**HTML**

**Main Page (/login)**

Make a page that asks for username, send it to /game using any method

**/game**

Watch the demo video. I address the following aspects of the HTML. . .

1. The tab should be titled “Shut The Box”.

2. You should have a header displaying a heading “Shut The Box”.

3. You should have a section describing the rules using numbering i, ii, iii, iv, . . .

You can copy my rules or you can improve on my description, whichever you prefer.

4. You should have a heading saying “Dice roll”.

5. Underneath there should be a button saying “Roll dice” and there should be a space for the

result of a dice roll. I used a <span> element for the space where results will appear.

6. You may want to use a <fieldset> element to enclose your dice button and result.

7. You should have a heading saying “Box selection”.

8. Underneath there should be a <table>.

The table head should contain the numbers 1, 2, 3, . . ., 8, 9.

The table body should contain corresponding checkboxes.

There should be no table foot.

9. There should be two buttons enclosed in a <fieldset> element saying:

“Submit box selection” and “I give up / I can’t make a valid move”.

10. There should be a footer with copyright information.

11. If username is not set, redirect to /login

**/scores**

The tab should be titled “Shut The Box”.

• You should have a header displaying a heading “Shut The Box”.

• You should have a section with a heading saying “Scores”.

• The section should have a paragraph saying “Well done! Here are the scores so far...”

• The section should have a paragraph where the scores will be displayed. It is probably better to leave this paragraph empty originally.

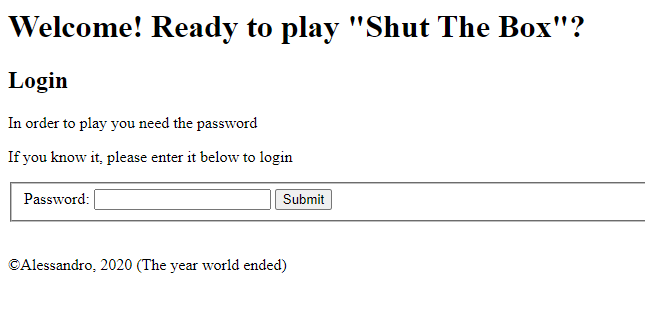
• There should be two <fieldset> elements.

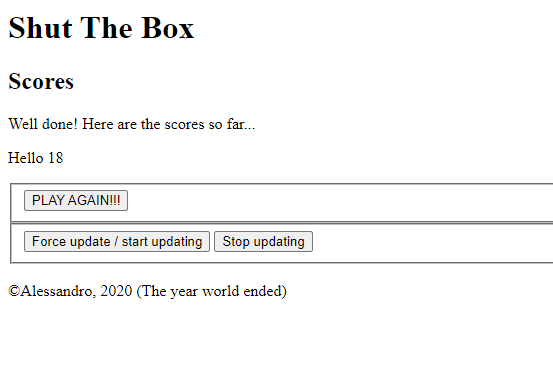
– The first should have a button saying “PLAY AGAIN!!!”.

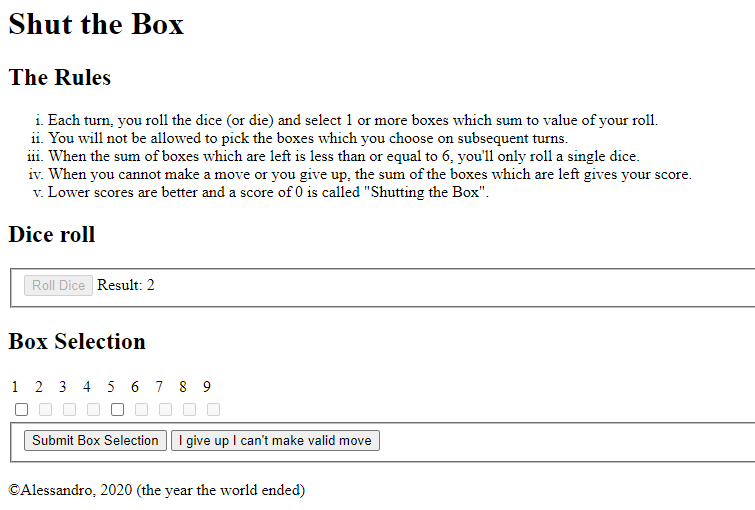
– The second <fieldset> element should contain two buttons with the text indicated above.

• There should be a footer with copyright information.

• If username is not set, redirect to /login







**Marking Criteria**

Here’s how your HTML will be graded. . .

• /login is worth 1 point.

• 1. - 2. are worth 1 point.

• 3. is worth 1 point.

• 4. - 6. are worth 1 point.

• 7. - 8. are worth 2 points.

• 9. - 10. are worth 1 point.

•/scores is worth 5 points

**JavaScript**

**/game**

Watch the demo video. I address the following aspects of the JavaScript. . .

1. Clicking the “Roll dice” button causes the result of a dice roll to be displayed.

2. Clicking on a number or the corresponding checkbox causes the checkbox to become checked

or unchecked.

3. When rolling the dice, the “Submit box selection” button is disabled.

When selecting boxes, the “Roll dice” button is disabled.

4. Upon clicking “Submit box selection”

1. either an alert message is displayed telling us we have made an invalid move:

The total of the boxes you selected does not match the dice roll.

Please make another selection and try again.

1. or our submission is successful. In this case,
   1. we disable the used checkboxes;
   2. if the remaining boxes sum to less than or equal to six, we start to use one die instead of two;
   3. we go back to rolling the dice and the last dice roll result is no longer displayed.

5. Upon clicking, “I give up / I can’t make a valid move” we get an alert telling us our score.

It should also make an AJAX request.

• The AJAX request should have its method set to POST and it should send the user’s username

and the user’s score to a separate page /score.

• /score should write the information it receives to the db, db will have the following structure: username varchar 255 and score int 11.

• After all this has been done successfully, we should be redirected to a separate page /scores

**/scores**

In terms of the JavaScript, scores.js should make sure the scores are updated every 10 seconds

(without refreshing the page) and that the buttons have the correct functionality.

• The “PLAY AGAIN” button should redirect a user to /login.

• “Force update / start updating” should update the scores immediately.

After clicking this button, the scores should continue to update every 10 seconds.

• “Stop updating” should stop the scores from being automatically updated.

You just need to implement an AJAX GET request to get the relevant information from server (/fetch)

**/login**

Checks for username

(a) alphanumeric characters: a, b, c, . . ., z, A, B, C, . . ., Z, 0, 1, 2, . . ., 9, or

(b) a character appearing in the following string: "\_.!+-\*'`|&%#~^".

**Marking Criteria**

Here’s how your JavaScript will be graded. . .

• 1., 2., 3. are worth 2 points each.

• 4.a) is worth 2 points.

• 4.b)i) is worth 2 points.

• 4.b)ii) and 4.b)iii) are worth 1 point each.

• 5. is worth 2 points.

• /scores is worth 5 points

• /login is worth 2 points

**Python**

* /submit should receive username & score using POST and add a row in db
* /fetch should display username & score in ascending order separated by <br>

**Marking Criteria**

Here’s how your Python & DB will be graded. . .

* 2.5 points for submit
* 2.5 points for /fetch

**Note:** JQuery + any other library not allowed, Use xhttp requests. use media queries (If necessary)