CSEE5590/CS490 APS
Programming for Web/Cloud/Mobile Applications

Lab Assignment 1
Course Registration App

Gulnoza Khakimova (8) and Jackie Batson (3)

Objective

The purpose of this project was to create a Web Application which helps UMKC students to enroll into classes by choosing different level of education (e.g. Undergraduate, Graduate), semester (e.g. Spring, Summer, Fall) and major (e.g. Computer Science, Biology, Math etc.). After selections are made, the list of available classes with information about it will be presented to a student. User will be able to to enroll in class and see number of available seats. Web page needs to be responsive and user friendly which should be achieved by using CSS for styling. HTML, CSS, Bootstrap, and JavaScript were used to implement application.

Features

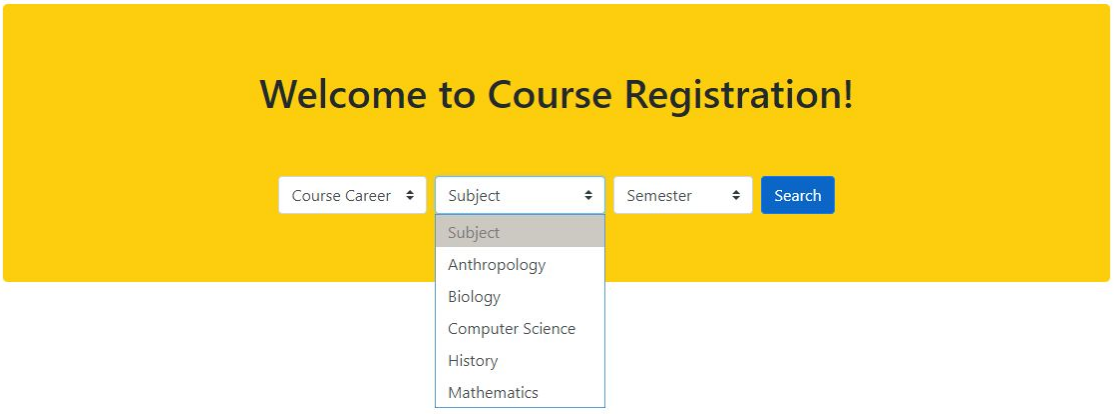
This application is for UMKC students who want to register for a course. A user will be able to see list of available courses during selected semester and major. Information about a class will be presented to user, like name of instructor who is offering a course, prerequisites, timing, description of the course, room where class will take place, number of credit hours, and reviews from students who previously took selected course. The table of courses is created dynamically with javascript based on the search terms the student submits. The student is able to click on name or image of the course to get more information about that course, which will also show number of available seats and number of already enrolled students. In order to register for a class, the user will have “Enroll” button available. After clicking on it, the student will see a pop up message if enrollment was completed successfully. The numbers of enrolled and available seats are also updated on the page without refreshing.

Steps

Course registration application takes you to selection page, where user can search for a class. Web page has following filtering options:

1. Course Career
    - undergraduate
    - graduate

2. Subject
    - Anthropology
    - Biology
    - Computer Science
    - History
    - Mathematics



3. Semester
    - Summer 2018
    - Fall 2018
    - Spring 2019

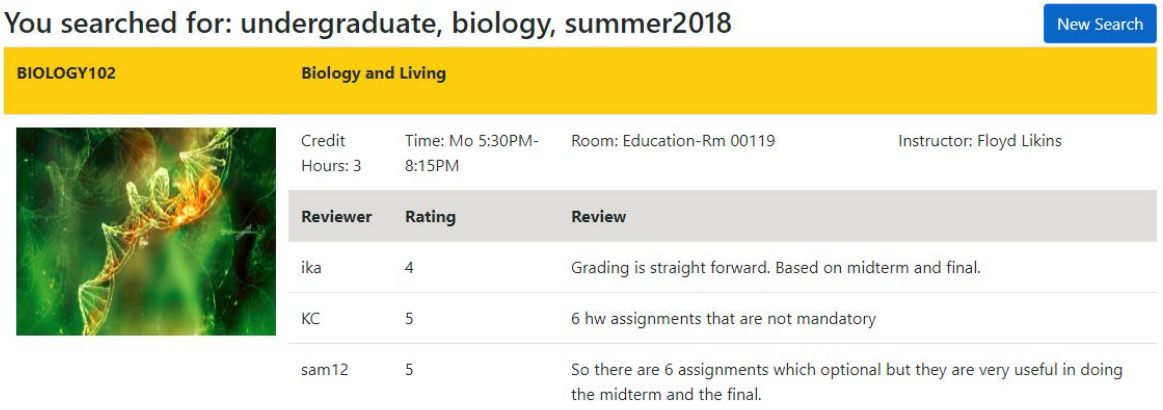
After required fields are selected and search button is clicked user will be presented with list of available course which will lay under requirements. Each course will have information relevant information such as:

- Name of the course
- Number of the course
- Credit Hours
- Timing

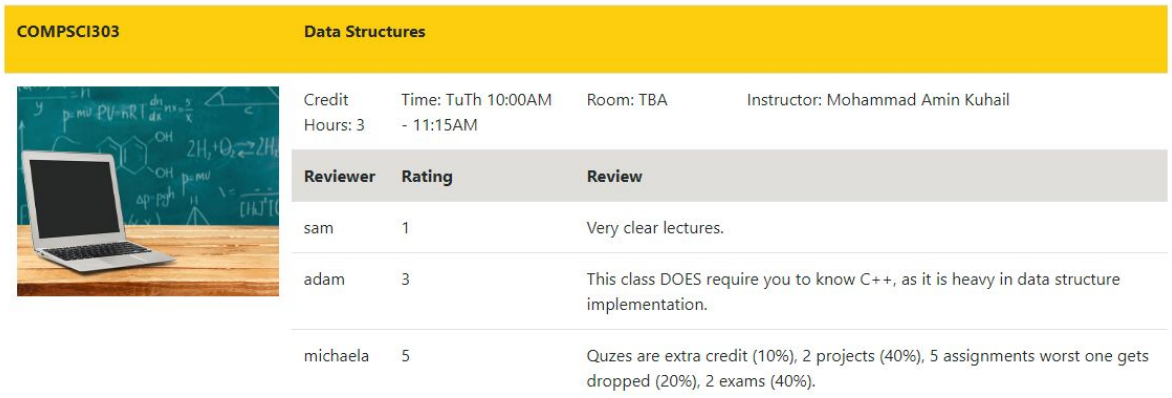
- Room where class will take place
- Instructor name
- Image related to offered course
- Reviews regarding a course from students

If the user wants to change the search criteria, our Web Application has a “New Search” button which will redirect a user to selection page.

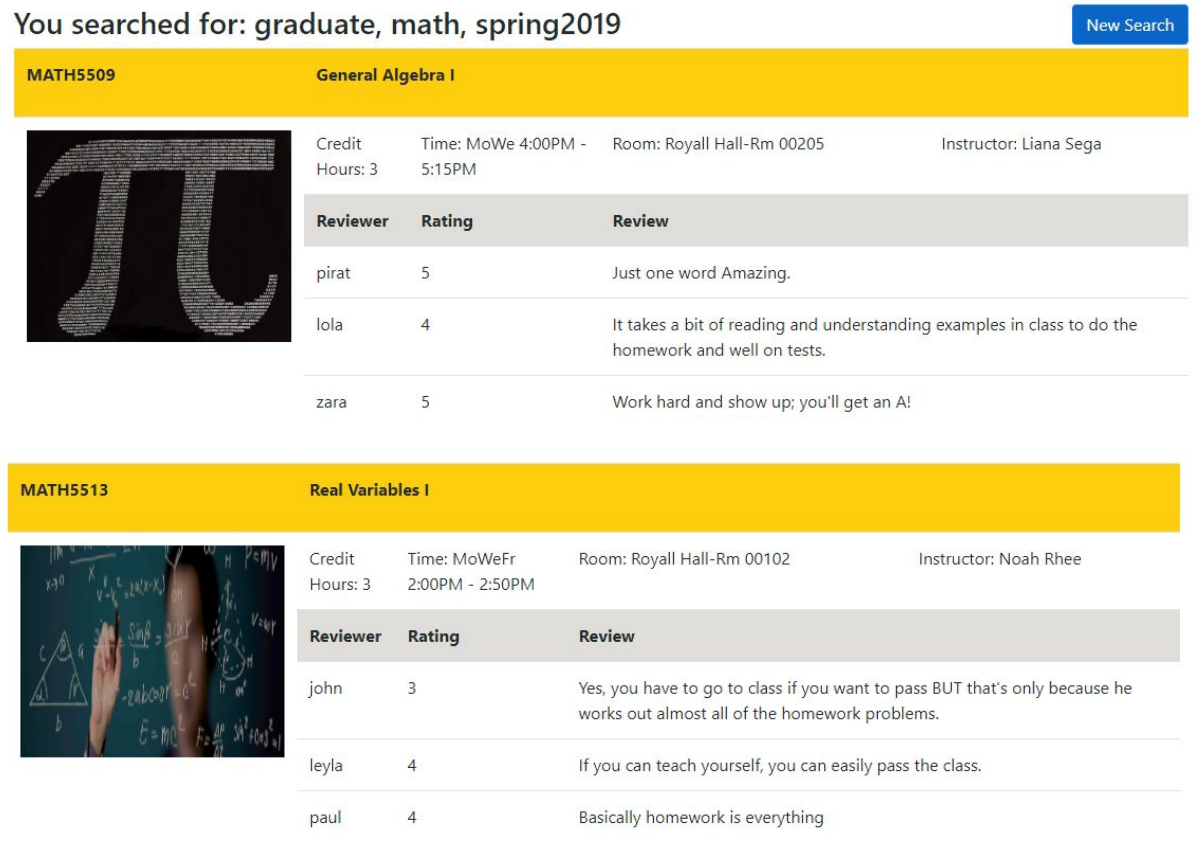
Output example when course career selected as: Undergraduate, Subject: Biology and Semester: 2018 Summer:



Output example when course career selected as: Undergraduate, Subject: Computer Science and Semester: 2018 Fall:

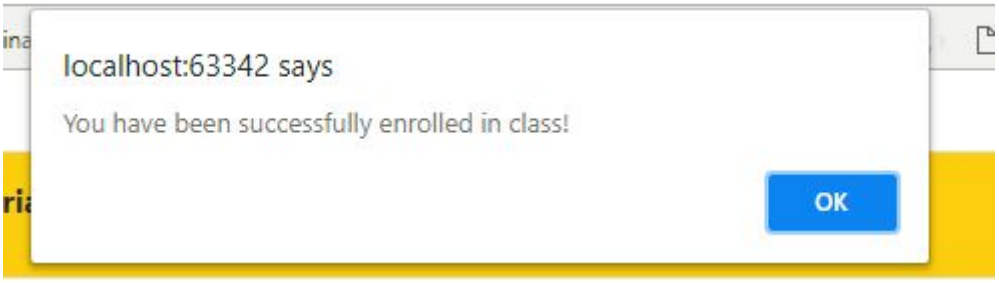


Output example when course career selected as: Graduate, Subject: Mathematics and Semester: 2019 Spring:

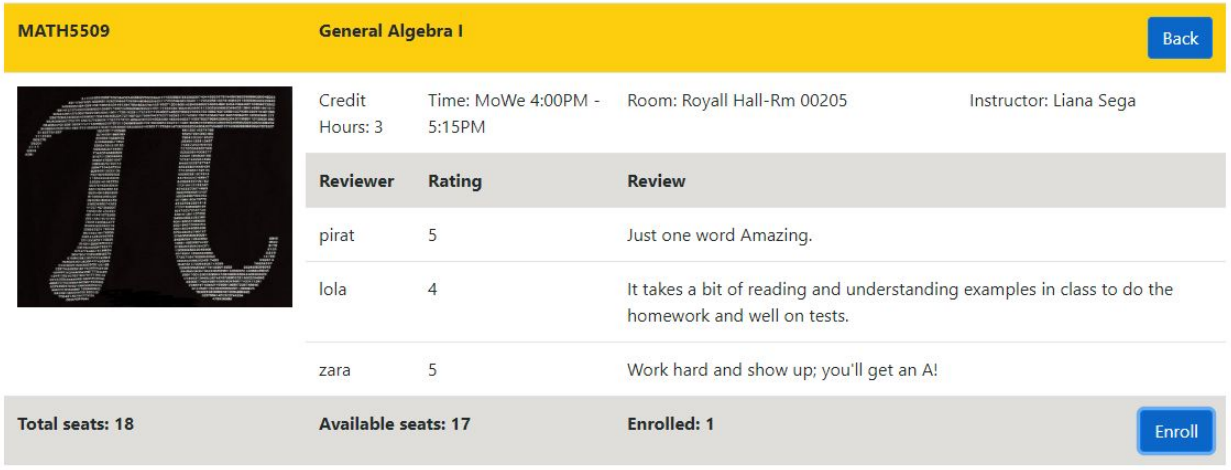


User can click either on Course number or an image in order to see more information regarding a course which will have information about total and available seats, also number of already enrolled students will be visible for the user. Enroll button is presented as well. If students decides to register for class “Enroll” button needs to be clicked which will output a message saying enrollment was successful and number of seats will be reflected dynamically.

Example of a message which shows us that registration for a course was successfully done:



After getting a message about successful enrollment number of available seats will be changed and enrolled number of students will be updated:



If course was selected by mistake or user wants to go back to list of courses - there is a “Back” button available to do that.

Code behind

Assignment implemented using WebStorm IDE version 2018.1.4. We utilized several techniques in order to meet requirements, such as HTML - to define a structure of Web

pages, CSS - to present a content of web pages and JavaScript programming language - to make page interactive. All pages are responsive - pages are automatically resized based on screen size. Bootstrap components were also used for styling, sizing, and interactive features. We used the Bootstrap buttons, selects, container sizing, and table format. The color scheme of our application was chosen to match UMKC colors.

HTML

There are three HTML pages:

1. Index.html
2. Enroll.html
3. Result.html

Index.html pages defines welcome page which give an option to make a selection for course search. Added UMKC logo using “<img>” tag. Defined classes for “ ” to make content of the page more user friendly. There are three selectors with multiple choice options in them. <Select> is bind to a CSS class. “Search” button calls “search()” method on click from JS.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Course Registration</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha384-WskhaSGFgHYWD
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
<div class="container">
    <img src="resources/umkc-with-name-two-color-digital-only.png" id="umkc-logo">
    <br><br>
    <div class="jumbotron">
        <h1>Welcome to Course Registration!</h1>
        <br><br>
        <div>
            <form action="" class="form-inline justify-content-center">
                <select name="" id="select-career" class="custom-select my-1 mr-sm-2">
                    <option selected disabled>Course Career</option>
                    <option value="undergraduate">Undergraduate</option>
                    <option value="graduate">Graduate</option>
                </select>
                <select name="" id="select-subject" class="custom-select my-1 mr-sm-2">
                    <option selected disabled>Subject</option>
                    <option value="anthropology">Anthropology</option>
                    <!--<option value="art">Art</option>-->
                    <option value="biology">Biology</option>
                    <!--<option value="chemistry">Chemistry</option>-->
                    <option value="cs">Computer Science</option>
                    <!--<option value="cj">Criminal Justice</option>-->
                    <!--<option value="dance">Dance</option>-->
                    <!--<option value="english">English</option>-->
                    <option value="history">History</option>
                    <option value="math">Mathematics</option>
```

```
                <select name="" id="select-semester" class="custom-select my-1 mr-sm-2">
                    <option selected disabled>Semester</option>
                    <option value="summer2018">2018 Summer</option>
                    <option value="fall2018">2018 Fall</option>
                    <option value="spring2019">2019 Spring</option>
                </select>
                <button type="button" class="btn btn-primary" onclick="search()">Search</button>
            </form>
        </div>
    </div>
</div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1P
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAd
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4
<script src="js/script.js"></script>
</body>
</html>
```

Result.html page shows list of classes which met search criteria. “ReadArray()” method is automatically called on page load which displays results of the page in <body>. At the top page shows search terms also there is a <button> “New Search” which takes a user to index.html page.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Search Results</title>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha384-WskhaSGFgHY
    <link rel="stylesheet" href="../css/style.css">

</head>
<body onload=ReadArray();>


<div class="container" id="table">
    <h3 id="searchTerms"></h3>
    <a href="../index.html" class="btn btn-primary">New Search</a>
</div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtT
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLa
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyY
<script src="../js/script.js"></script>
</body>
</html>
```

Enroll.html page displays more information about selected class. “Enroll()” method is called when <body> of the page is loaded. Also using defined classes in CSS to make page more user friendly.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Enrollment</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha384-WskhaSGFgHYW
    <link rel="stylesheet" href="../css/style.css">
</head>
<body onload="enroll()">
<div class="container" id="enrollTable">

</div>
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaU
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js" integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYm
<script src="../js/script.js"></script>
</body>
</html>
```

CSS

Style.css pages defines content of the page. Classes, IDs, header and body's design is specified in following code. Setting colors, margin values, background color, padding, text - align, border radius and etc. for different sections of the page.

```
.jumbotron {
    background-color: #F2CE0E;
    text-align: center;
}
#umkc-logo {
    background-color: #FFFFFF;
    border-radius: 10px;
    display: block;
    margin-left: auto;
    margin-right: auto;
    /*padding: 20px;*/
    /*width: 200px;*/
    /*height: 150px;*/
}
.btn-primary {
    background-color: #0066CC;
}

/*td {*/
    /*border: 1px solid #000;*/
    /*min-width:3em;*/
/*}*/
.bold {
    font-weight: bold;
    background-color: #DDDDDD;
}
.thead-dark {
    font-weight: bold;
    background-color: #F2CE0E;
}
```

JAVASCRIPT

Script.js class helps our application to be interactive. Defining Course function which has following parameters:

- Career
- Subject
- Number
- Name
- Semesters
- Instructor
- Hours
- Description
- Prerequisites
- Times
- Rooms
- Seats
- Reviews
- Logo

Created several classes and defined parameters by creating new “Course” object.

```
function Course(career, subject, number, name, semesters, instructor, hours, description, prerequisites, times, rooms, seats, reviews,logo)
    this.career = career;
    this.subject = subject;
    this.number = number;
    this.name = name;
    this.semesters = semesters;
    this.instructor = instructor;
    this.hours = hours;
    this.description = description;
    this.prerequisites = prerequisites;
    this.times = times;
    this.rooms = rooms;
    this.seats = seats;
    this.remaining = this.seats;
    this.reviews = reviews;
    this.enroll = function (semester) {
        for (i = 0; i < this.remaining.length; i++) {
            if (this.remaining[i].semester == semester) {
                this.remaining[i].seats--;
            }
        }
    };
    this.logo = logo;
}

var anthro103 = new Course("undergraduate", "anthropology", "anthro103", "Introduction to Cultural Anthropology",
    ["fall2018", "spring2019"], "Jefferey Bennett", 3,
    "An introduction to culture and the basic concepts of anthropology.  Topics include kinship, language, and cultural change.",
    "", [{semester:"fall2018", time:"Mo 5:30PM-8:15PM"}, {semester:"spring2019", time:"TuTh 8:30AM-9:45AM"}],
    [{semester:"fall2018", room:"Haag Hall-201"}, {semester:"spring2019", room:"Haag Hall-315"}],
    [{semester:"fall2018", seats:50}, {semester:"spring2019", seats:50}],
    [{reviewer:"anthrostudent", rating:4, review:"Class is easy. Don't buy the book"},
        {reviewer:"slacker", rating:2, review:"This class stinks, boring"},
        {reviewer:"anthro2", rating:5, review:"I loved this class, can't wait to be an anthropologist!"}],"anthro103.jpg");
```

“Search()” method performs a search of a course from a given array by getting carrer, subject and semester values which were picked by user. String results into “results” variable and saving it by invoking “sessionStorage.SetItem” method. Results are presented in results.html page which is done by redirecting a user from index.html to results.html page using “window.location” method.

```
function search() {
    career = document.getElementById("select-career").value;
    subject = document.getElementById("select-subject").value;
    semester = document.getElementById("select-semester").value;
    var results = getResults(career, subject, semester);

    sessionStorage.setItem("results", JSON.stringify(results));
    sessionStorage.setItem("semester", semester);
    sessionStorage.setItem("subject", subject);
    sessionStorage.setItem("career", career);

    window.location = './html/results.html';
}
```

“getResults()” function loops through array of courses and returns courses with matching criterias.

```
function getResults(career, subject, semester) {
    var results = [];
    for (var i = 0; i < courses.length; i++) {
        if (courses[i].career == career && courses[i].subject == subject) {
            for (var j = 0; j < courses[i].semesters.length; j++) {
                if (courses[i].semesters[j] == semester) {
                    results.push(courses[i]);
                }
            }
        }
    }
    return results;
}
```

“readArray()” function gets all values from array of courses and outputs in into a dynamic table view. Called when body of html page is loaded.

```
// Called when the results page loads.
// Reads the array returned by the previous function and dynamically creates a table to display the results
function ReadArray() {
    var results = JSON.parse(sessionStorage.getItem("results"));
    var semester = sessionStorage.getItem("semester");
    var div = document.getElementById("table");

    // display the searched terms at the top of the results page
    var head = document.getElementById("searchTerms");
    head.innerHTML = "You searched for: " + sessionStorage.getItem("career") + ", " + sessionStorage.getItem("subject") + ", " + semester;

    for (var i = 0; i < results.length; i++) {
        var table = document.createElement("table");
        table.className = "table";

        // Header row includes the course number and name.  Course number links to the Course enrollment detail page
        var header = table.createTHead();
        header.className = "thead-dark";
        var headRow = header.insertRow();
        var classNumber = headRow.insertCell(0);
        var result1 = results[i];
        classNumber.innerHTML = "<p id=\"num\" onclick=\"gotoenroll(this)\">" + results[i].number.toUpperCase() + "</p>";
        var className = headRow.insertCell(1);
        className.setAttribute('colspan', 4);
        className.innerHTML = results[i].name;

        // Table body
        var tableBody = document.createElement('tbody');
        table.appendChild(tableBody);

        // used to get the course number for looking up the image
        var value = results[i].number;

        // row1 includes the image, credit hours, time, room, and instructor
        var row1 = tableBody.insertRow();
        var logo = row1.insertCell(0);
        logo.setAttribute('rowspan', 5);
        logo.innerHTML = logo.innerHTML = "<img src=\"../resources/"+results[i].logo+"\" style=\"height: 200px;width: 250px\" onclick=\"gotoenroll("
        var creditHours = row1.insertCell(1);
        creditHours.innerHTML = "Credit Hours: " + results[i].hours;
        var time = row1.insertCell(2);
        var times = results[i].times;

        // must iterate through the times array to find the time for the selected semester
        for (var t = 0; t < times.length; t++) {
            if (times[t].semester == semester) {
                time.innerHTML = "Time: " + times[t].time;
```

“gotoenroll()” function is called when Number or Image of a course is selected. Saves course object using “sessionStrorage.setItem” and redirects a user to “enroll.html” page.

```
function gotoenroll(selectedclass) {
    var t = $(selectedclass).text();
    if($(selectedclass).text() == "")
     {
         t = selectedclass.number;
     }
    sessionStorage.setItem("selected",t);
    window.location = 'enroll.html';
}
```

“enroll()” function is responsible for outputting information about selected course in a table view and gives an ability to enroll for class. If “Enroll” button is clicked number of enrolled students and number of available seats are reflected dynamically. If registration was successful “alert” is displayed saying that user was enrolled for course.

```
function enroll() {
    var selectedclass = sessionStorage.getItem("selected");
    var results = [];
    var resultsJSON = JSON.parse(sessionStorage.getItem("results"));
    resultsJSON.forEach(function(result) {
        var course = Object.assign(new Course(), result);
        results.push(course);
    })
    var semester = sessionStorage.getItem("semester");
    var div = document.getElementById("enrollTable");

for (var i = 0; i < results.length; i++) {
    if(results[i].number.toUpperCase() == selectedclass.toUpperCase())
    {
        var table = document.createElement("table");
        table.className = "table";

        var header = table.createTHead();
        header.className = "thead-dark";
        var headRow = header.insertRow();
        var classNumber = headRow.insertCell(0);
        var result1 = results[i];
        classNumber.innerHTML = results[i].number.toUpperCase();
        var className = headRow.insertCell(1);
        className.setAttribute('colspan', 3);
        className.innerHTML = results[i].name;
        // var enroll = headRow.insertCell(2);
        // var button = document.createElement('button');
        // button.className = 'btn btn-primary';
        // button.id = results[i].number;
        //
        // button.innerHTML = "Enroll";
        // button.onclick = function () { enrollButtonClick() };
        // enroll.appendChild(button);
        var back = headRow.insertCell(2);
        var backButton = document.createElement('button');
        backButton.className = 'btn btn-primary'
        backButton.innerHTML = "Back";
        backButton.onclick = function () { window.history.back(); }
        back.appendChild(backButton);
```

```
var tableBody = document.createElement('tbody');
table.appendChild(tableBody);
var row1 = tableBody.insertRow();
var logo = row1.insertCell(0);
logo.setAttribute('rowspan', 5);
logo.innerHTML =  "<img src=\"../resources/"+results[i].logo+"\" style=\"height: 200px;width: 250px\">";
var creditHours = row1.insertCell(1);
creditHours.innerHTML = "Credit Hours: " + results[i].hours;
var time = row1.insertCell(2);
var times = results[i].times;
for (var t = 0; t < times.length; t++) {
    if (times[t].semester == semester) {
        time.innerHTML = "Time: " + times[t].time;
    }
}
var room = row1.insertCell(3);
var rooms = results[i].rooms;
for (var r = 0; r < rooms.length; r++) {
    if (rooms[r].semester == semester) {
        room.innerHTML = "Room: " + rooms[r].room;
    }
}
var instructor = row1.insertCell(4);
instructor.innerHTML = "Instructor: " + results[i].instructor;

var row2 = tableBody.insertRow();
row2.className = "bold";
var headReviewer = row2.insertCell(0);
headReviewer.innerHTML = "Reviewer";
var headRating = row2.insertCell(1);
headRating.innerHTML = "Rating";
var headReview = row2.insertCell(2);
headReview.setAttribute('colspan', 3);
headReview.innerHTML = "Review";

var reviews = results[i].reviews;
for (var j = 0; j < reviews.length; j++) {
    var row = tableBody.insertRow();
    var reviewer = row.insertCell(0);
    reviewer.innerHTML = reviews[j].reviewer;
    var rating = row.insertCell(1);
    rating.innerHTML = reviews[j].rating;
    var review = row.insertCell(2);
    review.setAttribute('colspan', 3);
    review.innerHTML = reviews[j].review;
}
```

```
    var seatsRow = tableBody.insertRow();
    seatsRow.className= "bold";
    var seatsTotal = seatsRow.insertCell(0);
    seatsTotal.innerHTML = "Total seats: "+ totalSeats;
    var seatsAvailable = seatsRow.insertCell(1);
    seatsAvailable.id = "available";
    seatsAvailable.setAttribute('colspan', 2);
    seatsAvailable.innerHTML =  "Available seats: "+availableSeats;
    var enrolledSeats = seatsRow.insertCell(2);
    enrolledSeats.id = "enrolled";
    // enrolledSeats.setAttribute('colspan', 2);
    enrolledSeats.innerHTML = "Enrolled:";

    var enroll = seatsRow.insertCell(3);
    var button = document.createElement('button');
    button.className = 'btn btn-primary';
    button.id = results[i].number;

    button.innerHTML = "Enroll";
    //button.onclick = function () { enrollButtonClick(results[i]); };
    courseToEnroll = results[i];
    button.addEventListener("click", function () {enrollButtonClick()});
    enroll.appendChild(button);


    var emptyRow = tableBody.insertRow();
    emptyRow.innerHTML += "<br>";

    div.appendChild(table);
}
}
```

Configuration

1. IDE: WebStorm version 2018.1.4.
2. HTML 5
3. CSS 3
4. JavaScript
5. Bootstrap 4.1.1
6. JQuery 3.1.1

Limitation

Since we did not have access to Course database, classes were hard-coded with all parameters listed and application does not insert a record into database of university for enrolling a student. It does not have login page, so we are not differentiating users and not using security checks - not performing validation on username and password. Code is also

a little verbose; we think AngularJS might have simplified some of the methods we used. We would refactor the code in the future to use AngularJS now that we have learned it.

References

1. https://www.w3schools.com
2. https://umkc.umsystem.edu
3. http://www.ratemyprofessors.com
4. In class lectures
5. in Class Use cases and ICPs