**Student ID:** \_\_\_\_\_**610732\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Full Names:** \_\_\_**Margulan Aubakirov**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Web Application Programming

(CS472)

(40 points)

(May 2020)

Instructor: O. Kalu

Midterm Exam

1. The exam duration is 2 hours.
2. The exam is an online, computer-based exam; so you may use a computer for both the Part 1 (theory) and Part 2 (coding) tasks.
3. **This exam paper document is a copyrighted material and so it must not be copied or reproduced or transferred or shared or distributed**.
4. You are expected to type all your answers for Part 1, as text into this document. You may also use an IDE or any Code Editor tool of your choice to implement your solutions for the questions in the Part 2 (Web Application Coding).
5. During the exam, if you have any question, please **send it via a Chat to me** in the Microsoft Teams app.
6. Upon completion, put your entire Exam (including the projects/folders with your **source code** and this document with your typed-in answers (in either Microsoft Word or Adobe PDF format, only)**)** into a single zip file named **MidtermExam.zip**, and submit to Sakai, under the Assignment titled, “Midterm Exam”.
7. **NOTE**: ***If you fail to submit your exam to Sakai because it has past the Submission due time, and you then email it instead, then be aware that your maximum possible score will be 80%,and only if your work scores up to that level.***

--------------------------------------------------------------------------------------------------------------------

Make sure to include the screenshots of your results, where it is required.

--------------------------------------------------------------------------------------------------------------------

(CS472 - WAP)

(May 2020)

Part 1 - Theory (10 points)

**Part I – Theory (True/False, Short answers, Multiple choice questions):** (10 points)

**Note:** *For these questions, please follow the instructions given for each individual question and do type all your answers right here in this document.*

1. (7 points) Answer the following questions with True or False. For each answer, give a rationale (i.e. If True state how, if False state why. No rationale, earns you just half of the points if your True/False answer is correct, and zero point if your True/False answer is incorrect).
   1. (1 point) Consider the following code:

printHello("John");

var printHello = function(name) {

console.log("Hello " + name);

}

It will print out “Hello John”, to the console.

True or False?

\_**False**\_\_\_\_\_\_\_\_\_\_

**Rationale:**

*The value of printHello is undefined at the time it was invoked. It is not printed because it is function expression, and it is not hoisted. So if we put printHello("John") after it will be true.*

* 1. (2 points) The JavaScript expression,

1 || 0 && 3 && 0

Evaluates to the result, 1

True or False?

\_\_\_\_ **True** \_\_\_\_\_\_\_\_

Rationale (show your working/analysis/evaluation steps):

*The correct answer is True because to evaluate the expression 1 || 0 && 3 && 0*

*1 step. evaluate 0&&3 (due to the java script operator precedence rules) and 0&&3=0(due to the meaning of the logical AND operator) i.e for any two operands, exp1 and exp2, exp1 && exp2=exp2 if exp1 is truthy; otherwise the result is exp1.*

*And since the number 0 is falsy, therefore,0&&3=0.*

*2 step. 0&&0 the same as in 1st step will be applied so the answer will be 0,*

*3 step. 1||0 =1(due to the logical OR operator i.e for any two operands, exp1, exp2,*

*Exp1||exp2=exp1.if exp1 is truthy; otherwise the result is exp2 and since the number is truthy, therefore 1||0=1*

*1||0&&3&&0*

*=1||(0&&3)&&0*

*=1||(0&&0)*

*=1||0*

*=1*

* 1. (2 points) Consider the HTML and embedded CSS given in the figure 1 below.

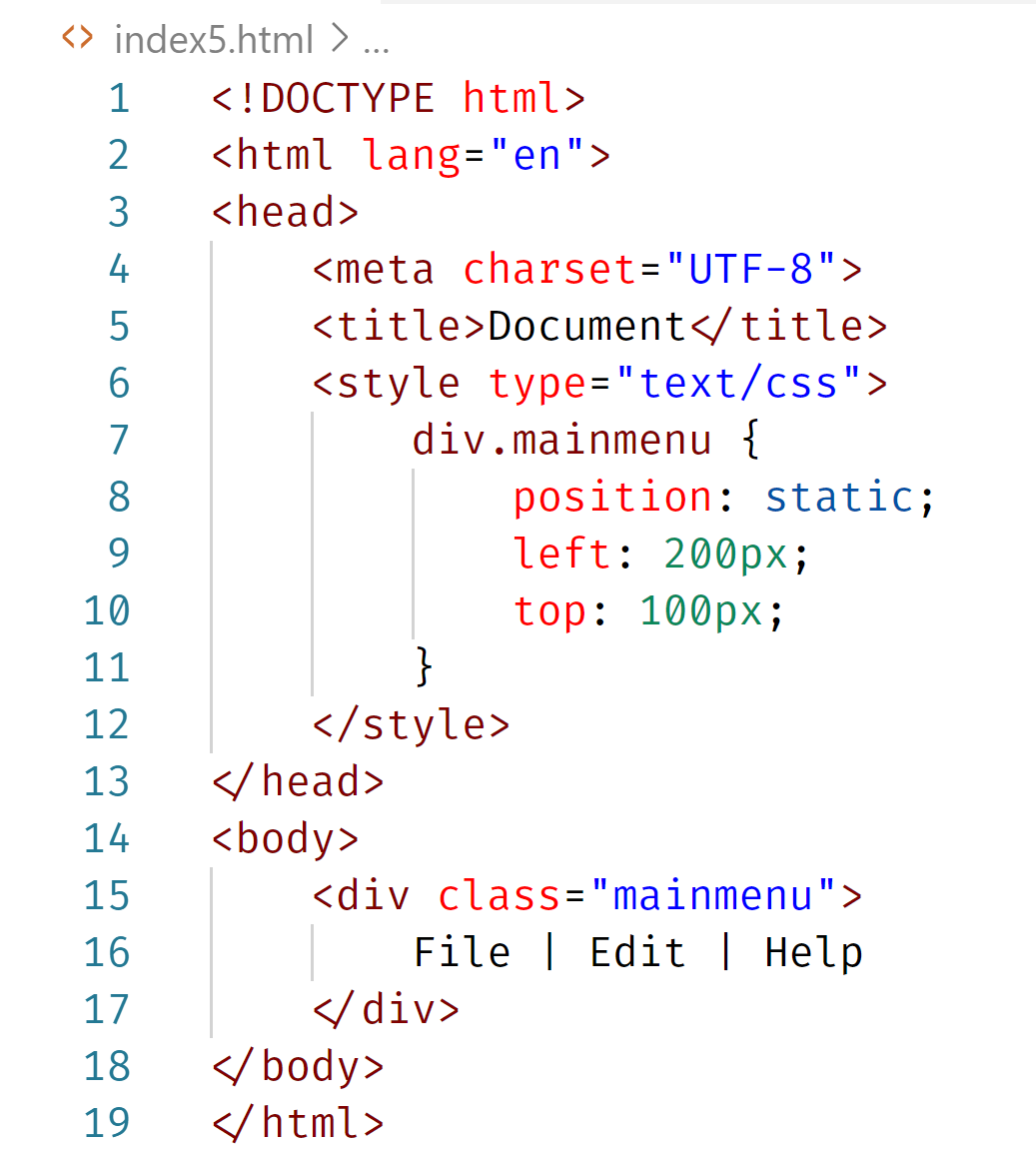


Figure1:

Based on the code, the <div> with class named, mainmenu, will be displayed on the webpage, positioned at 200pixels from the left margin and 100pixels from the top margin.

True or False?

\_\_\_\_**False**\_\_\_\_\_\_\_\_

**Rationale**:

*Static positioning displays elements the way they are rendered. Statically*

*positioned elements don't obey left, top, right and bottom rules and therefor there*

*will be no off-set.*

* 1. (2 points) Consider the browser screenshot marked as Figure 2 below, which shows a web (html) page displaying some content that includes a hyperlink to [Maharishi University of Management](http://www.mum.edu) website.

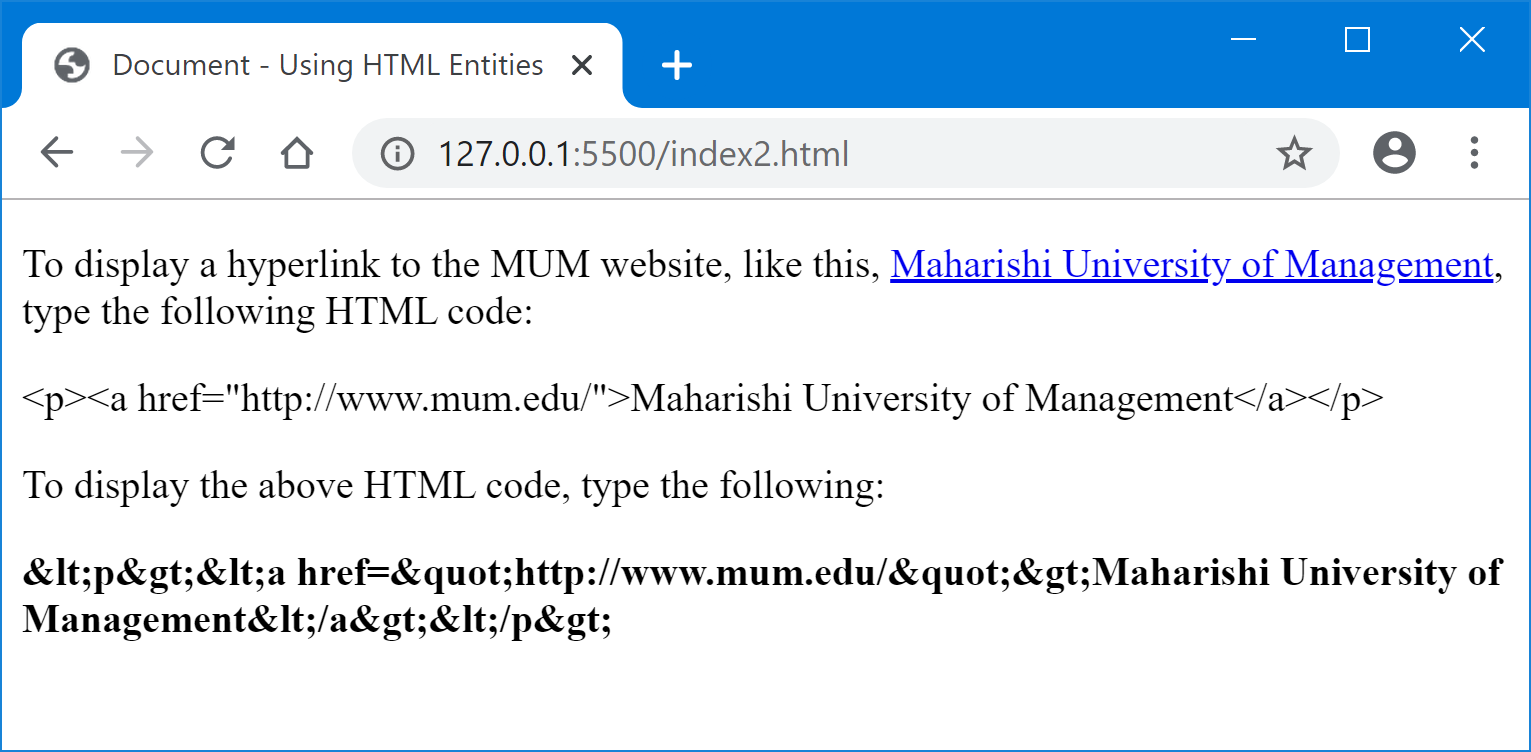


Figure 2:

Given below is the starter HTML code that will display the above web page, with its content as presented in the screenshot. Write the missing portion of the HTML code required to display the content.

Starter HTML Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Document - Using HTML Entities</title>

</head>

<body>

    <div>

        <p>

To display a hyperlink to the MUM website, like this, <a href="http://www.mum.edu/">Maharishi University of Management</a>, type the following HTML code:

</p>

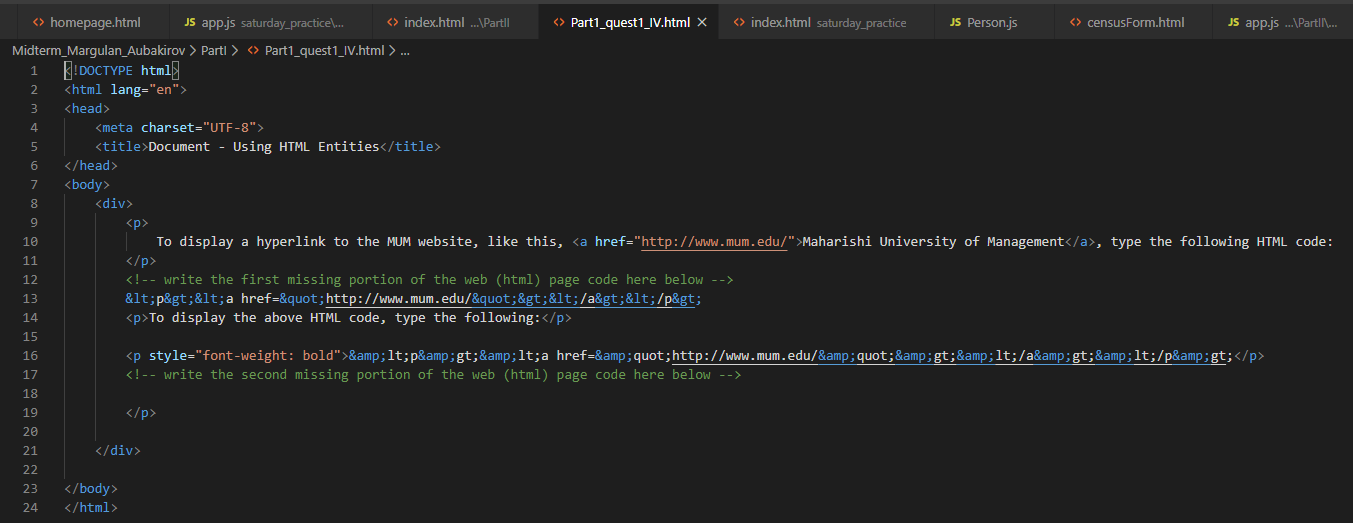
<!-- write the first missing portion of the web (html) page code here below -->

<!-- write the second missing portion of the web (html) page code here below -->

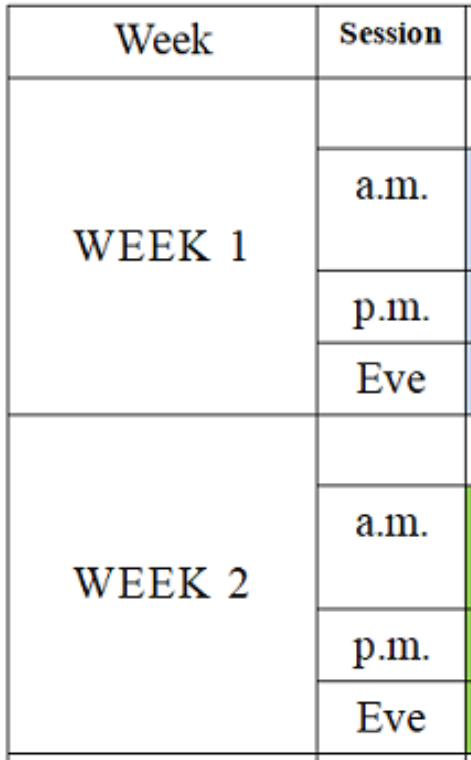
    </div>

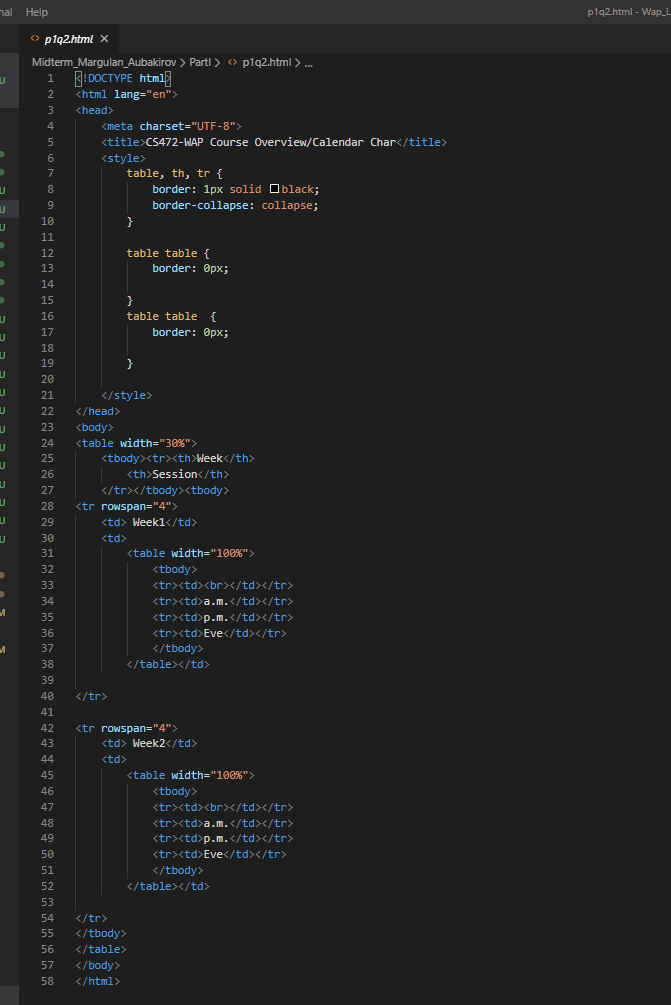
</body>

</html>



1. (3 points) Implement HTML code that displays the data for part of the CS472-WAP Course Overview/Calendar chart, as shown below (Hint: You may consider the data to be appropriate for tabular presentation. Also, assume that the cell heights on the 2nd column are the same/uniform):





(CS472 - WAP)

(May 2020)

Part 2 - Coding (30 points)

**Part II – JavaScript/Web Application Programming skills:** (30 points)

**Note:** *For the tasks in these questions, where applicable, you are expected to take screenshot(s) of your web UI(s), save into a .png or .jpg image file, placed inside a folder named, screenshots and include these in the MidtermExam.zip file, you submit or you may simply copy/paste your screenshots to the bottom of the associated question(s) right here in this document.*

1. (20 points) **Implementing a client-side JavaScript Web Application**

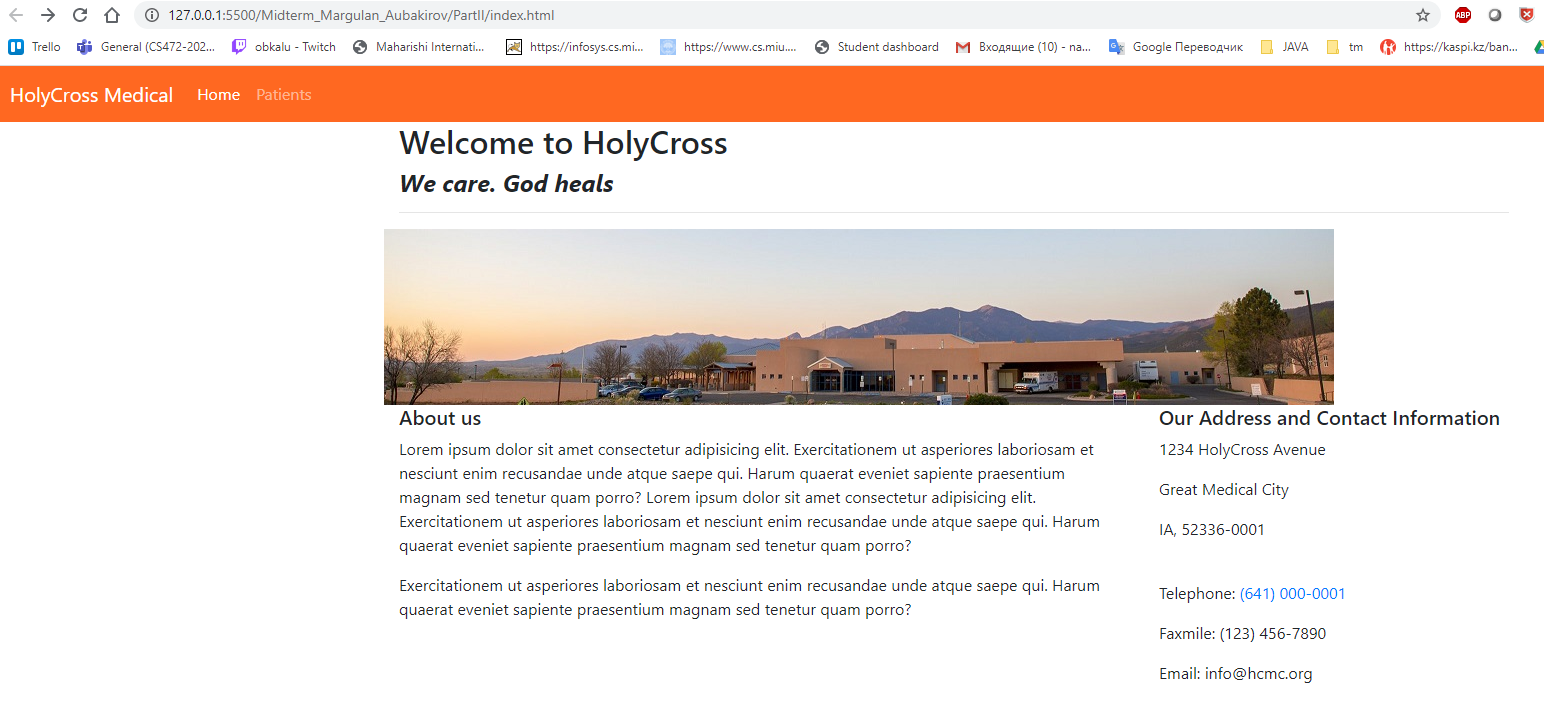
A popular city hospital, named HolyCross Medical Center (HCMC), has hired you to develop a web-based system for them, which they will be using to run part of their hospital operations. Specifically, the system will be used for registering their **Patient**s. Especially important to the Chief Medical Director of HCMC is, the data that provides information about their Elderly Patients and Out-patients.

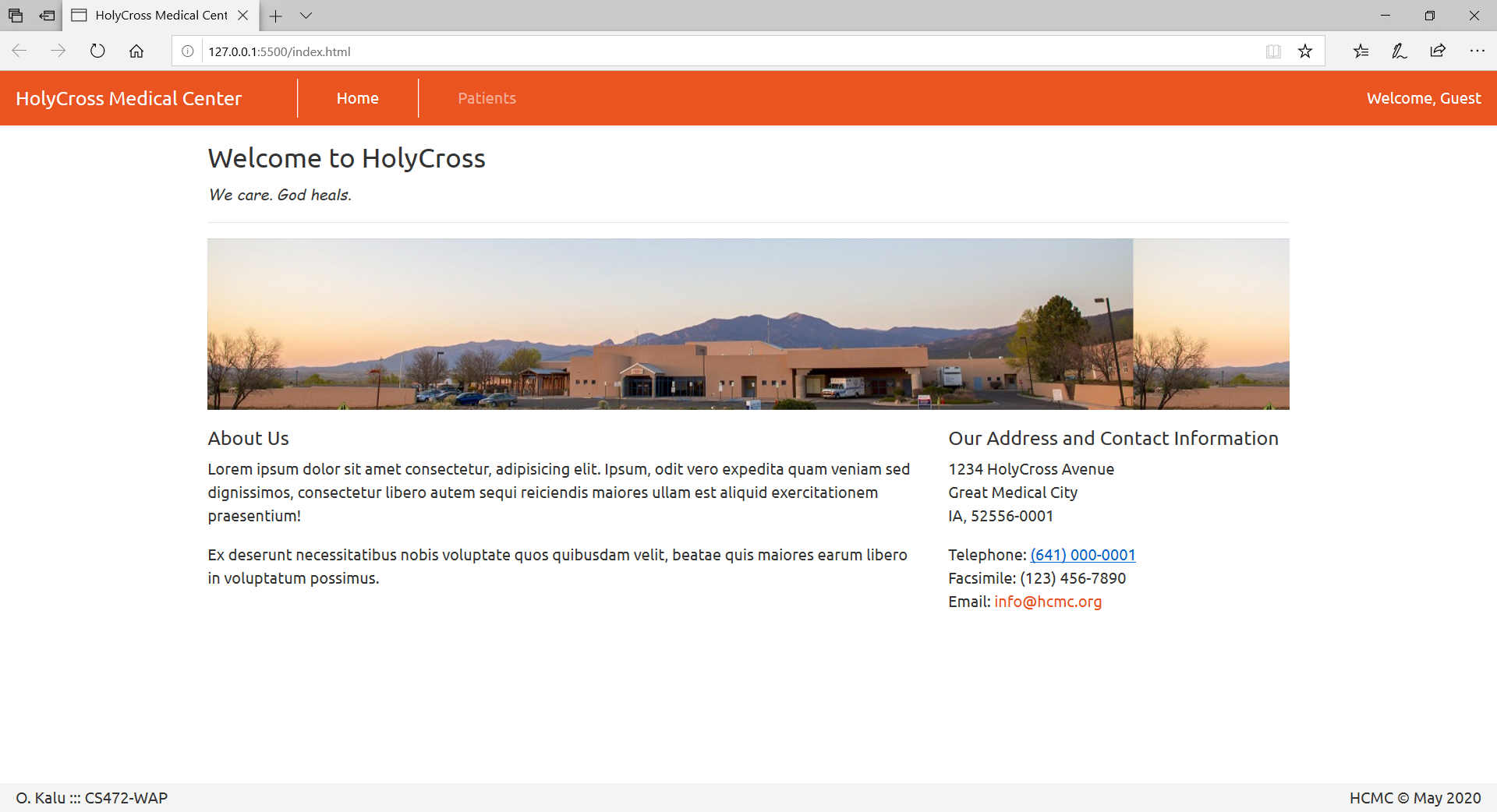
**An Elderly Patient is any patient who is of age, 65 or older.**

The website you are asked to build should have just a Homepage and a Patients Registration Form page (see the sample web UIs shown below).

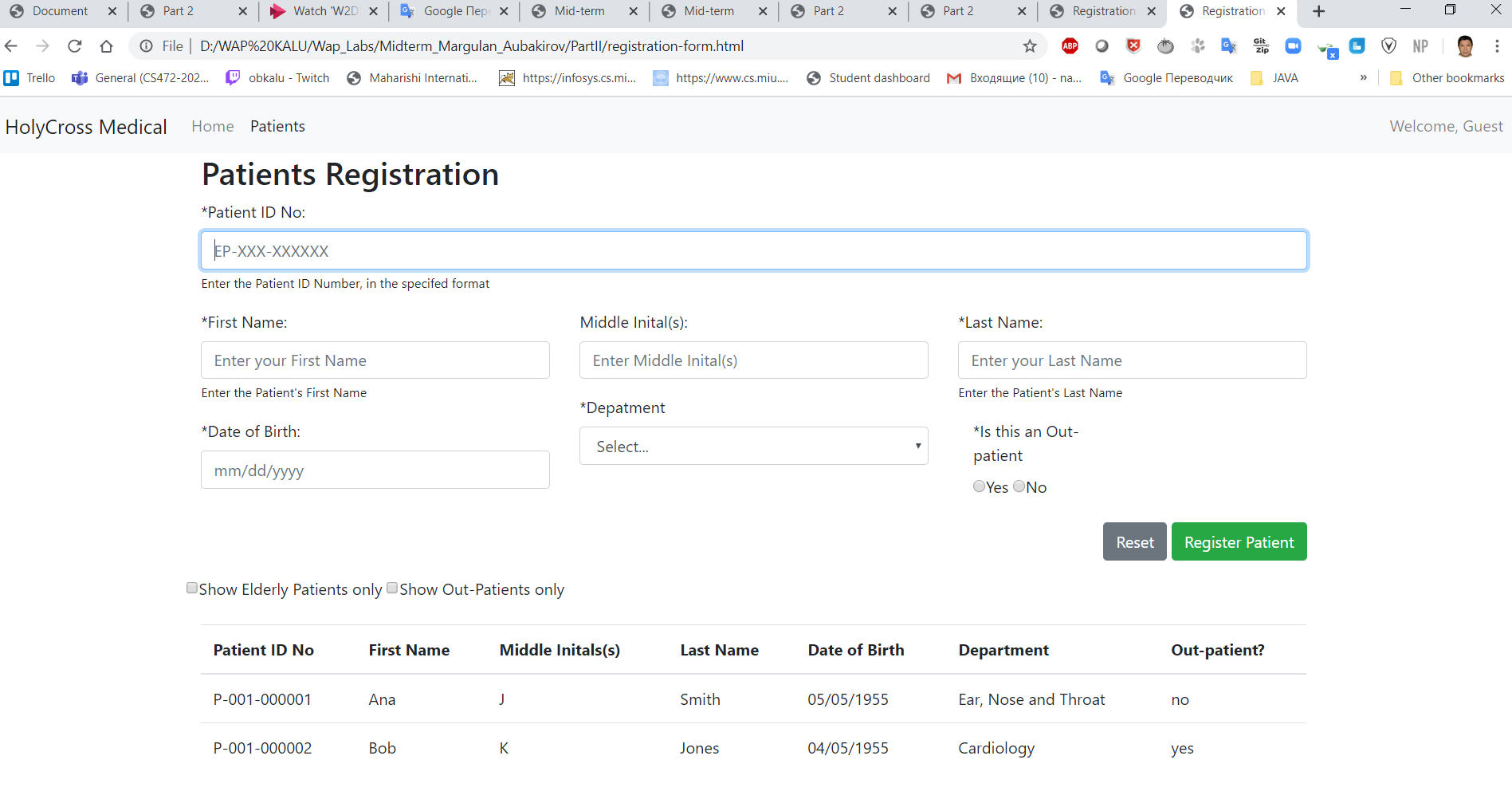
Using HTML, CSS and JavaScript, implement the website, as shown in the UI screenshots below, with the features and functionalities, specified/listed further below:

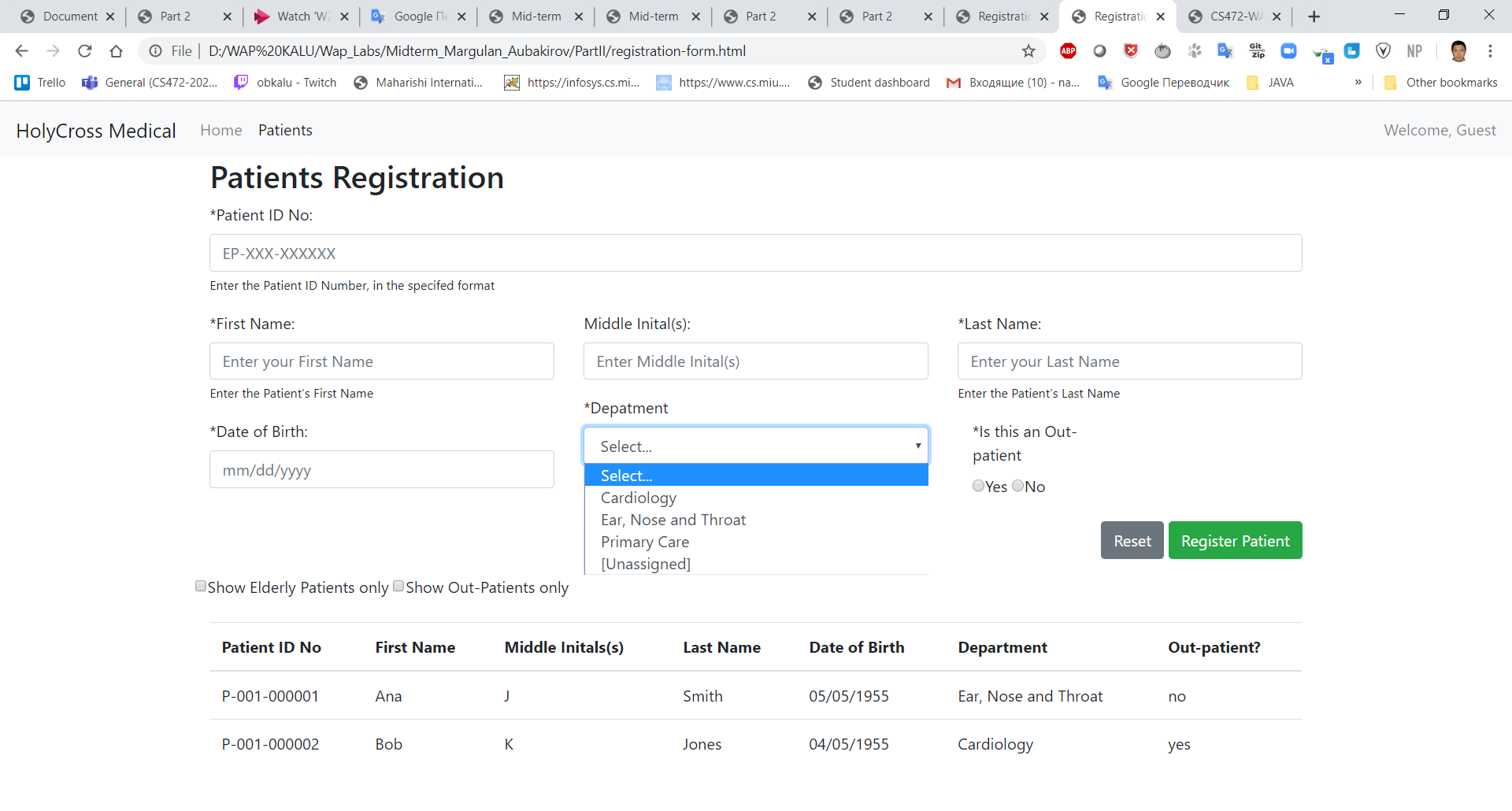
**Homepage**:

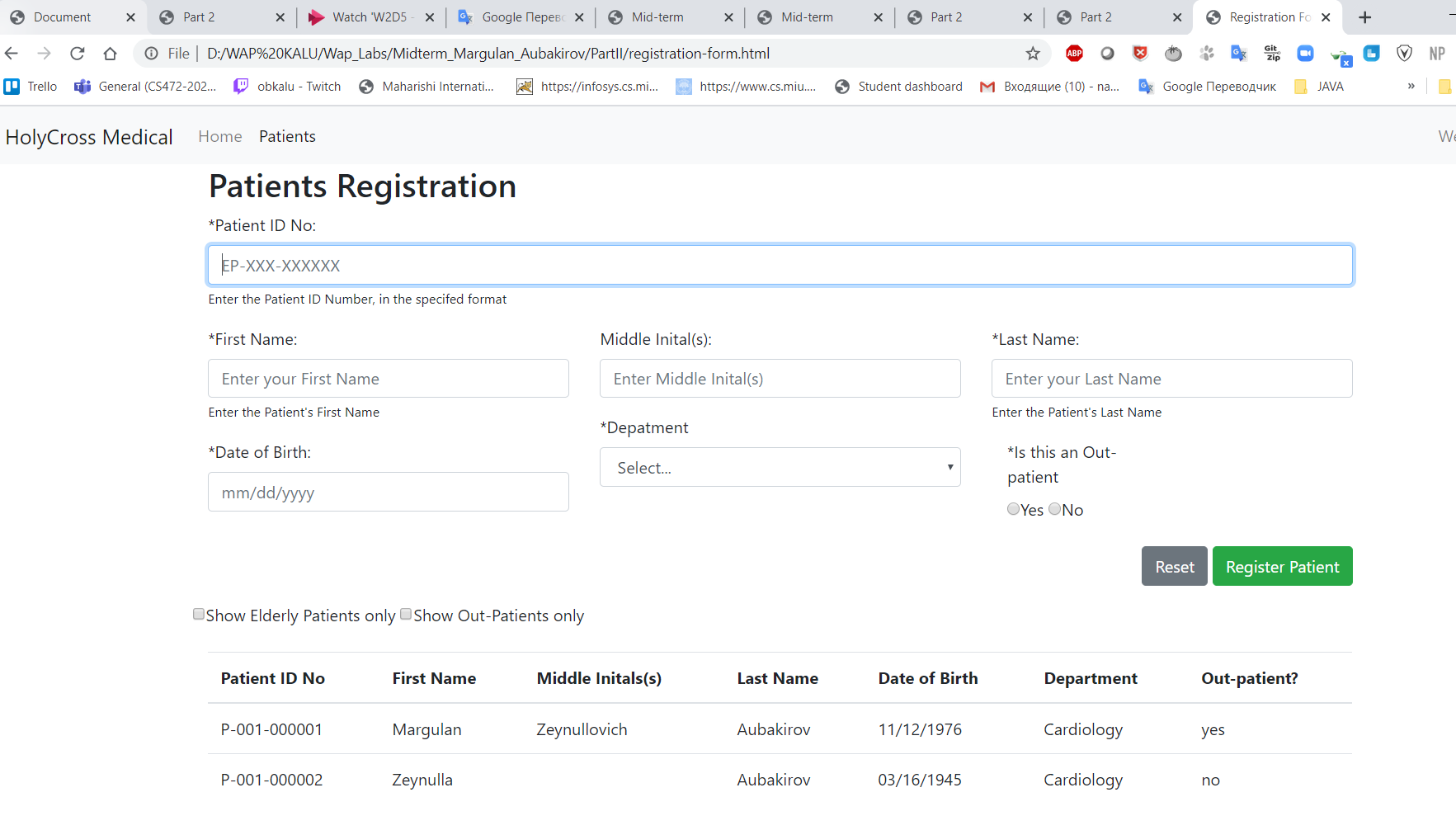


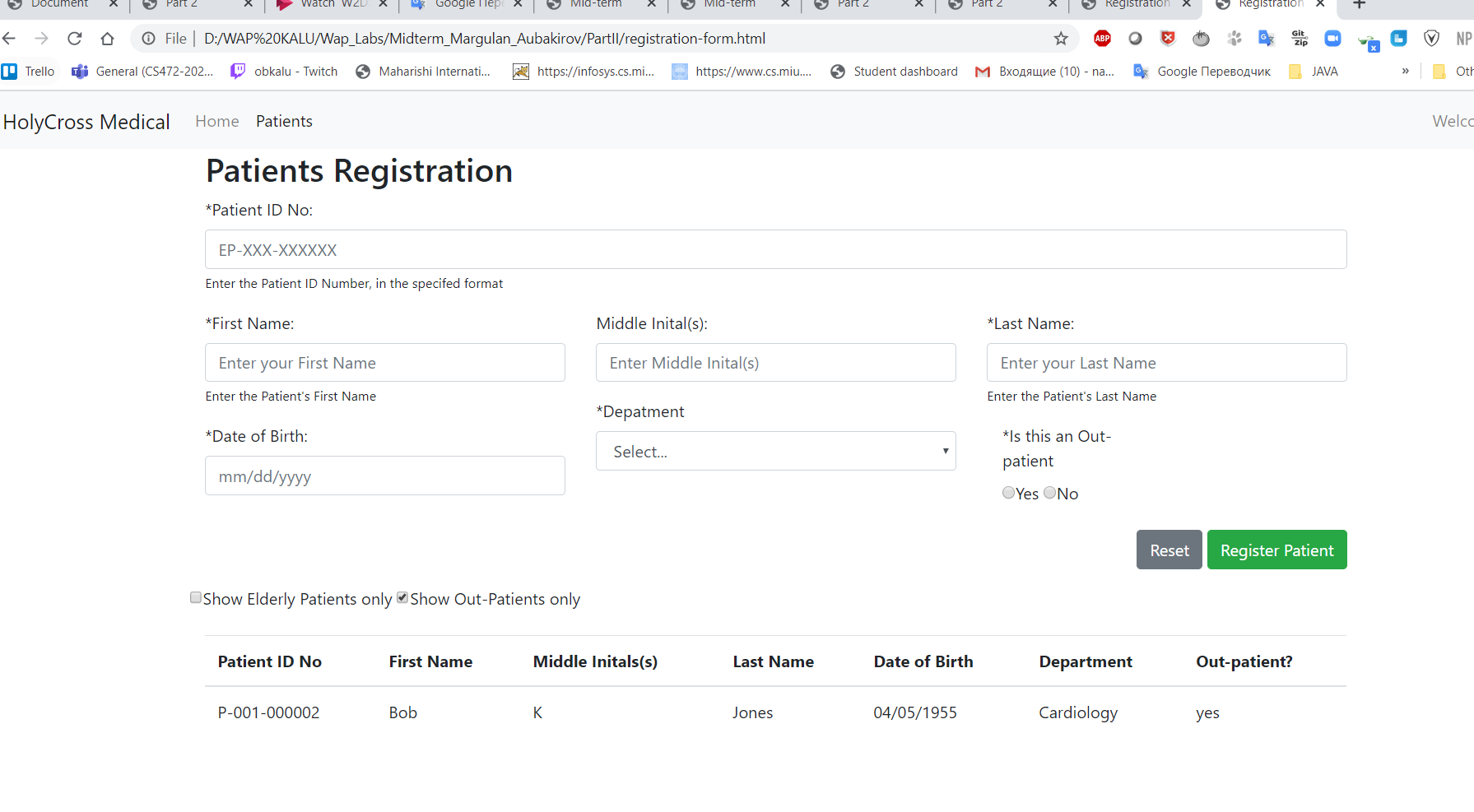


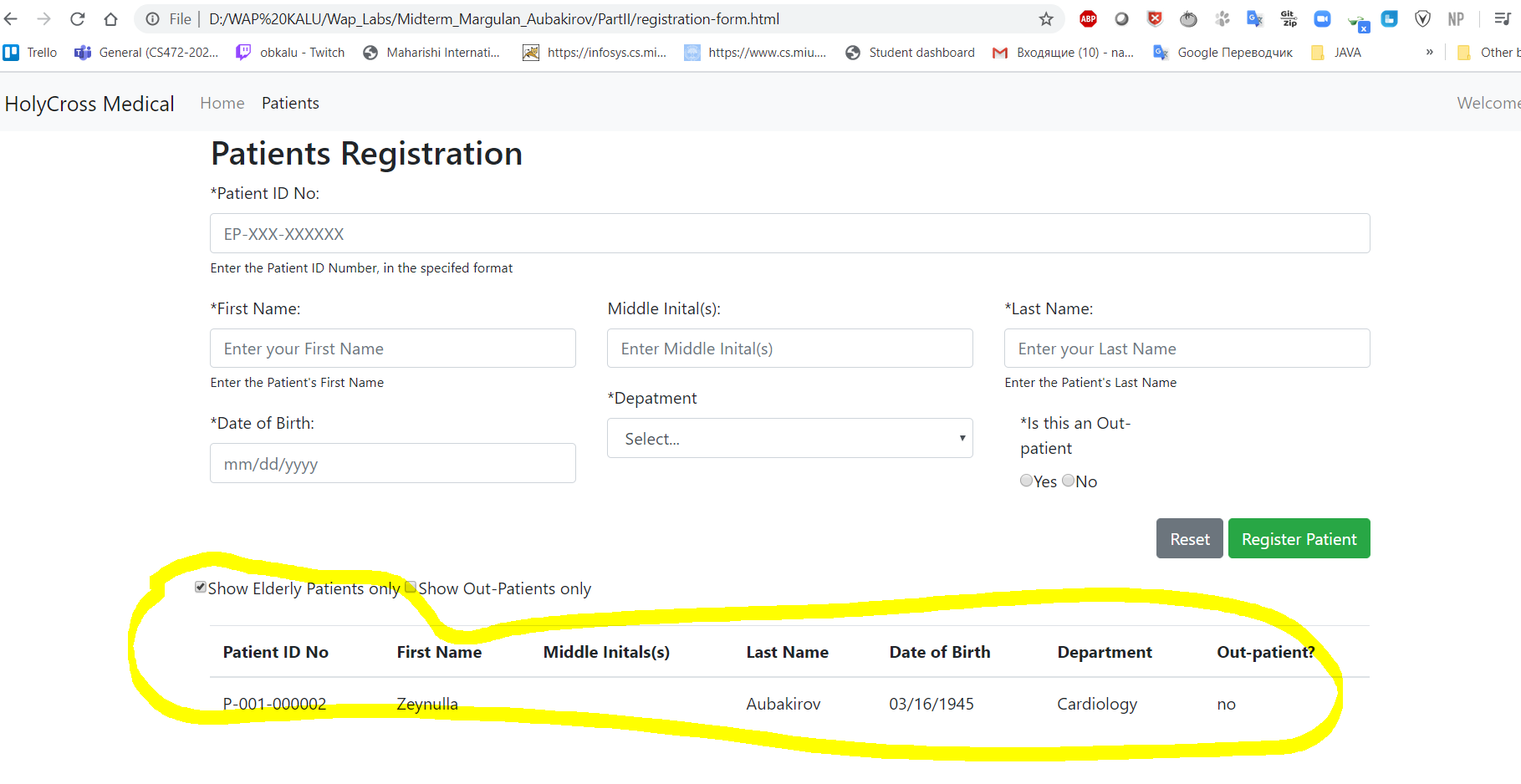
**Patients Registration Form page**:

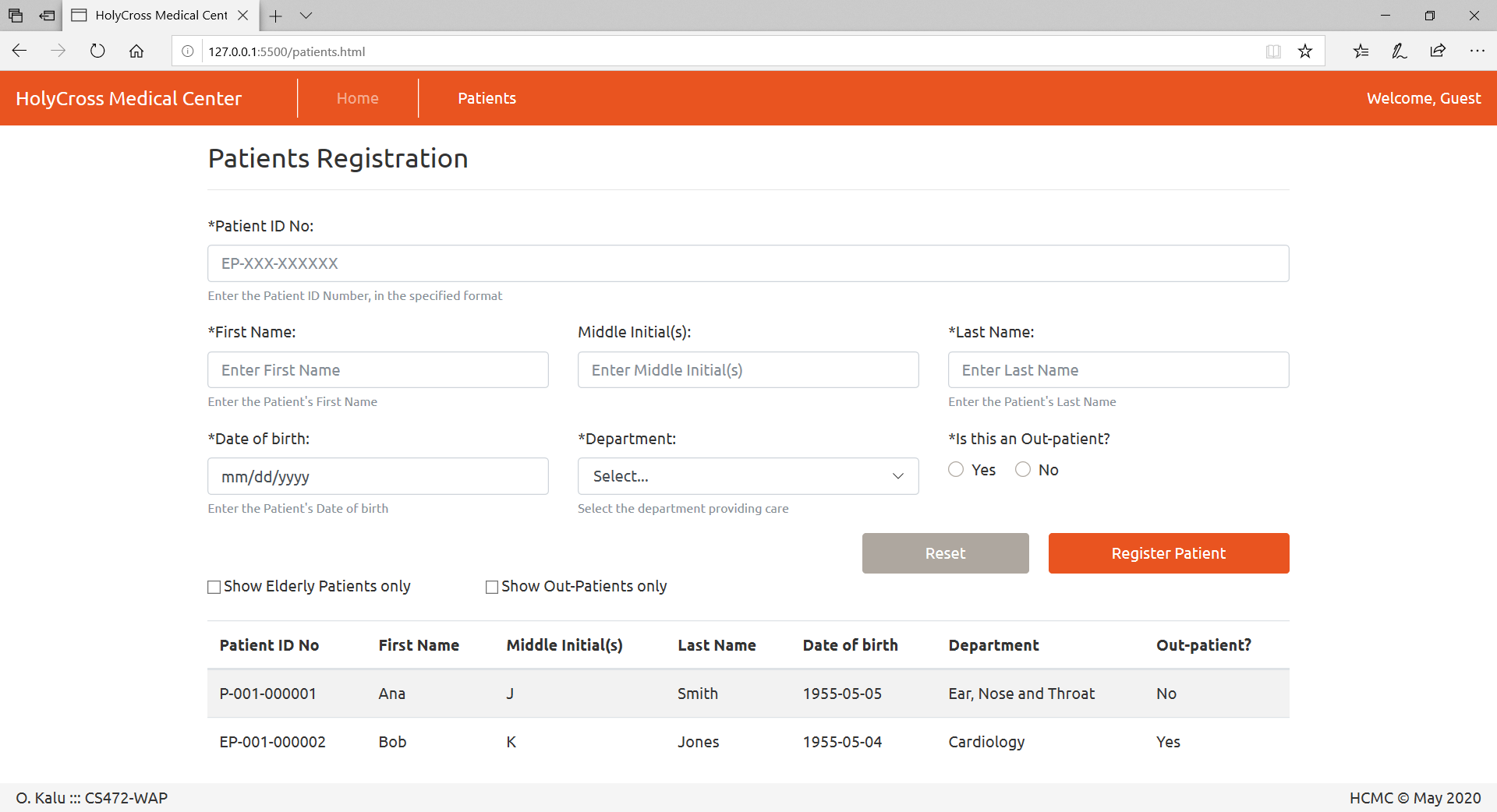




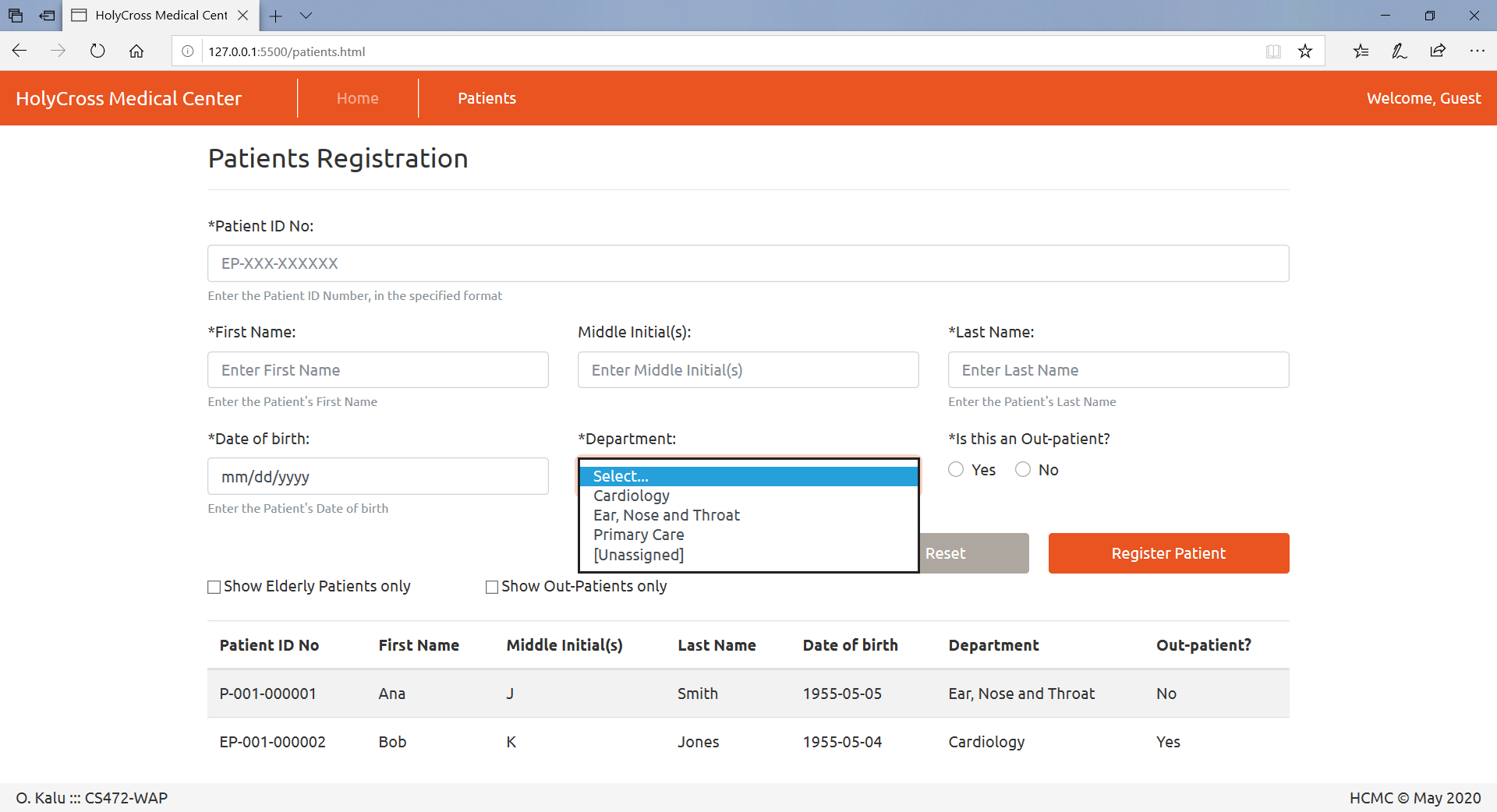








The data inside the Department drop-down list:



* 1. Code the User interface of the web application using standards-compliant, semantically correct HTML5 markup, including both the web pages and form fields as shown in the UI screenshots above.
  2. Apply styling using Bootstrap or your own custom CSS styling, to produce the same or similar look and layout. **Note:** Your UI does NOT necessarily have to be exactly the same as the sample shown above. But it should have all the necessary form data input fields, labels, images and buttons etc.
  3. Patient ID Number, First Name, Last Name, Date of birth, Department and Out-patient are all required data, to register a Patient into the system.
  4. Add data validation check using appropriate regular expression to ensure that each Patient ID Number entered must be in the specified format of, either P-XXX-XXXXXX or EP-XXX-XXXXXX, as shown on the Patient Registration form above. And where X is a numeric digit.
  5. Using JQuery and/or plain-vanilla JavaScript and the DOM API, implement code to register the 2 sample patients’ data listed below, which also appends/displays the data in a table at the bottom of the page, when the “Register Patient” button is clicked and the form is submitted, as shown in the screenshot above. The user should be able to register many patients’ data and see them listed on the webpage.

**Sample Patients’ Data for submission to test/demonstrate your web application**:

Patient 1:

Patient ID Number: P-001-000001,

First Name: Ana,

Middle Initial(s): J,

Last Name: Smith,

Date of birth: 5 May 1955

Department: Ear, Nose and Throat,

Is out-patient?: No

Patient 2:

Patient ID Number: P-001-000002,

First Name: Bob,

Middle Initial(s): K,

Last Name: Jones,

Date of birth: 4 May 1955

Department: Cardiology,

Is out-patient: Yes

* 1. Implement code for just ONE of the following two optional features:
     1. Implement code such that when the use selects/checks the checkbox labelled, “Show Elderly Patients only”, the data table will display only data for the Elderly Patients. And when the user deselects/unchecks the checkbox labelled, “Show Elderly Patients only”, the data table will re-display all the Patients data again. **Note**: The criterion for determining which Patient is considered an Elderly Patient is given above.

OR

* + 1. Implement code such that when the use selects/checks the checkbox labelled, “Show Out-Patients only”, the data table will display only data for the Out-Patients. And when the user deselects/unchecks the checkbox labelled, “Show Out-Patients only”, the data table will re-display all the Patients data again.

(**Please note:** Points will be awarded based on your adherence to Web programming recommended best practices such as use of standards-compliant, semantically-correct HTML5 markup, unobtrusive CSS, unobtrusive JavaScript etc).

1. (10 points) **Implementing** **JavaScript objects - composition and Inheritance**:

Define a base object named, Account, as JavaScript object literal, with the following three properties:

**accountNumber** - set its initial value as null

**balance** - set its initial value as 0.0

**accountType** - set its initial value as null

and include the following method:

**toString** - takes no parameter, and returns a string representation of the account data, in the following format – { AccountNumber: [accountNumber], balance: [balance], AccountType: [accountType] }

Using inheritance, create the following 2 derived objects named, **savingsAccount**, and **checkingAccount**, both of which inherit the properties and methods from the Account object, defined above.

For the savingsAccount object, assign the values, ‘10001’, 125000.00 and ‘Savings’ to the respective properties of accountNumber, balance, and accountType.

For the checkingAccount object, assign the values, ‘10002’, 75090.50 and ‘Checking’ to the respective properties of accountNumber, balance, and accountType.

Implement code to add a new method named, addInterest, to the savingsAccount object. The addInterest method should take a parameter named interestRate, which is a numeric percentage value and it computes and add the interest earned to the Savings account

balance.

Implement code to add a new method named, deductMaintenanceFee, to the checkingAccount object. The deductMaintenanceFee method should take a parameter named maintenanceFeeRate, which is a numeric percentage value and it computes and deducts the maintenance Fee from the Checking account balance.

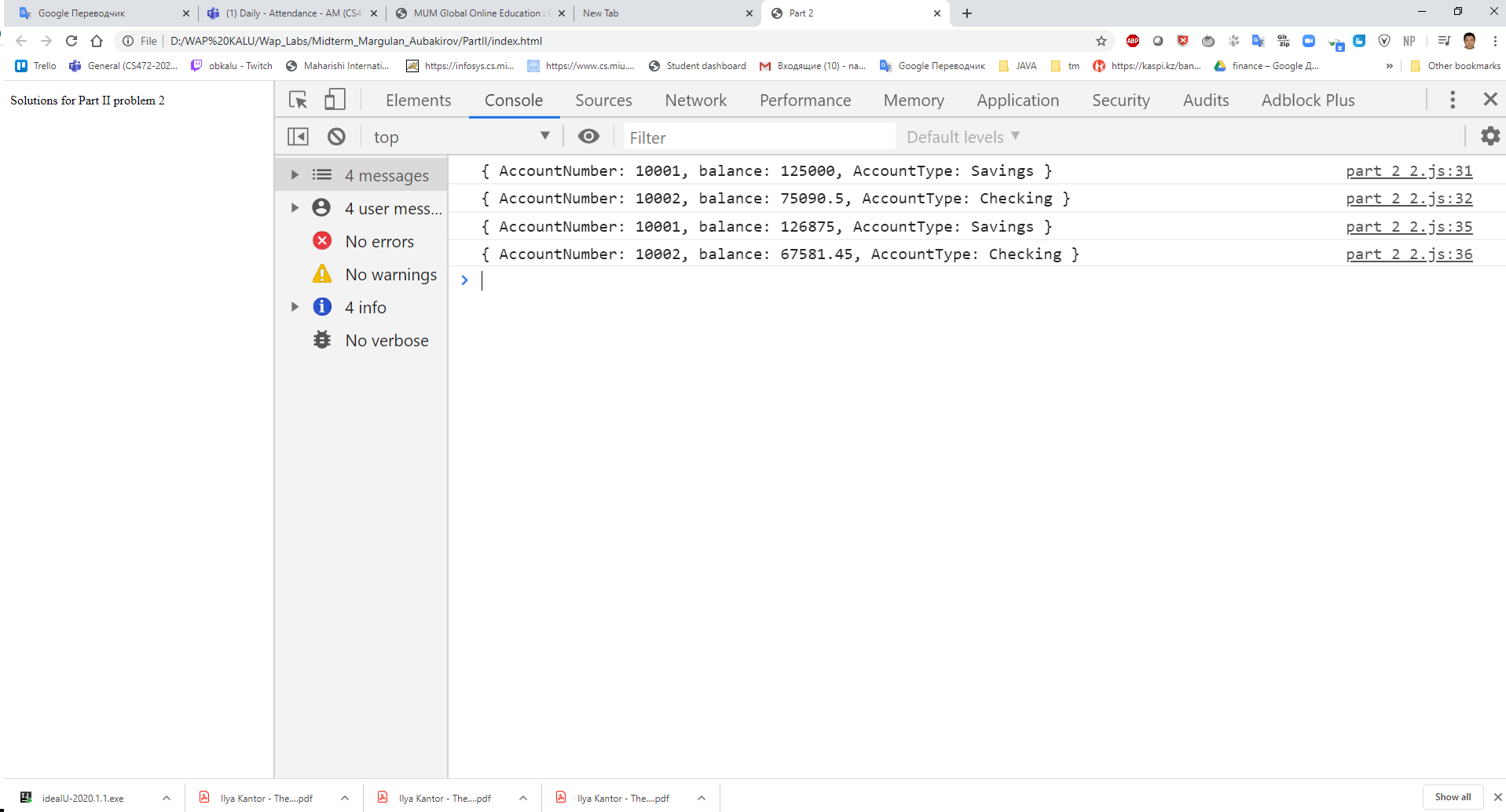
Implement code to invoke your savingsAccount.addInterest method, passing in 1.5% as the interestRate.

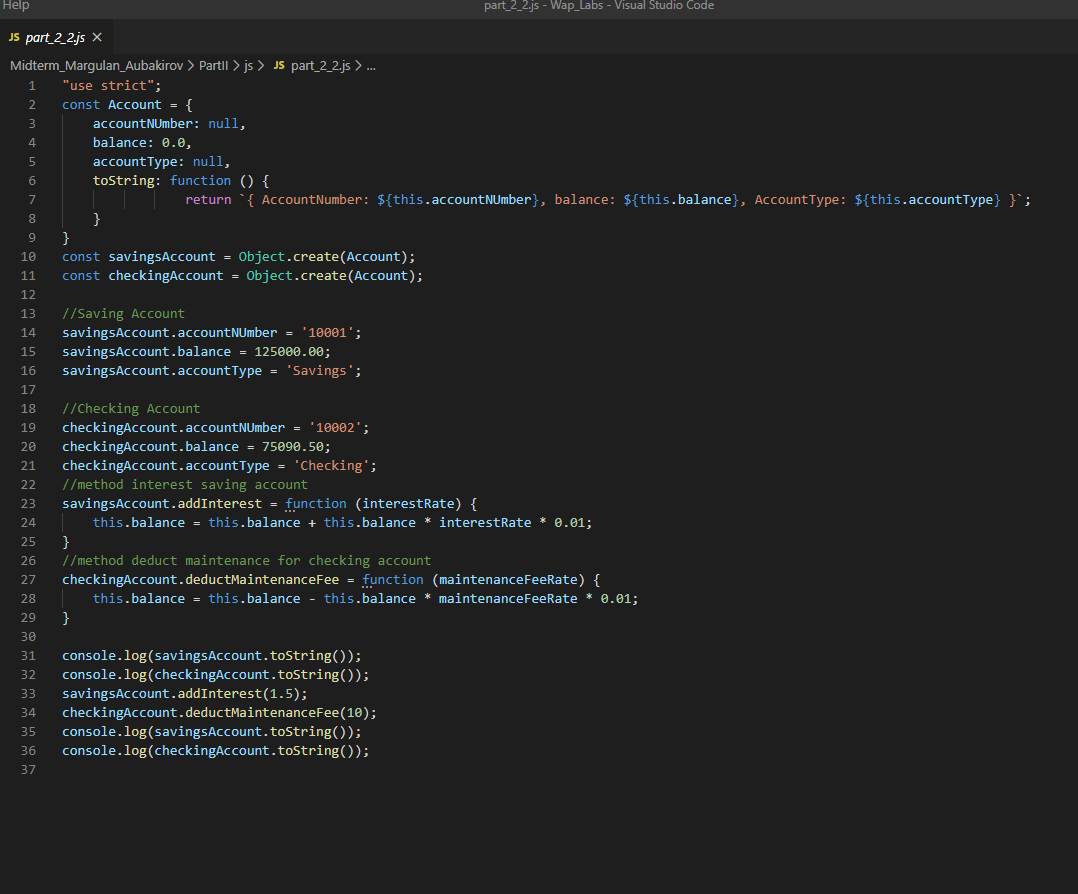
Implement code to invoke your checkingAccount.deductMaintenanceFee method, passing in 10% as the maintenanceFeeRate.

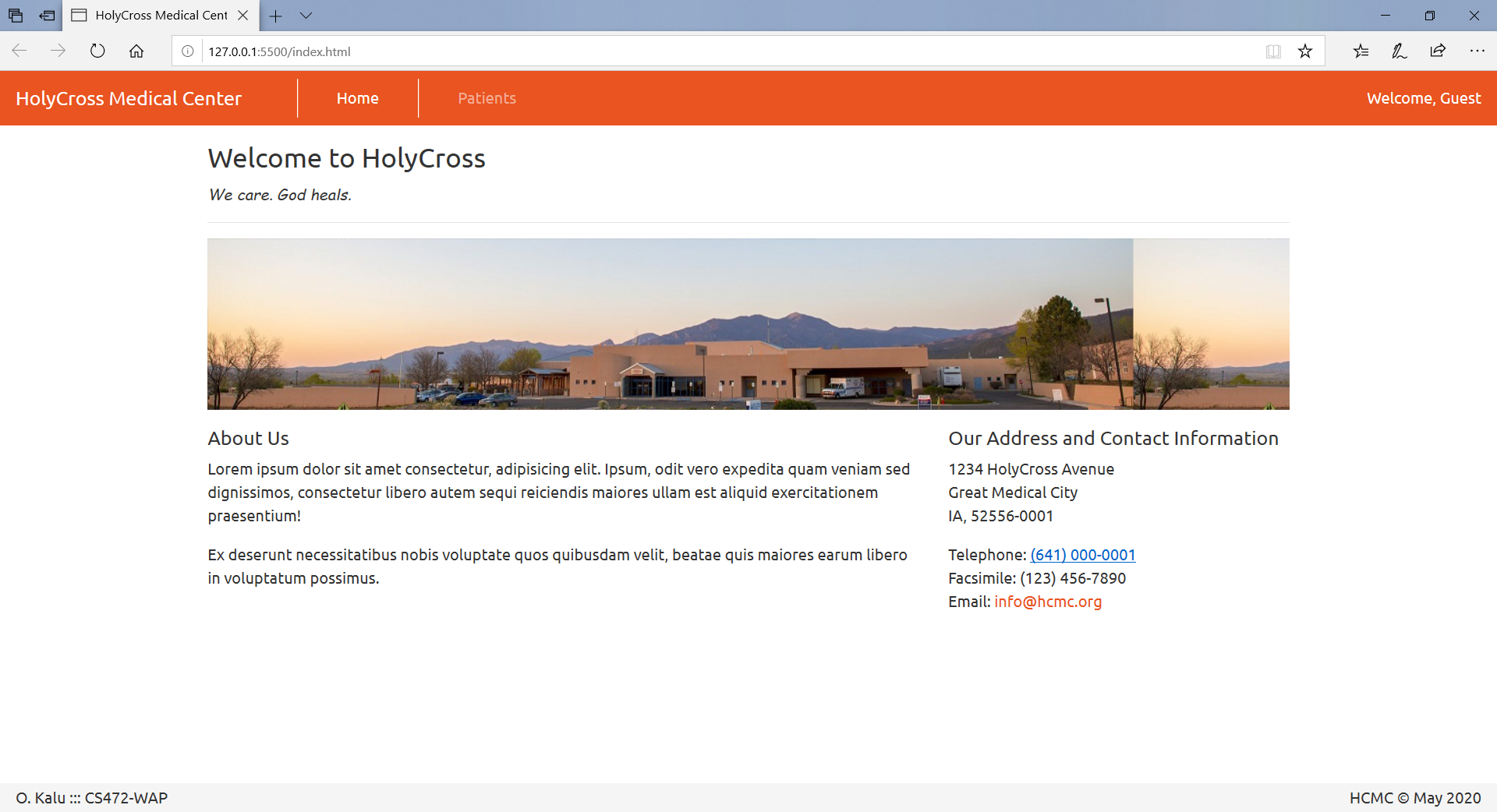
Implement code to invoke your savingsAccount.toString method and print-out its output to the console.

Implement code to invoke your checkingAccount.toString method and print-out its output to the console.

Open the Web browser console and take a screenshot of your results and save it to an image file named, P2Q2.png.







**//-- The End --//**