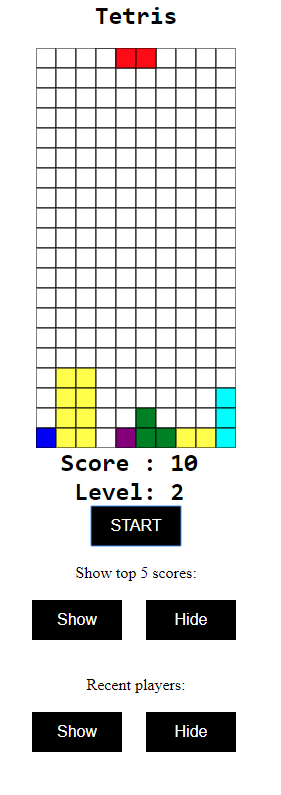
# OPIS FUNKCIONALNOSTI PROJEKTA

## POČETNA STRANA

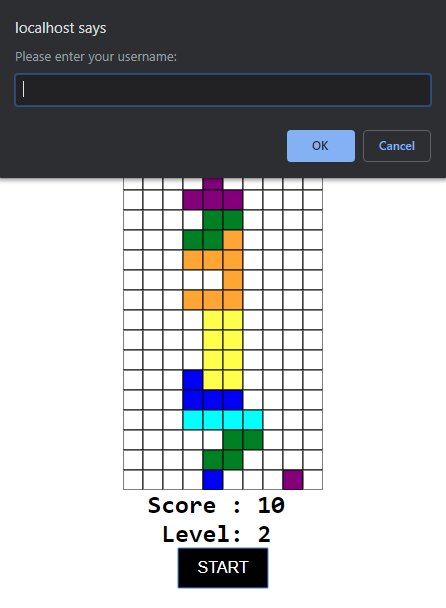


## 

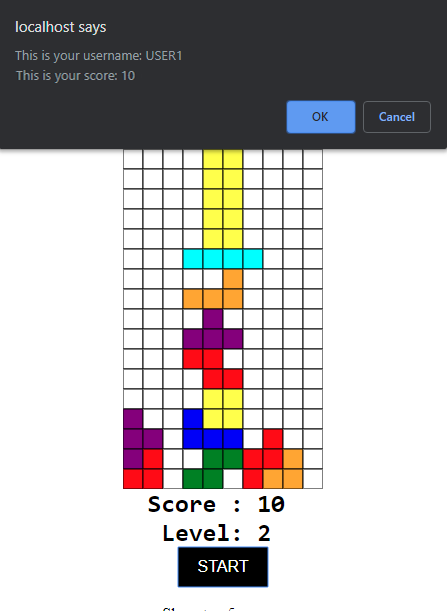
## PRITISAK START BUTTON-a, IGRA U PROCESU



## PROMPT ZA KORISNIKOVO IME



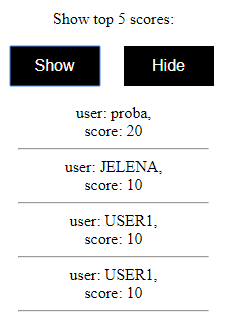
## POTVRDA PODATAKA



## POTVRDA PODATAKA - CLICK NA OK

## CLICK NA REMEMBER ME BUTTON, CLICK NA SHOW KOD RECENT PLAYERS

## CLICK NA SHOW KOD TOP 5 HIGHSCORES



# KORIŠĆENA LITERATURA

W1: <http://www.w3schools.com/>

W2: <https://developer.mozilla.org/en-US/>

W3: <http://stackoverflow.com/>