UGNews:

By: Sm Shahniaz
Course: Front-End Fundamentals
ID:100251

This is a gaming news-providing website.

Note: Please run the index.html for this website using LiveServer extension of VSCode.

Github Repo: https://github.com/sshahniaz/UGNews

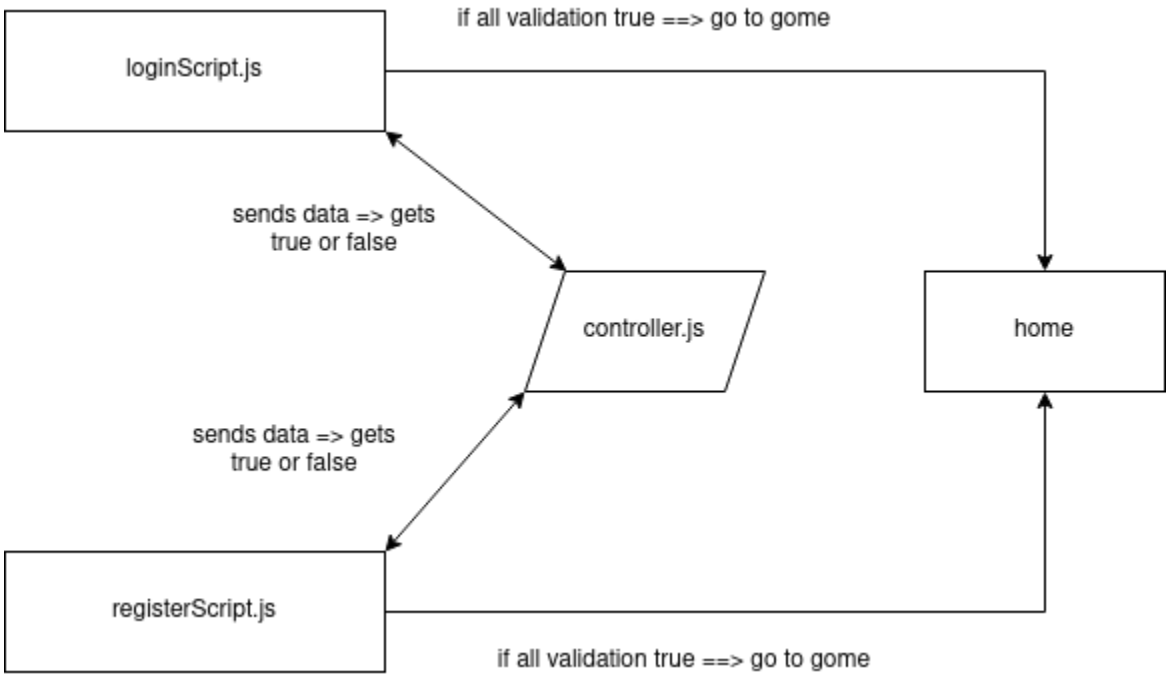
Code Snippets:

The code aims for login and registration validation of users for the website.

The code is divided into three main modules for ease of access and reusability into the following three files:

- loginScript.js: for the login functionality.
- registerScript.js : for registering.
- controller.js: contains validation code that sends messages to login and registers.

Code Flow:

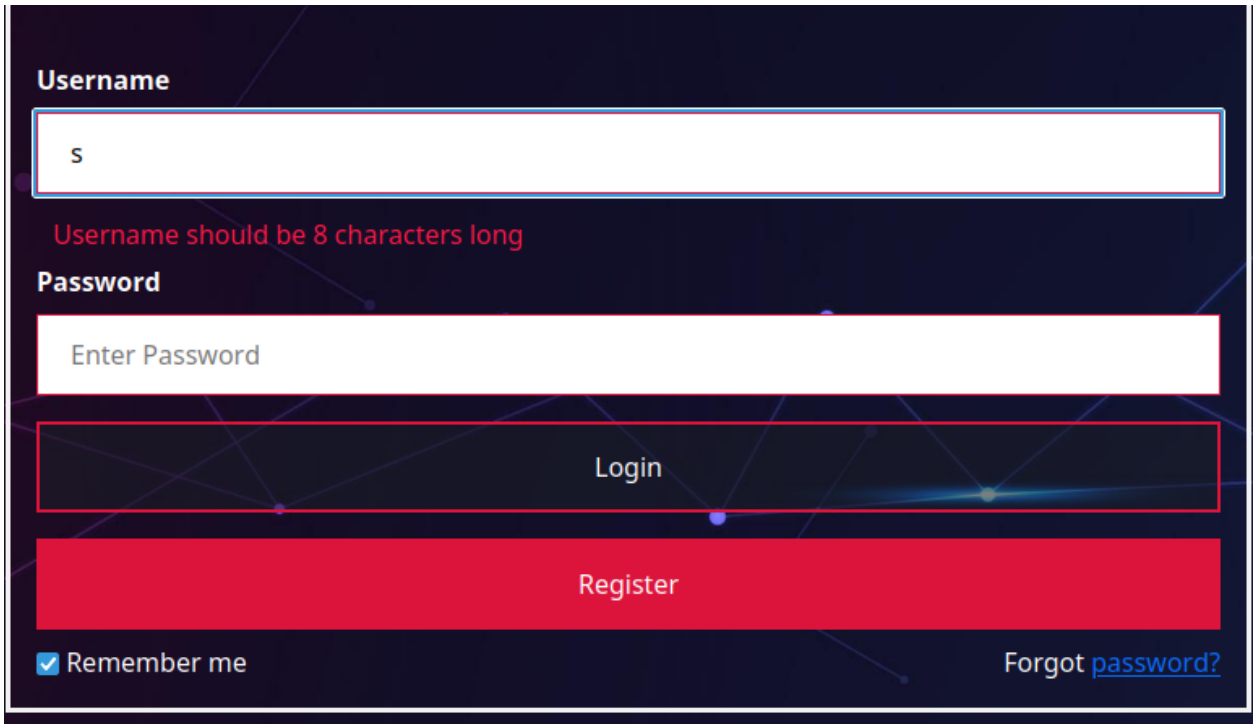


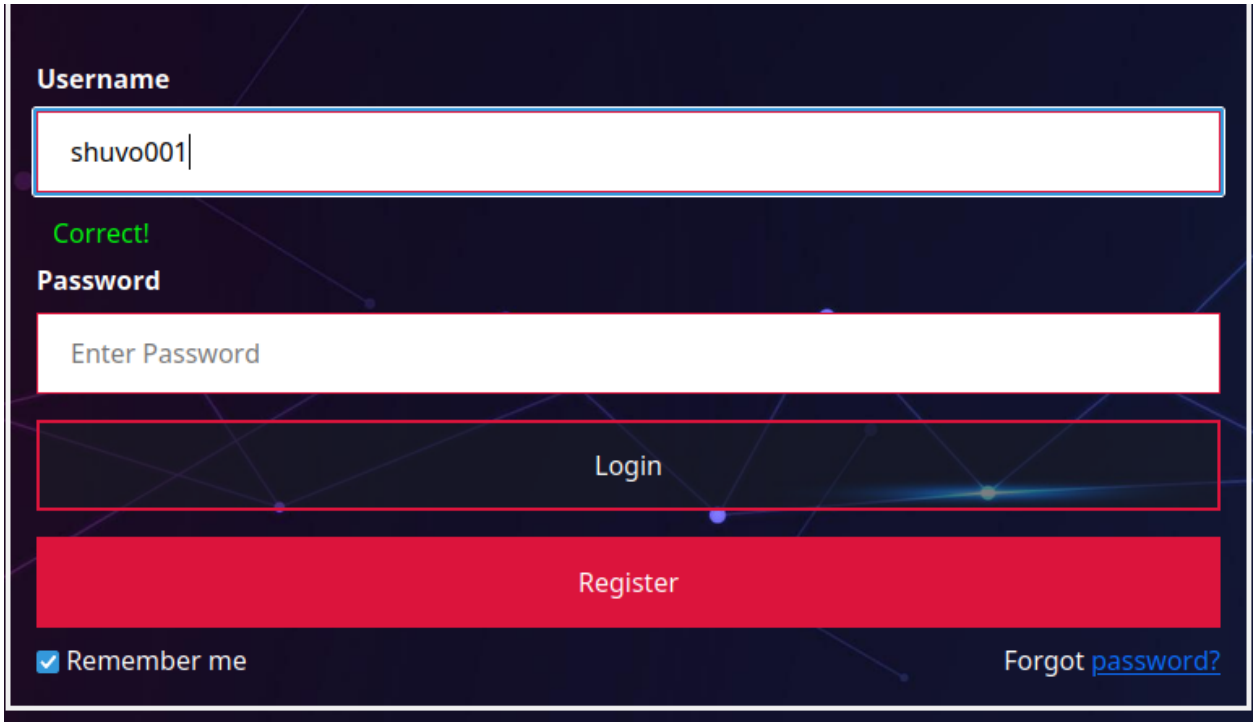
The code explanation with the flow:

Users go to the Login/Registration page to input data. As they input the data, the data is simultaneously read and validated in the following way. This is an example of the user logging in:

The user clicks the login button on the index page, which redirects to the login page.

When a user enters data, the data is read, and simultaneously errors or valid is displayed:





As the user enters data, the following code activates:

```
import {
 uNameCheck,
 emailCheck,
 pswdCheck,
 confirmPasswordCheck,
} from "./controller.js";

let form = document.querySelector("#loginForm");

//Check for the username
let uName = document.getElementById("uname");
//let unameError = document.getElementById("unameError");
let err = document.querySelectorAll(".msg");
//unameError.style.display = "block";
uName.addEventListener("keypress", (event) => {
 let text = uName.value + `${event.key}`;
 let error = uNameCheck(text);
 if (error[0] === false) {
   err[0].style.display = "block";
```

```
    err[0].classList.add("errMsg");
    uName.classList.add("error");
    err[0].classList.remove("succMsg");
    uName.classList.remove("success");
    err[0].innerHTML = error[1];
  } else {
    err[0].classList.add("succMsg");
    uName.classList.add("success");
    err[0].classList.remove("errMsg");
    uName.classList.remove("error");
    err[0].innerHTML = error[1];
  }
});
```

This code gets the username, and as there is an event listener that listens for every keypress in a field that simultaneous update is possible. The value of the field, in this case, the username, is then stored in the variable text and then calls the function uNameCheck, and the variable is passed here. The uNameCheck function is imported from the controller.js file containing the function. The function then returns an array containing the validation result(i.e. true/false) and a message corresponding to it.

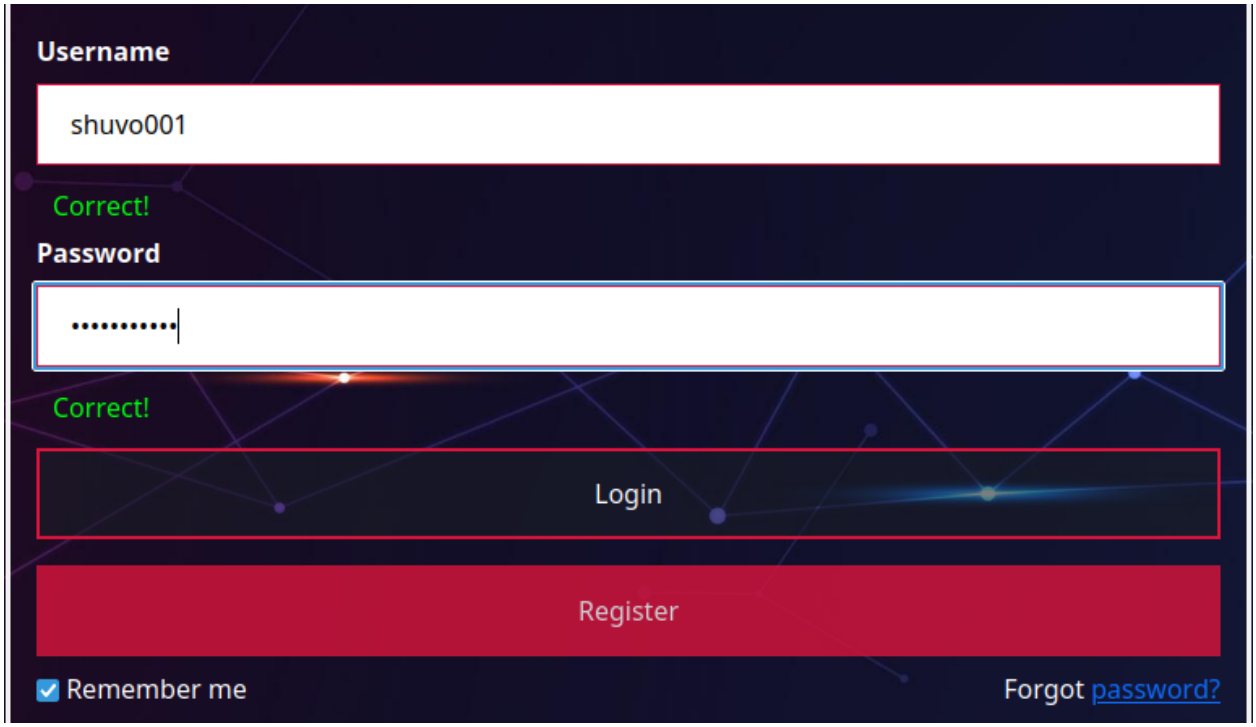
Now let's go to controller.js and see the validation check for uNameCheck function.

```
//Username regex check.

export const uNameCheck = (text) => {
 let condition1 = /(?=.*[a-zA-Z0-9])/;
 let condition2 = /(?=.*[\s])/; //for the space
 let condition3 = /.{8,}/; //atleast 8 characters
 let error = [];

 // uNameError.style.color = "red";

 if (text == "" || text == null) {
   // uNameError.style.display = "block";
   error.push(false, "Username should not be empty");
 } else if (!text.match(condition1)) {
   // uNameError.style.display = "block";
   error.push(false, "Username should not contain special characters");
```

```
 } else if (text.match(condition2)) {
   // uNameError.style.display = "block";
   error.push(false, "Username should not have space");
 } else if (!text.match(condition3)) {
   error.push(false, "Username should be 8 characters long");
 } else {
   error.push(true, "Correct!");
 }
 return error;
 // if(isNaN(text)){
 //     uNameError.style.display = "block";
 //     uNameError.innerText = "Username should not only contain numbers";
 //
};
```

As seen above, the data is then checked if it is empty, then checked for the regex conditions of alphanumeric without special characters, without spaces and a minimum of 8 characters. The result and a message are then pushed into an array with the .push() function, which is used in the loginScript to display the message and DOM style changes, such as message colour changes and border colour changes.

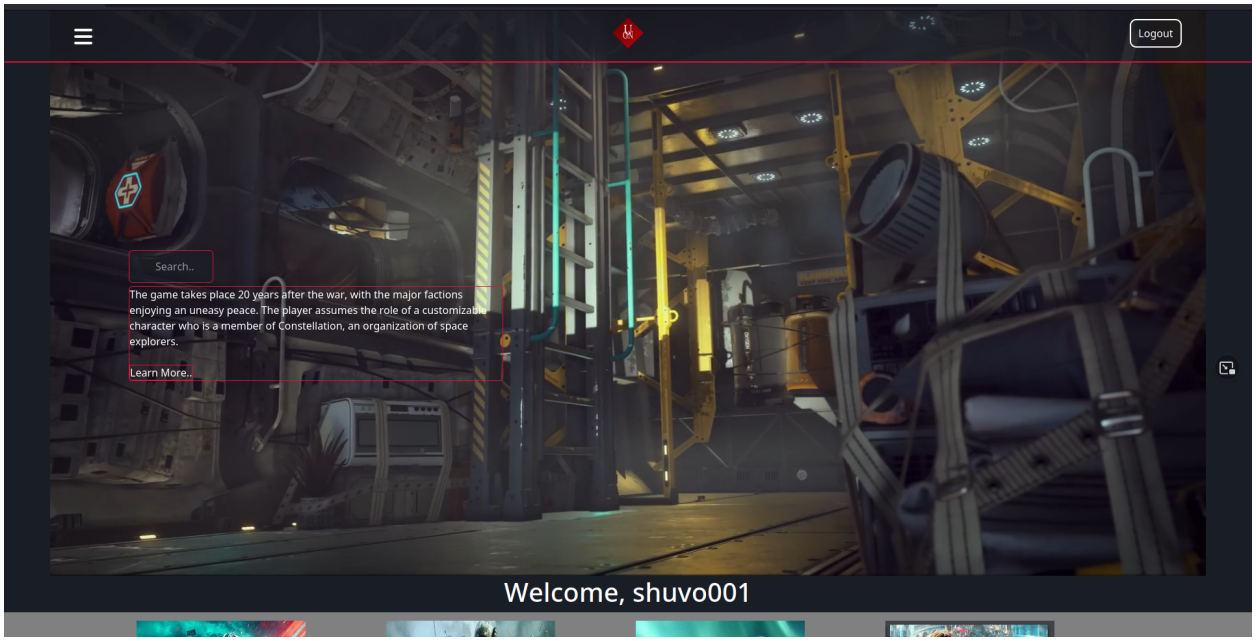
When the validation is passed for username and password:



The following code is activated, redirecting to the home page if the validation is valid.

```
//Final Submit

form.addEventListener("submit", (e) => {
 e.preventDefault();

 let isFormValid = uNameCheck(uName.value)[0] &&
pswdCheck(password.value)[0];

 if (isFormValid) {
   sessionStorage.setItem("userName", uName.value);
   window.location.href = "./../../../home.html"; //Regirect to home page
 }
});
```

Home page(see bottom with welcome and top logout button):



The same procedure is followed for all checks in both login and registration pages.