# Assignment

There are 36 students in a web technology class. In order to organize the project demonstrations, the students will demonstrate their projects at six different one-hour long time slots. Up to 6 students can give their demonstrations in a given time slot. The instructor decides to have a registration webpage to allow students to sign up for one of the time slots. A student visiting the page should be able to submit his/her *ID*, *first name*, *last name*, *project title*, *email address*, *phone number*, and *book a seat in one of the available time slots*. A student is uniquely identified by his/her ID.

The submitted data should be stored in a MySQL database which is maintained on a server. The webpage and the server should interact with each other at every step of the registration process. The page should show how many free seats are available in each time slot, announcing and blocking all fully booked time slots. After a student makes a data submission, it should check whether the student has been already registered. If not, the data is stored on the server and the student is notified about her registration. Otherwise, if already registered, the student should be prompted to ensure that they want to change the registration to the new section (and removed from the current one they are registered for). For example, the time slots may look like the following list:

1. 4/19/2070, 6:00 PM – 7:00 PM, 6 seats remaining
2. 4/19/2070, 7:00 PM – 8:00 PM, 5 seats remaining
3. 4/19/2070, 8:00 PM – 9:00 PM, 3 seats remaining
4. 4/19/2070, 6:00 PM – 7:00 PM, 2 seats remaining
5. 4/19/2070, 7:00 PM – 8:00 PM, 4 seats remaining
6. 4/19/2070, 8:00 PM – 9:00 PM, 0 seats remaining

In addition, you need to write a separate webpage that will display the list of students (including their IDs, names, project titles, email addresses, phone numbers, and time slots) who are registered, after querying the database.

# Requirements

1. For your assignment, you should use HTML, CSS, ([*optionally*] JavaScript), MySQL, and PHP (unless you prefer to use something else like node.js. ***If you do use something besides PHP, such as node.js, indicate setup and execution instructions in a readme file in the submission deliverable***).
2. The database must fully implement the registration process, maintaining the submitted information. The server and page behavior must meet the requirements listed above.
3. The specific look and feel of the pages as well as the database implementation is left intentionally vague, allowing considerable design freedom on your part. However, the page should have a nice look and the code should satisfy common standards.
4. User inputs should be validated at the server side (e.g., PHP) and/or at the client side (e.g., JavaScript). ***Hint***: consider using ***regex*** for validation.
   1. The first and last name fields cannot be empty and consist of alpha letters only.
   2. ID must be 8 digits.
   3. Email begins with series of alphanumeric characters, followed by the “@” character, and domain name.
   4. Domain name consists of dot separated labels of 1 to 20 alphanumeric characters each, up to a maximum total length of 80 characters (including dot delimiters).
   5. Phone number must be in the form 999-999-9999. *Note, the email format is much more restrictive.*
5. If there are any errors, your program must **highlight the error input** and keep the correct inputs. You should not ask a user to “go back” to the previous page or input everything again.

**For help getting started:**

* Checkout this [Student Demo Registration Assignment Instructions and Details video](https://youtu.be/iwbHNukFs1M)
* Assuming you plan to use PHP/MySQL, check out the embedded ***Dynamic form sql php example.zip*** and other PHP files below as well as this [PHP/MySQL playlist](https://www.youtube.com/playlist?list=PLBkzRVY0qBad2S6aB-VbMalutW8Rd7aO5).



# Notes on Launching a MySQL Database Instances at AWS

The [following link](http://docs.amazonwebservices.com/AmazonRDS/latest/GettingStartedGuide/LaunchDBInstance.html) and/or embedded guide below give the detailed instructions to launch a MySQL DB instance using AWS RDS service:



**Key Notes:**

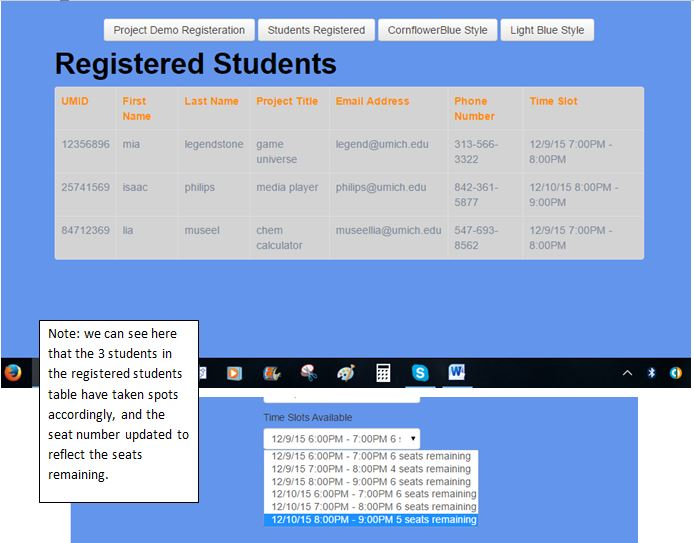
* DB instance class should be **“db.t2.micro”** to avoid any additional cost.
* Allocated storage should be about 5GB
* Database port should be 3306
* DB security group should be “default”. Note, make sure that it authorizes the connection: CIDR/IP: 0.0.0.0/0
* Take a note of your master username (e.g. root) and password
* Take a note of the “endpoint of the DB instance,” which is the host name of your MySQL server.
* Database and table names are case sensitive.
* To connect to your DB instance, you will need to install a MySQL client in your computer. MySQL workbench is one of the best tools. You may download and install it by following the embedded guide below:



**Note:** ***If you want to use a different service/storage option***, that’s fine as well, just make sure you document what you did, perhaps in a separate user guide, with screenshots and concrete steps similar to the guides embedded here so that if someone is following it, they’d be able to reproduce the same result easily 😊

## Example GUI

Note this doesn’t show the full GUI as you should be able to gather requirements from above and formulate into the 2 pages a client would look for as indicated in the requirements scope, in this case, me being the client 😊



# Deploy using a hosting service like Amazon Web Services

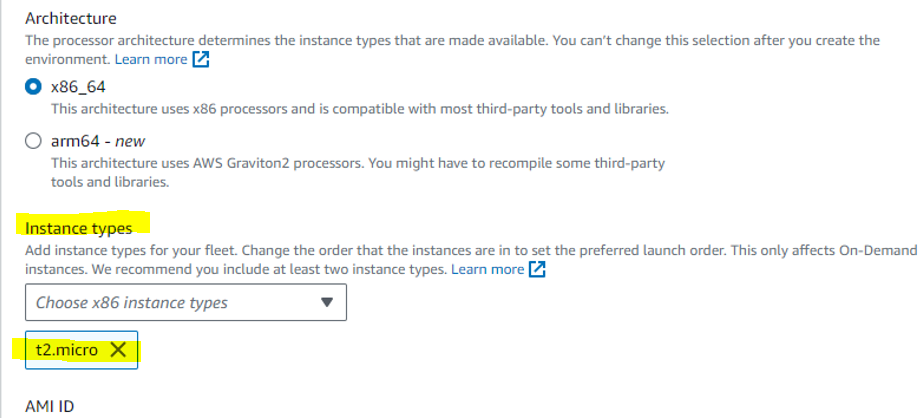
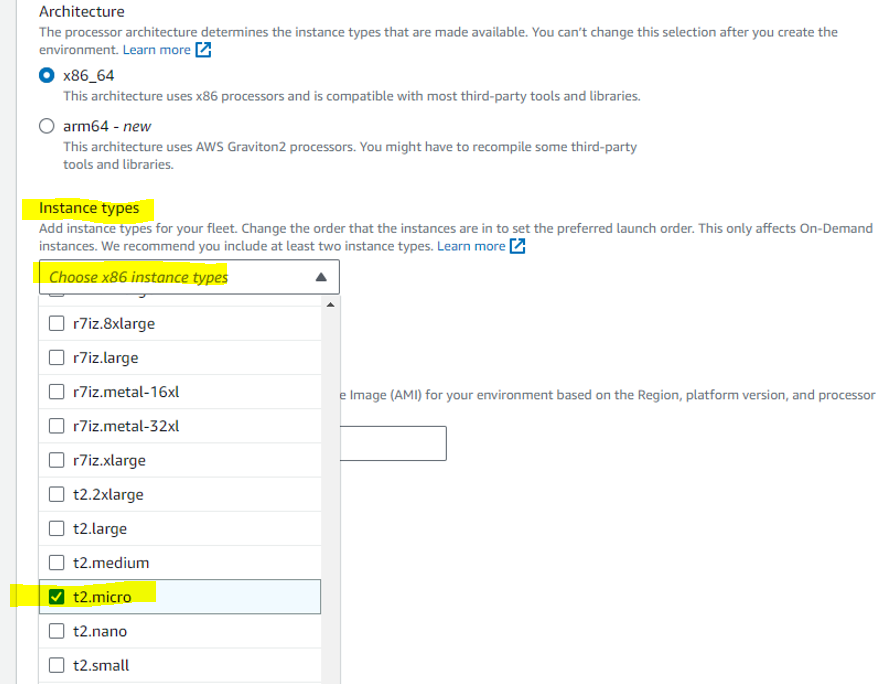
You may host your website using AWS Elastic Beanstalk, or Heroku (or any other hosting service you want to use like Microsoft’s Azure App Service, infinityfree, hostinger, etc.).

Some guides embedded below



Some key notes if you are deploying your website using the AWS Elastic Beanstalk service:

* Name your Environment in the format: **firstName-LastName-Project3**, e.g. john-smith-project3
* Remember to only launch **t2.micro (or t3.micro)** instances. All other instance types cost more money!



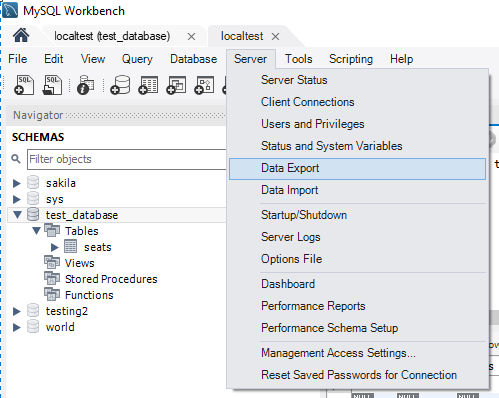
* Just create one application. If you make any changes, you can deploy a new version in your current application.
* Your main page must be named “**index**”
* Put all your html, CSS, JS, image files, etc... under the same folder, and zip them up. Please don’t include the parent folder in the zip file. In other words, the “index” file should be in the root directory of your zip file. Otherwise, the system will not be able to locate your files. Refer to the guide for more details.
* Following evaluation/grading, terminate the environment after each use (unless I ask you leave it on for grading). It will terminate the underlying EC2 instances.

**Grading**

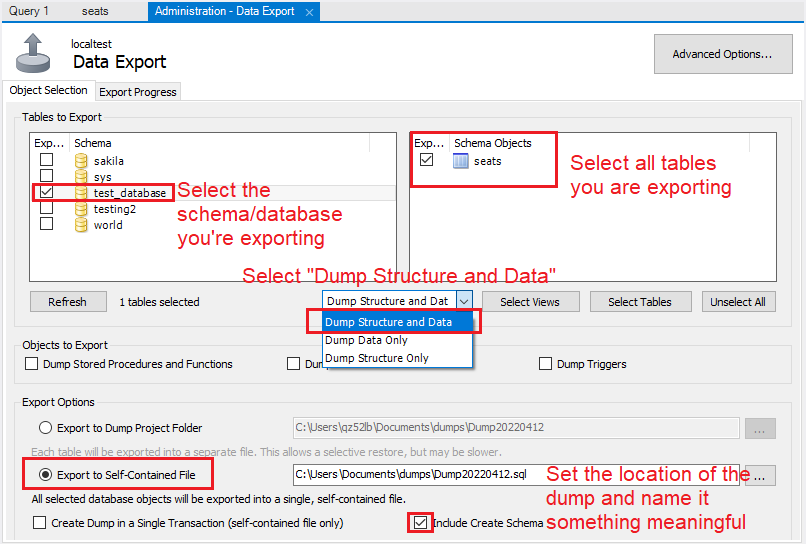
The grading criteria for this assignment will include: the extent to which your pages contain all the things described in the previous section, compliance to standards, the lack of broken links, the conciseness of the HTML, CSS, and JS files, and the lack of visually disturbing artifacts that make it very unpleasant to look at the pages. Your program should work in Google Chrome.

**What to turn in:**

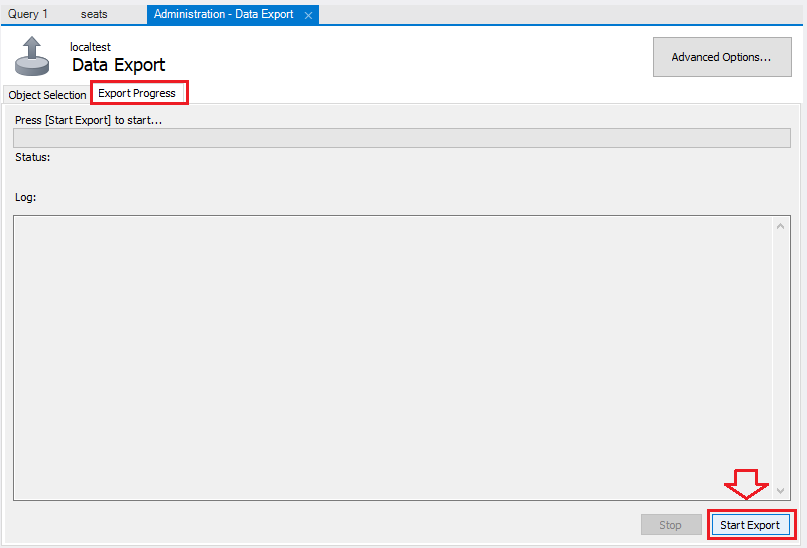
1. Upload all your files, including source codes and image files as **a single ZIP file** to the relevant assignment. Within the zip, include a SQL dump of the relevant table(s) that you are using for this assignment. To do this in MySQL workbench for example, you can use the "**mysqldump**" command or **data export** feature.
   1. Navigate to **Server** menu
   2. Select “**Data Export”**



* 1. Select the schema and relevant tables to export



* 1. You may need to switch to “**Export Progress**” tab to click “**Start Export**”



2. Include the URL of your web site in the comment section when submitting.