**ITS290F Human Computer Interaction & User Experience Design**

**Lab 3. HTML DOM (Part 2)**

What you’ll learn in this lab:

* HTML DOM (Document Object Model)
* Coding with CodePen

**1. Readings**

Three suggested readings from w3schools

1. JavaScript HTML DOM tutorial

<https://www.w3schools.com/js/js_htmldom.asp>

1. JavaScript and HTML DOM reference

<https://www.w3schools.com/jsref/default.asp>

1. HTML Basic

https://www.w3schools.com/html/default.asp

**2. Short Questions**

**Question 1**

User experience (UX) design is the process of creating products that provide meaningful and relevant experiences to users. This involves the design of the entire process of acquiring and integrating the product, including aspects of branding, design, usability, and function.

What does a UX designer actually do? Read the following article and write down the six responsibilities of a UX designer.

<https://theblog.adobe.com/what-does-a-ux-designer-actually-do/>

|  |
| --- |
| **Six responsibilities of a UX designer** |
| 1. Product Research  2. Creating Personas and Scenarios  3. Information Architecture  4. Creating Wireframes  5. Prototyping  6. Product Testing |

**Question 2**

|  |
| --- |
| Following Question 1, a person who represents a target user group that would interact with a product (or system) in a given scenario is called \_\_\_. |
| Personas |

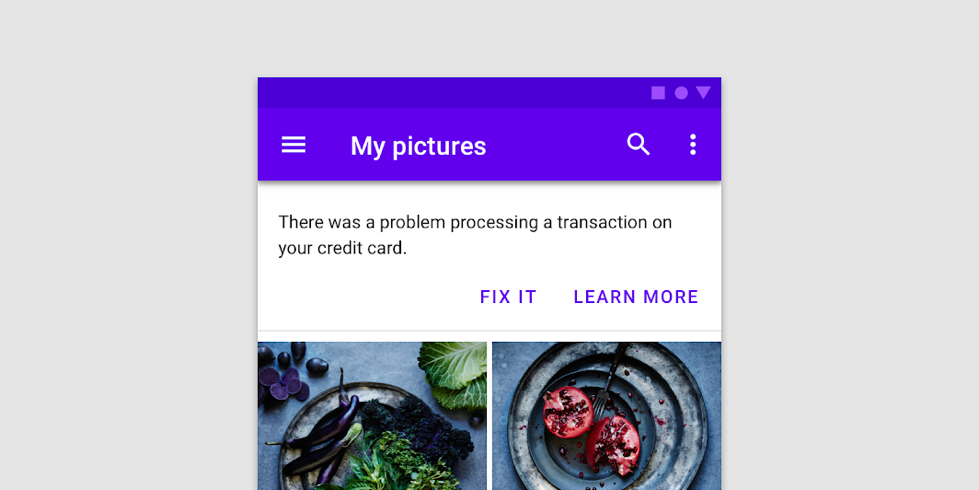
**Question 3**

|  |
| --- |
| How does a UX designer find out the problems the users experience when they interact with a product? |
| By doing Product testing |

**Question 4**

A banner displays an important, succinct message, and provides actions for users to address (or dismiss the banner). It requires a user action to be dismissed. Study the guidelines of the Banners UI component in Google’s Material Design.

[https://material.io/components/banners/#](https://material.io/components/banners/)



|  |
| --- |
| What are the guidelines of dismissing banners? |
| Banners may have one or two low-emphasis text buttons |

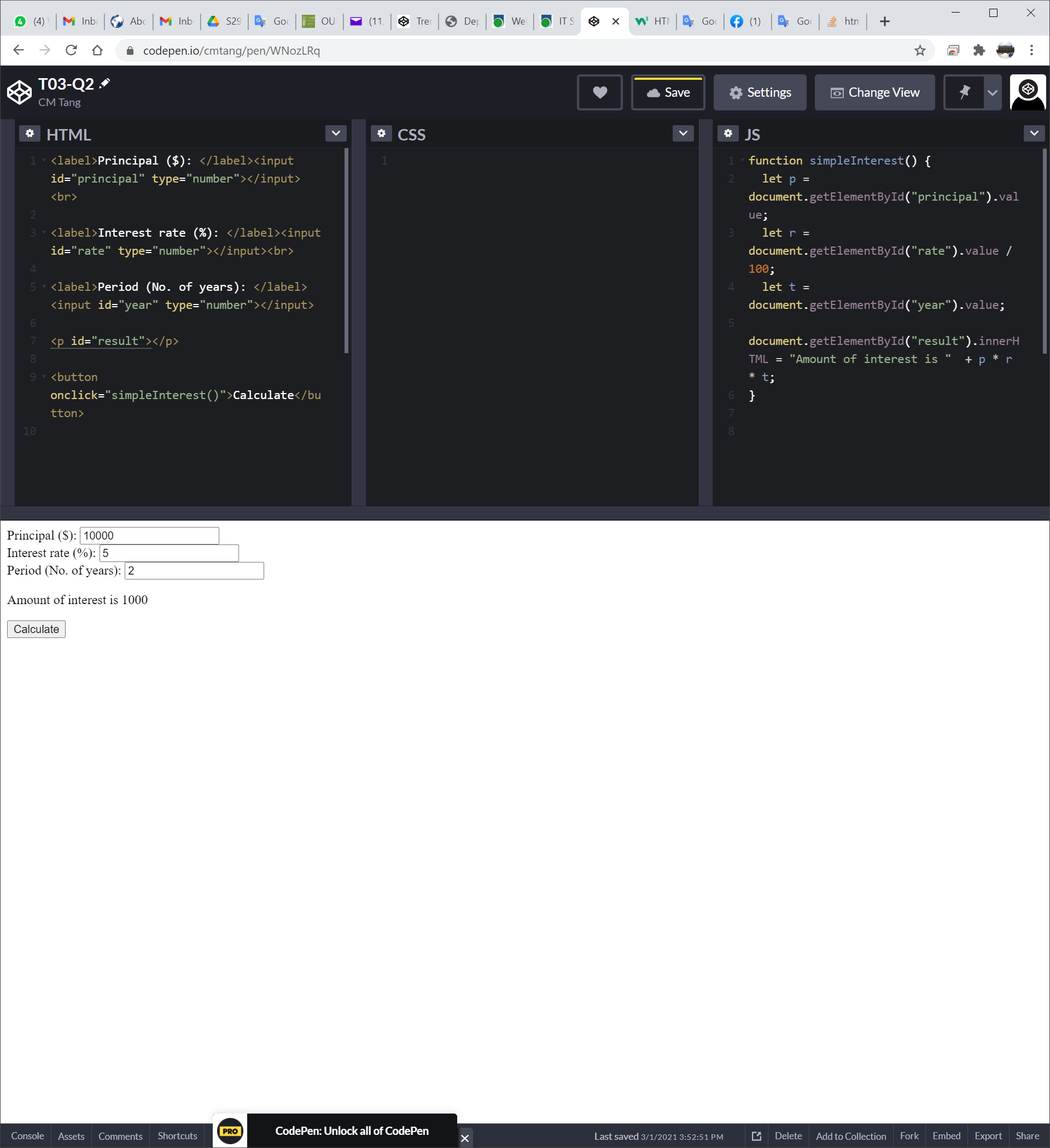
**3. Lab Exercises**

On your own, follow the instructions to finish the UI coding exercises. You may edit the code with CodePen by copying and pasting the provide HTML/JS code in the worksheet to the corresponding CodePen windows. Put your solution code to the solution textboxes in the worksheet when finished.

You may refer to the online JavaScript HTML DOM tutorial from w3schools (<https://www.w3schools.com/js/js_htmldom.asp>) and the reference (<https://www.w3schools.com/jsref/default.asp>).

**Exercise 1**

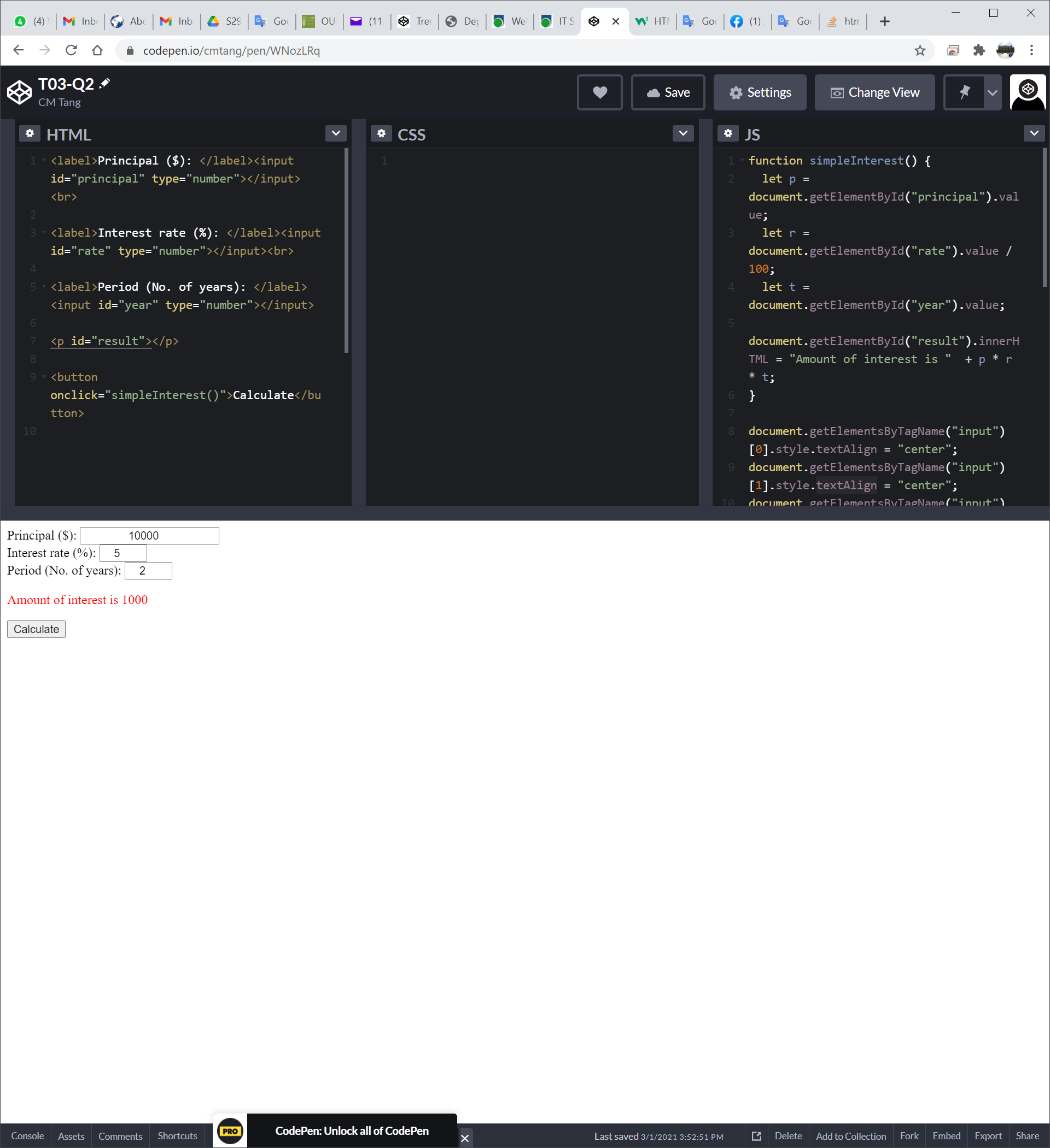
You are a UX designer who is designing a web-based system that calculates interest on a loan. You have created a dummy user interface in the beginning as shown below and asked for feedback from a user group on the user experience.



After analysing their feedback, you decide to make the following changes on the UI.

1. Align all input values to the center (**Hints:** look for the textAlign property).
2. Fix the width of the text fields of “Interest rate” and “Period” to 50px (**Hints:** look for width of the style property).
3. Pre-input “5” (5%) as a default value of the Interest rate.
4. Change the message of the calculated interest to **red** color (**Hints:** look for color of the style property).

The updated UI will look similar to the follows:



Implement the above four requirements by appending your code to the given JS solution box next page. The simpleInterest() function for computing the interest has been finished for you. There is no need to modify the HTML code.

Beside the getElementById() method you may be adopted, you MUST also include at least one getElementsByTagName() method in your code to show your proficiency.

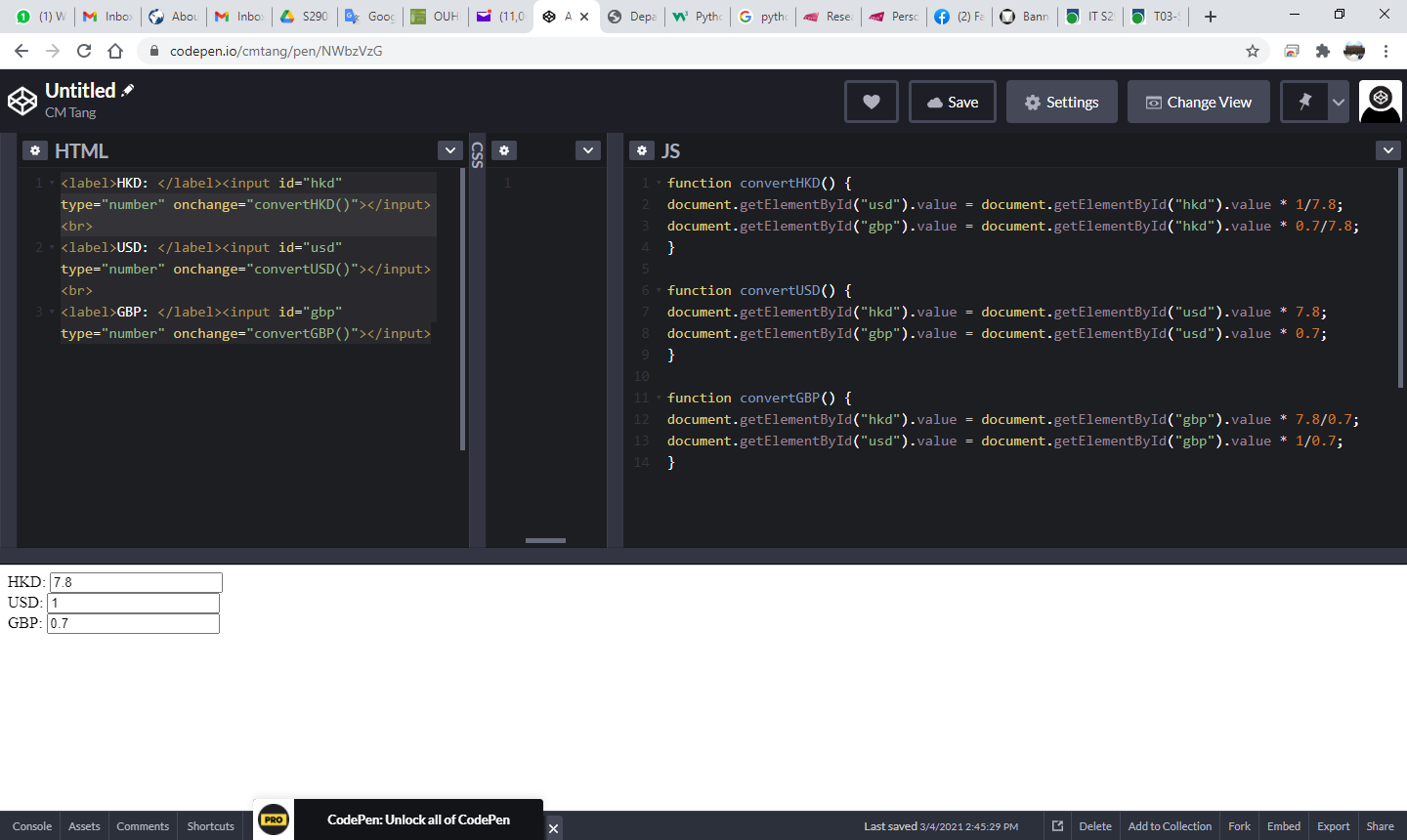
|  |
| --- |
| **Given HTML code for the UI (No update is needed)** |
| <label>Principal ($): </label><input id="principal" type="number"></input><br>  <label>Interest rate (%): </label><input id="rate" type="number"></input><br>  <label>Period (No. of years): </label><input id="year" type="number"></input>  <p id="result"></p>  <button onclick="simpleInterest()">Calculate</button> |

|  |
| --- |
| **Your Solution in JS** |
| function simpleInterest() {  let p = document.getElementById("principal").value;  let r = document.getElementById("rate").value / 100;  let t = document.getElementById("year").value;  document.getElementById("result").innerHTML = "Amount of interest is " + p\*r\*t;  }  // put your code below  document.getElementById(“year”).style.width = “50px”; document.getElementById(“rate”).style.width = “50px”;  document.getElementById(“year”).style.textAlign = “center”;  document.getElementById(“rate”).style.textAlign = “center”;  document.getElementById(“principal”).style.textAlign = “center”;  document.getElementById(“result”).style.color = “red”; |

**Exercise 2**

Given a currency exchange system that converts an amount over three currencies – HKD, USD and GBP. Its UI accepts an amount of one of the currencies and then converts it to the amounts of other two currencies according to a fixed exchange rate. Assume the exchange rate for HKD, USD and GBP are 7.8, 1.0 and 0.7, respectively. For example, if a user enters 1 in the USD field, the corresponding HKD and GBP fields will then be updated to 7.8 and 0.7 accordingly.

The UI of the system is as the follows:



There is no “Submit” button, but the input fields will invoke a function associated with the onchange event, when the value of the input field is changed. Implement the missing functions in JS (**Hints**: get and set values in the input fields with the .value property).

|  |
| --- |
| **Given HTML code for the UI (no update is needed and do not update)** |
| <label>HKD: </label><input id="hkd" type="number" onchange="convertHKD()"></input><br>  <label>USD: </label><input id="usd" type="number" onchange="convertUSD()"></input><br>  <label>GBP: </label><input id="gbp" type="number" onchange="convertGBP()"></input> |

|  |
| --- |
| **Your Solution in JS** |
| function convertHKD() {  let hkd = document.getElementById("hkd").value;  let usd = hkd / 7.8;  let gbp = usd \* 0.7;    document.getElementById("usd").value = usd;  document.getElementById("gbp").value = gbp;``  }  function convertUSD() {  let usd = document.getElementById("usd").value;  let hkd = usd \* 7.8;  let gbp = usd \* 0.7;    document.getElementById("hkd").value = hkd;  document.getElementById("gbp").value = gbp;``  }  function convertGBP() {  let gbp = document.getElementById("usd").value;  let usd = gbp / 0.7;  let hkd = usd \* 7.8;    document.getElementById("hkd").value = hkd;  document.getElementById("usd").value = usd;``  } |

**Exercise 3**

The UI in Exercise 2 supports only three currencies. You then want to add more currencies to enrich the functionality of the system. Besides HKD, USD and GBP, you also want to add EUR, CNY, JPY, AUD, NZD, CHF and CAD to the application so that it supports 10 major currencies. However, you found that the UI that lists 10 currency labels and 10 text fields is too messy, and users usually interested in the conversion of one specific currency to another only. **Design** a new dummy UI to satisfy the user’s experience. No need to make it functional.

|  |
| --- |
| **Your Solution UI in HTML** |
| <select id="transferFrom">  <option value = "USD">USD</option>  <option value = "HKD">USD</option>  <option value = "GBP">USD</option>  <option value = "EUR">USD</option>  <option value = "CNY">USD</option>  <option value = "JPY">USD</option>  <option value = "AUD">USD</option>  <option value = "NZD">USD</option>  <option value = "CHF">USD</option>  <option value = "CAD">USD</option>  </select>  <input id="transTo" type="number"></input>  <label>to</label>  <select id="transferFrom">  <option value = "USD">USD</option>  <option value = "HKD">USD</option>  <option value = "GBP">USD</option>  <option value = "EUR">USD</option>  <option value = "CNY">USD</option>  <option value = "JPY">USD</option>  <option value = "AUD">USD</option>  <option value = "NZD">USD</option>  <option value = "CHF">USD</option>  <option value = "CAD">USD</option>  </select>  <input id="transToOutput" type="number"></input> |

|  |
| --- |
| **Justify why your new design is better (50-100 words)** |
| Because In this design: The user can find they want accurately and more efficiently. By selecting the currency they want. The user can have a more friendly experience. |

**Submission**

You will get 0 to 3% of OCAS score upon completing the lab exercises.

1. Put your solution code inside the answer boxes and save the file.
2. Upload this file via the corresponding submission link on OLE.

**Note:**

1. The submission deadline is 23:59 of the forthcoming Thursday. The submission link will be closed after the deadline.
2. Your submission not necessary to be all correct but should show your efforts. We grade your work according to its quality.
3. Make sure you submitted a correct file. You can resubmit your work before the deadline.

1. DO NOT modify the layout and format of this document, and DO NOT save it as other file formats, i.e., PDF. Your work will be extracted using an automated document parsing tools for grading.

--- End ---