**1. Introduction of Team**

Daniel Jun <[djun1@umbc.edu](mailto:djun1@umbc.edu)>

* Constructed science trees
* Javascript updates and modifications
* Wrote documentation report

Victor Konen <[vkonen1@umbc.edu](mailto:vkonen1@umbc.edu)>

* Database reverse engineering
* PHP modified to work with database
* Wrote documentation report

Ka Tse <[ktse2@umbc.edu](mailto:ktse2@umbc.edu)>

* Constructed math tree and updated computer science trees
* Javascript updates and modifications
* Wrote documentation report

**2. Location of Project**

Web Page:

<http://datahole.ddns.net/cmsc433/project2/index.php>

Database:

<http://datahole.ddns.net/phpmyadmin/>

Username: accessaccount

Password: accesspass

**3. Project Description**

This project enables UMBC computer science majors to navigate their major requirements and select courses according to prerequisites they have met. On the website, users toggle between CMSC, Math, and Science classes to view the simplified tree structure of the CMSC degree path. Users may also enter their identifying information and their selected classes to a database through the website.

**4. What was added (Use this part to prove you deserve the 'A')**

**Class Type Toggle Feature**

The original tree structure of the CMSC class prerequisites was subdivided into different sections: Core CMSC requirements, the list of CMSC electives from which 2 must be taken, the list of CMSC electives from which 3 must be taken, optional and required Math courses, and optional and required Science courses. These sections are able to be toggled by the user so that only the section they wish to navigate is shown.

**Math and Science Classes**

Math classes required by the major and science classes that can be used to fulfill the 12 credit requirement and science track requirement were added to the database and tree structure as per the project requirements.

**5. What was Improved Upon**

**Database Design and PHP Modification**

The database structure and entries were not provided, and had to be reverse engineered from the PHP code. The code interacting with the database was also slightly modified to reflect the database design.

advisingInfo was intended to have a table for all of the courses in the system, a table for user entered information, and a table to link courses to users signifying that user has taken that course for every corresponding entry. The design of this database deviates from the intended design only by the addition of actual course IDs matching up the course names used by the javascript portion provided.

classPrereqs was intended to have a table for every class that had entries signifying the prerequisites to that class. The design of this database was heavily modified. Instead we implemented a single table that contains an entry for every prerequisite of a class linked to that class by actual course ids. The requirementClass column was borrowed from the intended design to account for the 400 level cmsc requirement for cmsc 447. If the value is set to one, only one of the prerequisites associated with a class is necessary to take that class.

The PHP code was modified to allow this design to work with the existing javascript, by providing functions to convert back and forth from actual course ids to the class names used by the provided code. It was also changed to work with the array method used by the previous group to store classes and their prerequisites.

These modifications reduced the complexity of the database design and made it trivial to migrate the course data from our project 1 to be used in this reverse engineered database for project 2.

**Aesthetic Organization of Website**

The student information box was moved to the center of the page to simplify things for users when they are ready to input data. The tree was also shifted to the middle and enlarged to improve readability.

**Divided Tree Structure**

The overall tree structure was broken up to into different sections make it easier to read and navigate.

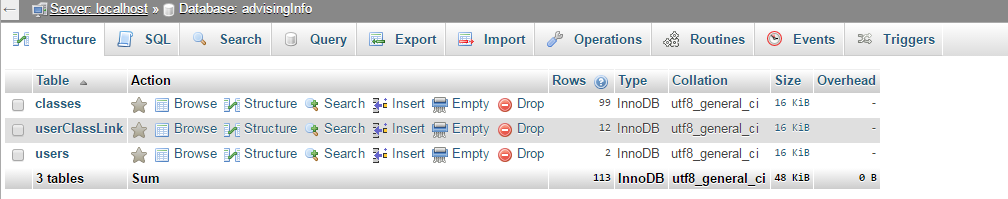
**Student Information Box**

The information for inputting and submitting student information was toggled to save space and make the website easier to navigate.

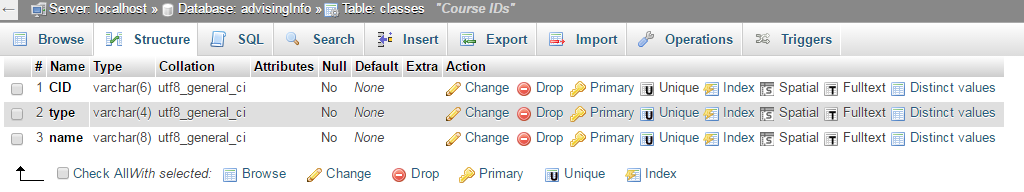
**Javascript Tree Interaction**

The provided code used a form to add and remove classes from the taken list and then updated the tree accordingly. Our improvement was to remove this form and have classes be added or removed when they are clicked in the actual tree structure. This improvement makes it easier for the user to add and remove classes from their taken list, as it removes typing and form navigation from this part of the interaction. Unfortunately the javascript manipulation of the tree styling based on the classes taken was coded in such a way that we were not able to modify it without recoding it completely. Our new functionality therefore uses the same form with javascript, hiding that from the user.  
 **6. Database Setup**

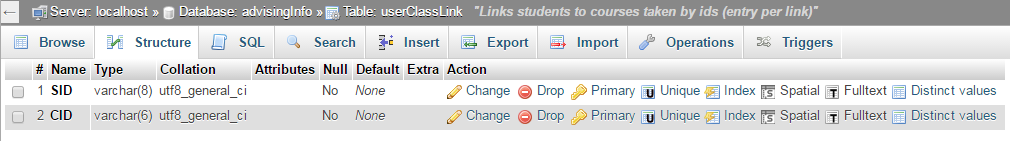
advisingInfo Database



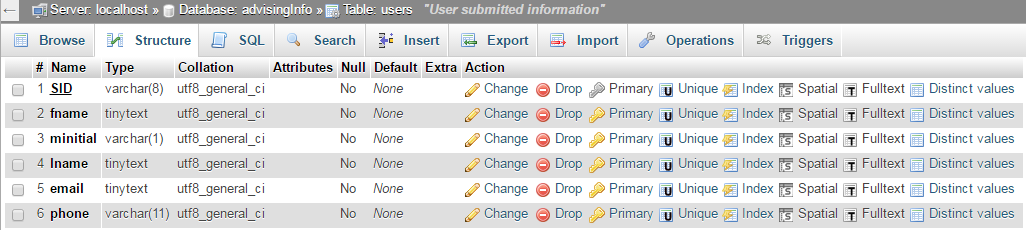
classes Table



userClassLink Table

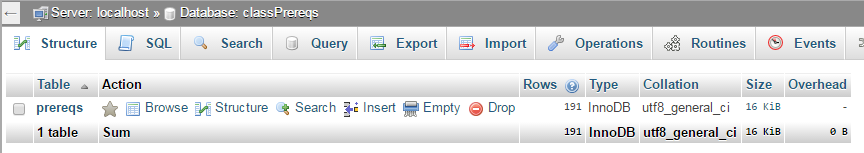


users Table

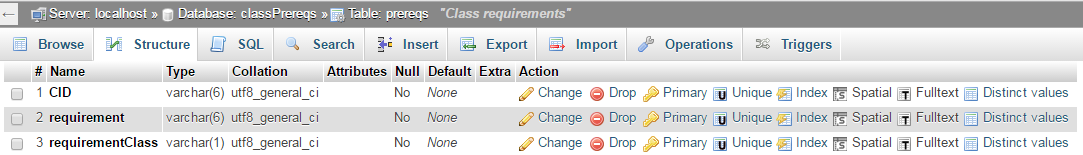


*--------------------------------------------------------------------------------------------------------------------*

classPrereqs Database



prereqs Table



*--------------------------------------------------------------------------------------------------------------------*

The database is setup as described in the improvements portion of this document under Database Design and PHP Modification. The reason for this structure is to allow for linking of student info to classes taken stored in the database and to allow for easy access to class prerequisites as stated in that section.

**7. DB dump**

The database dumps are in the db\_backups folder as .sql files.

advisingInfo\_initial.sql and classPrereqs\_initial.sql are dumps with just the database structure

advisingInfo\_populated.sql and classPrereqs.sql are dumps with the structure and all of the class information used by the PHP and javascript.

populate\_classes.sql contains the SQL statement used to populate the classes table within the advisingInfo database.

populate\_prereqs.sql contains the SQL statement used to populate the prereqs table within the classPrereqs database.

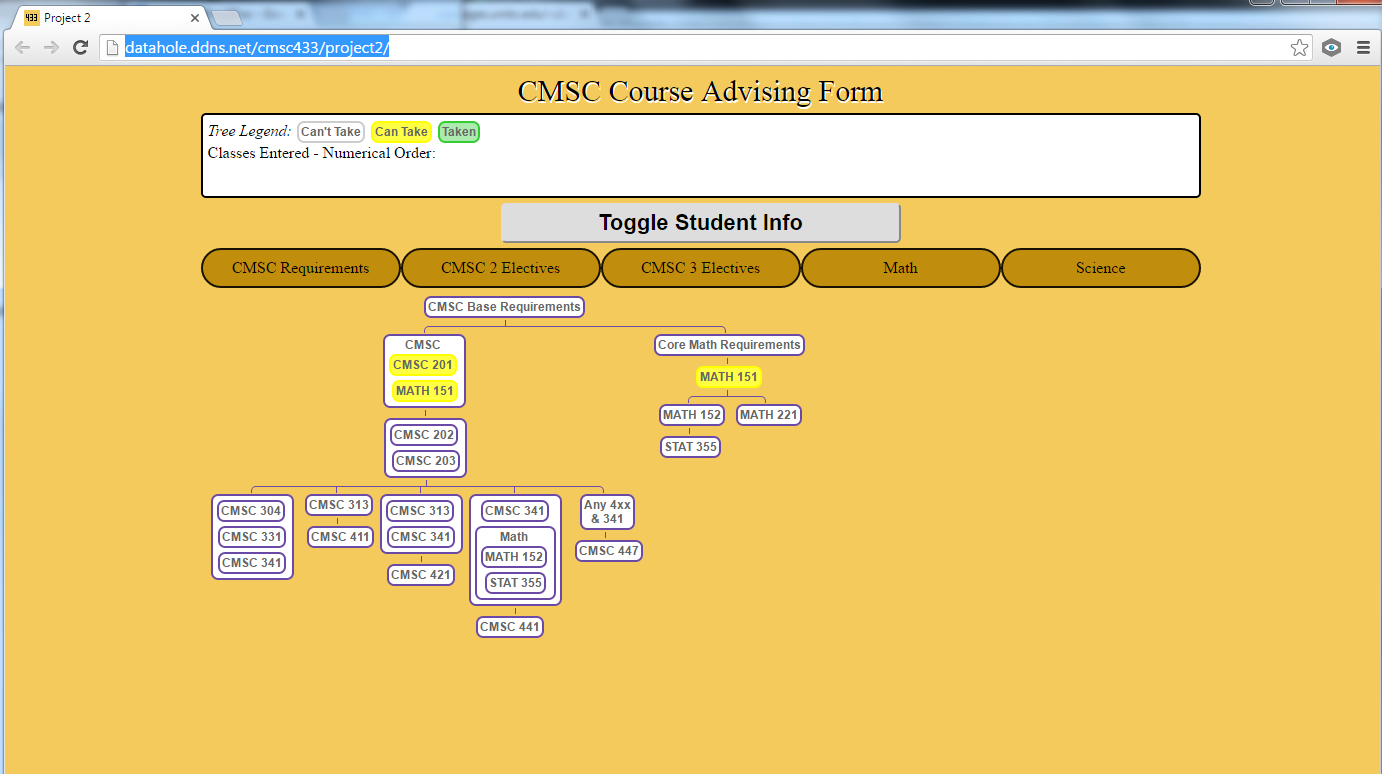
**8. Languages Used**  
 a. What was used and where  
 i. e.g. PHP, JQuery, JavaScript  
 b. Don't include HTML or CSS

PHP was used by the provided code to access the database, generate a class information array with prerequisites, and then pass that data to javascript for style manipulation of the tree structure. It was also used to validate the class data entered and ensure user information was set through post for submission into the database. Finally, it was used to generate the various validation messages on the final page after submission. The overall PHP functionality was not drastically altered to accommodate our improvements and added features.

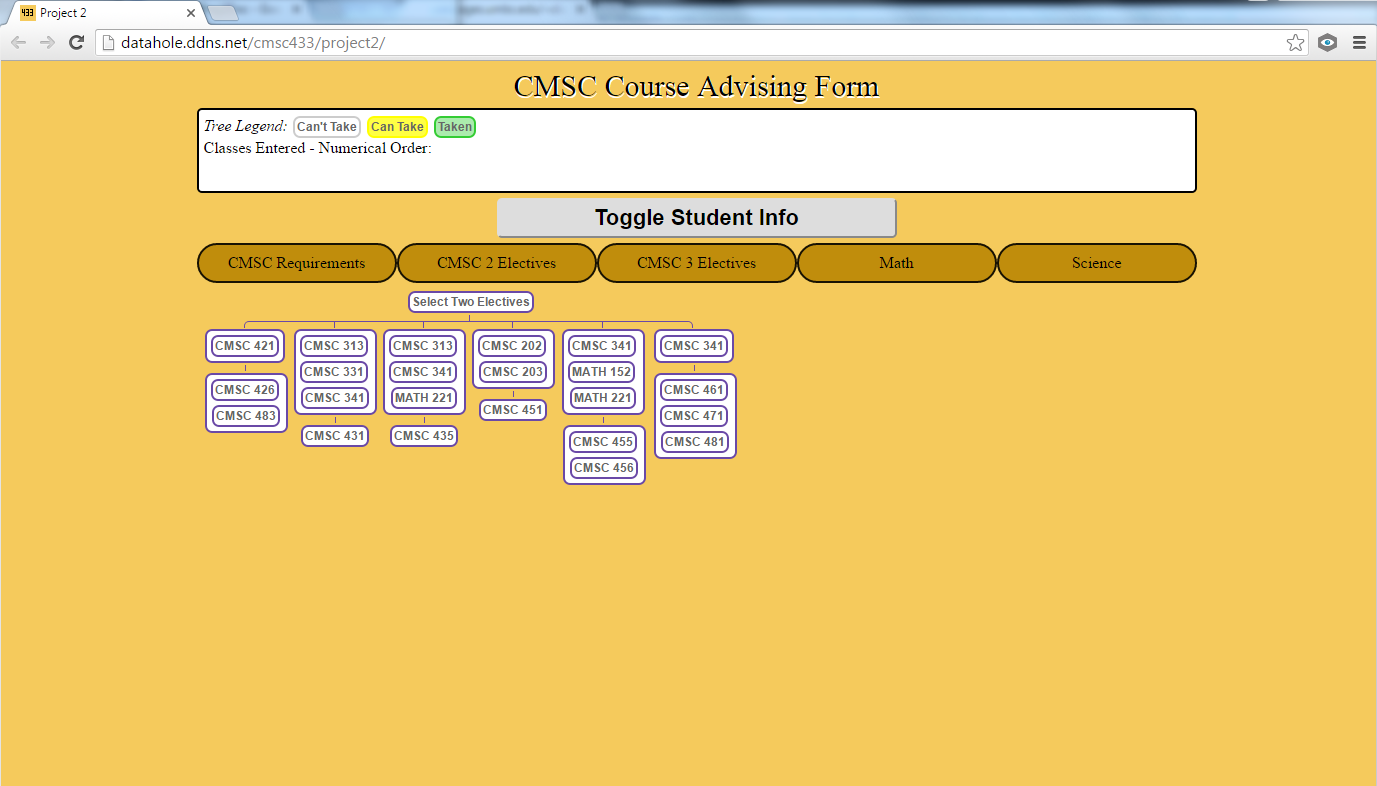
JavaScript was used in the implementation of the Class Type Toggle feature to hide and show the appropriate trees upon click events. The user interaction with the tree of adding and removing classes from the taken list and updating the style accordingly was also facilitated by JavaScript. Finally, it is used to toggle between the student information form and the tab/tree content.

**9. Slick Sheet**

***Starting Page***

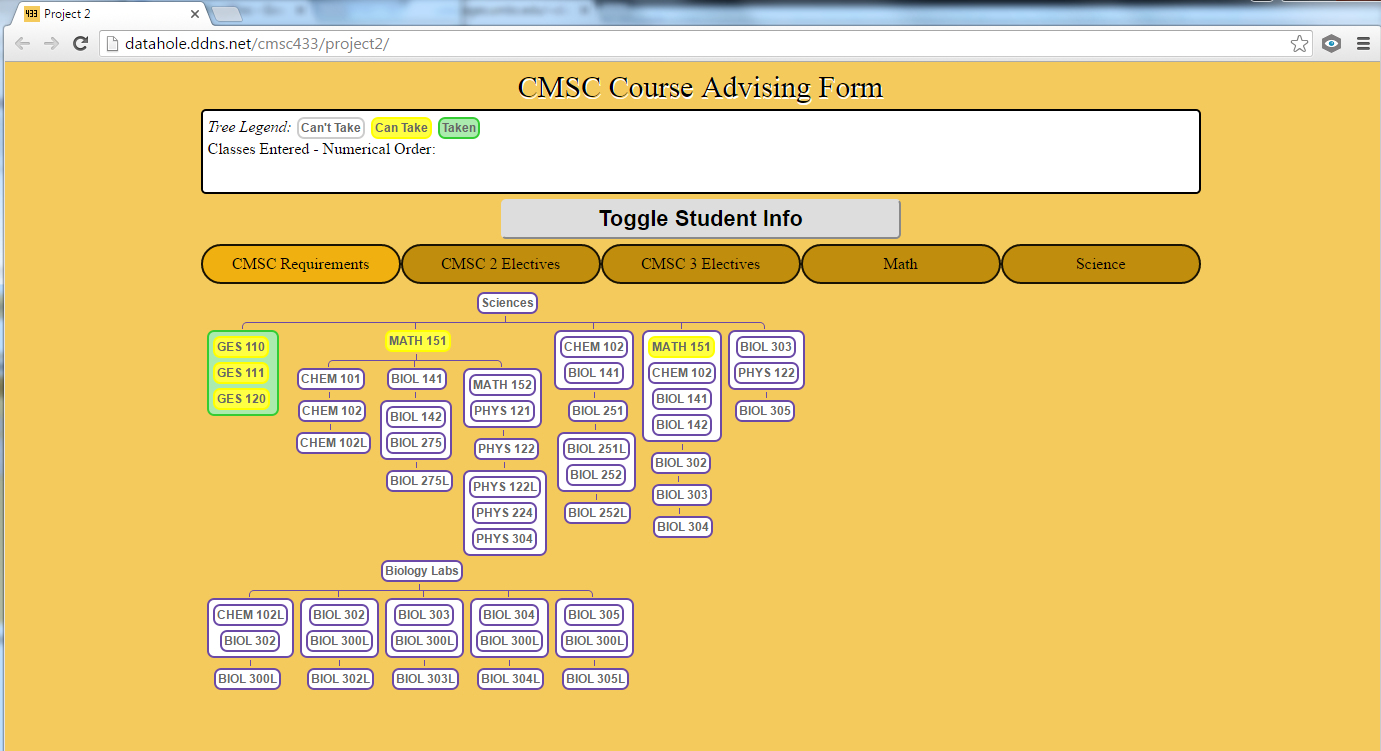


*--------------------------------------------------------------------------------------------------------------------*

***Class Type Toggling***

*Clicking “CMSC 2 Electives” button*

*-Displays the Select Two Electives tree from which two electives must be taken to fulfill the CMSC degree requirements.*

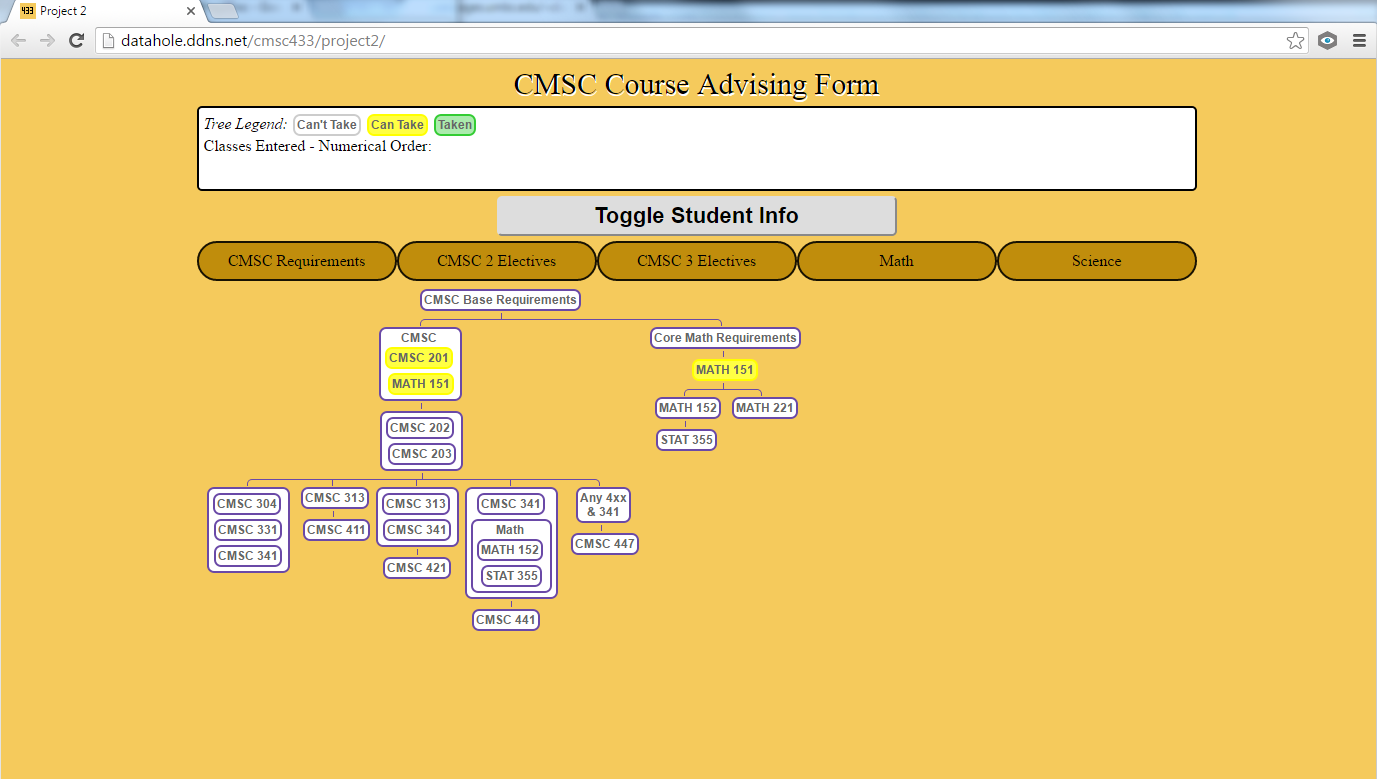


*Clicking “Science” button*

*-Displays Science trees*

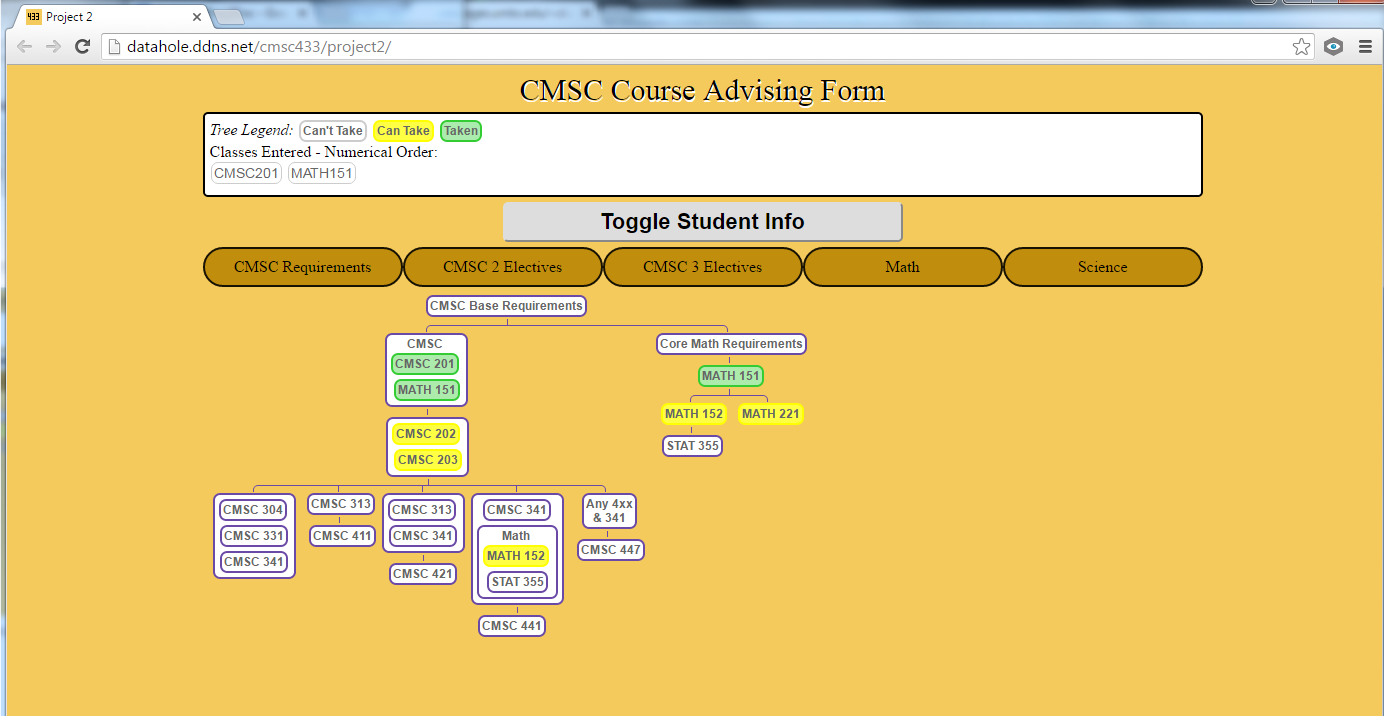
*--------------------------------------------------------------------------------------------------------------------*

***Selecting Courses***



*Before Selecting CMSC 201 and MATH 151*

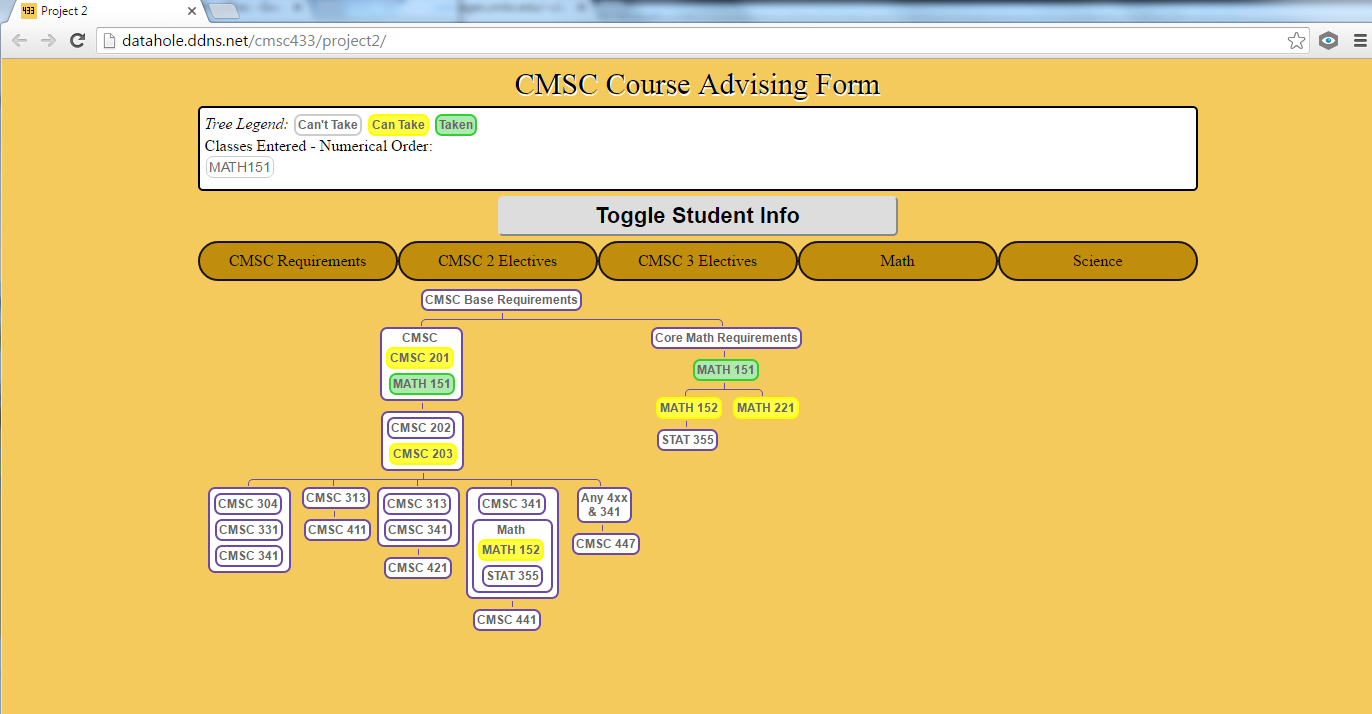
*-CMSC 201 and MATH 151 are highlighted yellow to indicate that they can be taken.*



*After CMSC 201 and MATH 151 are selected*

*-CMSC 201 and MATH 151 are highlighted green to indicate that the user has taken them. CMSC 202 and CMSC 203 are highlighted yellow to indicate that the user can take them. MATH 152 and MATH 221 are also highlighted yellow because MATH 151 was selected.*

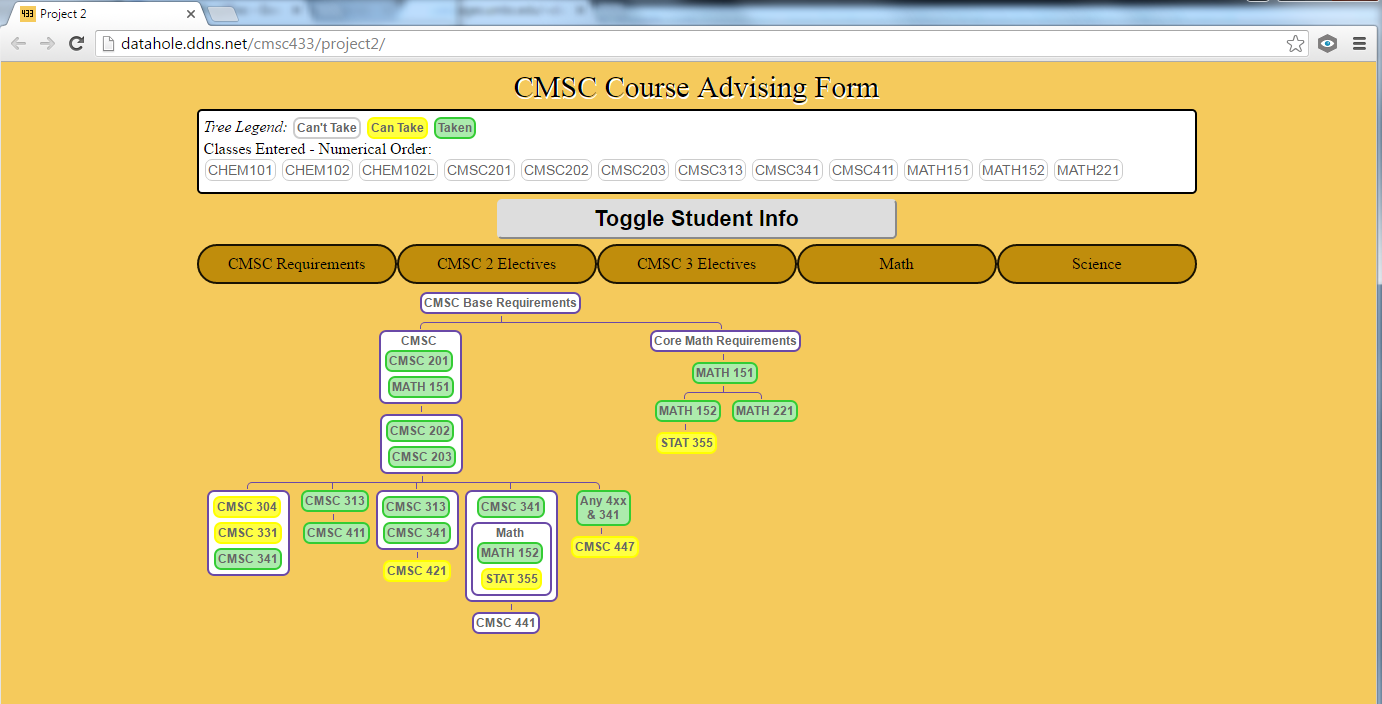
*-CMSC 201 and MATH 151 are displayed on the Classes Entered box at the top.*



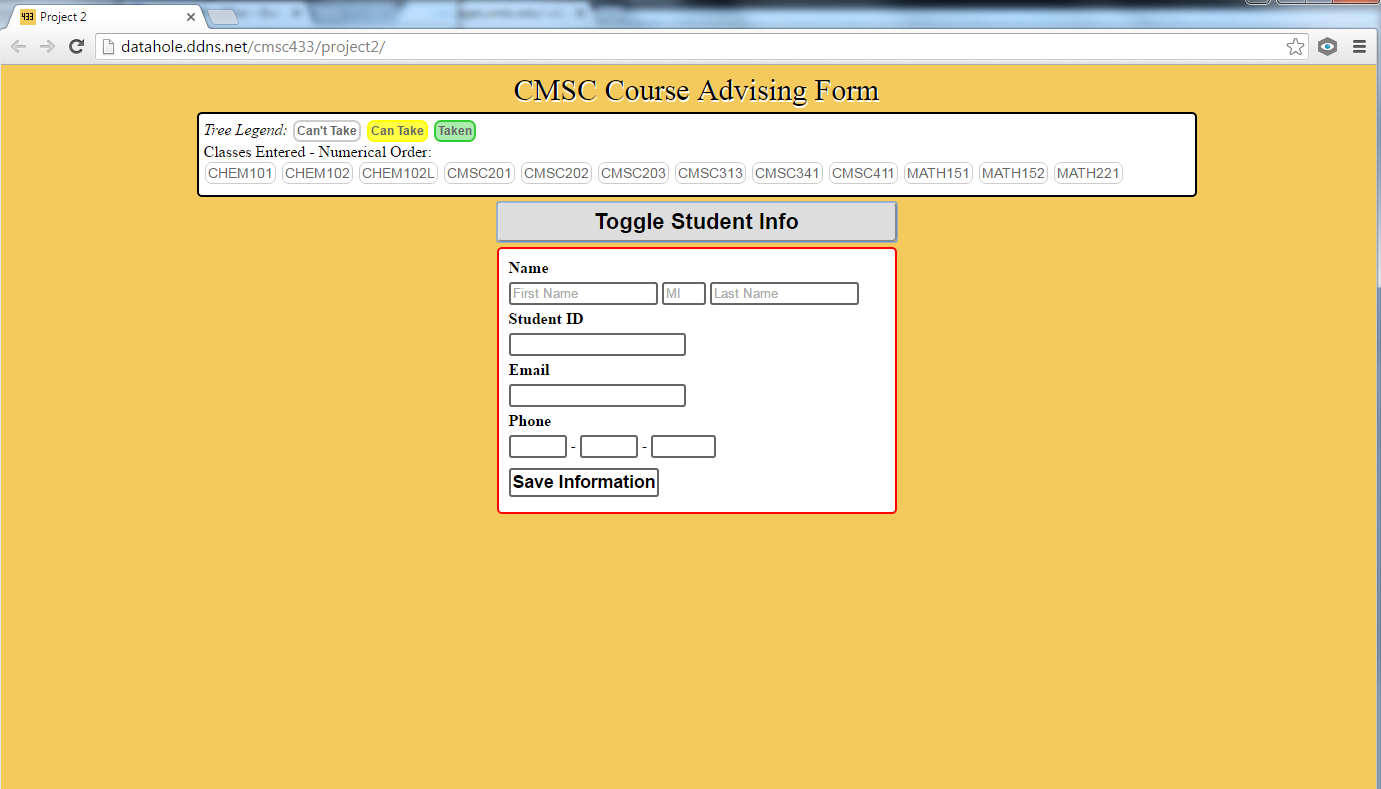
*CMSC 201 is clicked again*

*-If CMSC 201 is clicked again, CMSC 201 changes back to yellow and the classes for which CMSC 201 are a prereq change back to white. In this case, for example, CMSC 201 is a prerequisite for CMSC 202. As such, CMSC 202 changes from yellow to white because it can only be taken if CMSC 201 has already been taken. However, MATH 152, MATH 221, CMSC 203 remain yellow because their only prerequisite is MATH 151.*

*--------------------------------------------------------------------------------------------------------------------*

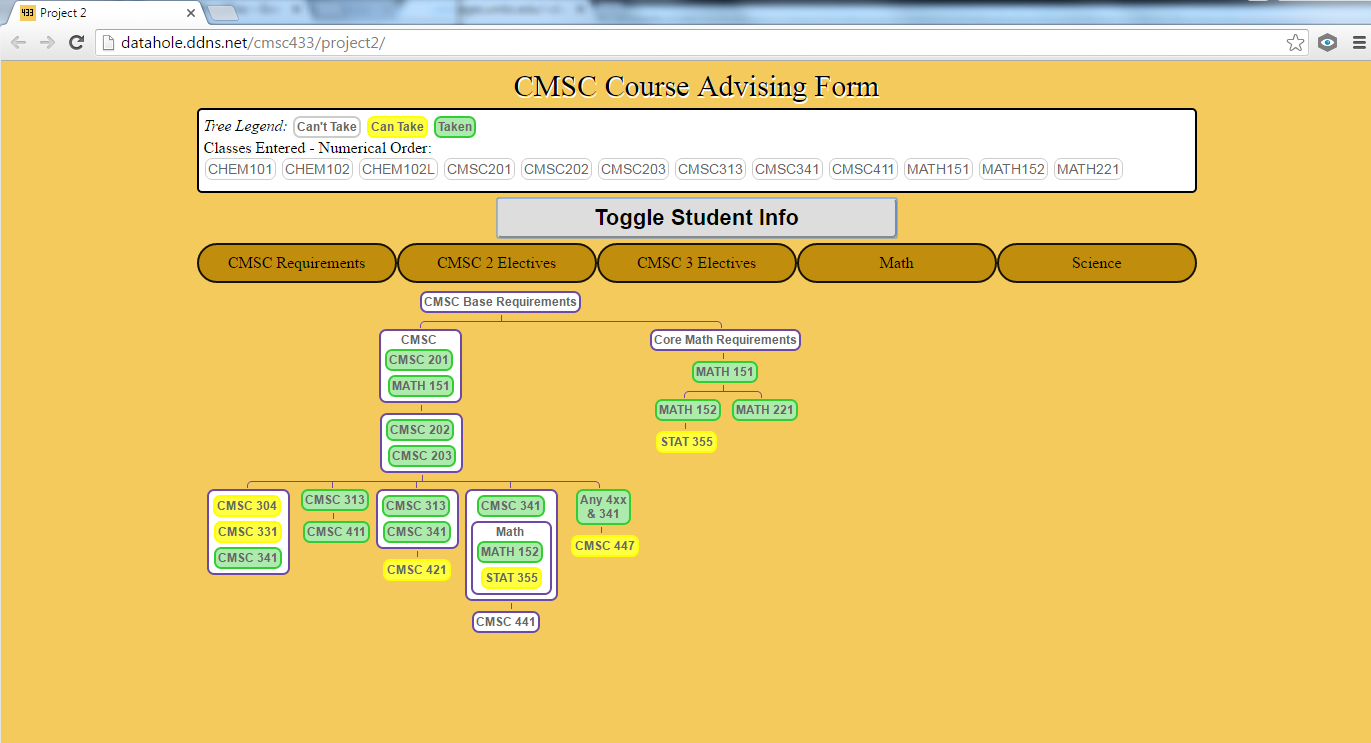
***Data Submission***

*Before clicking “Toggle Student Info” button*



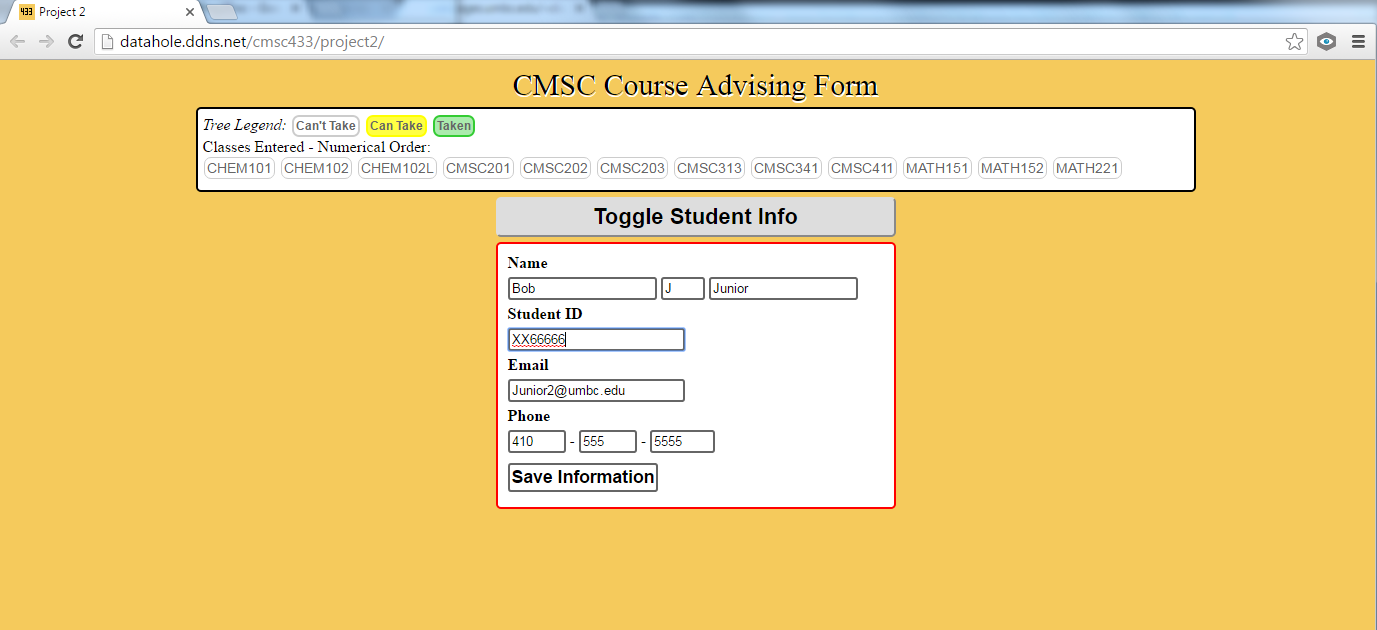
*After clicking “Toggle Student Info” button*

*Clicking the “Toggle Student Info” button displays a form for the user to fill out and submit data to the database. It also hides the tab and trees. (Notice the classes that were selected remain visible in the box above)*

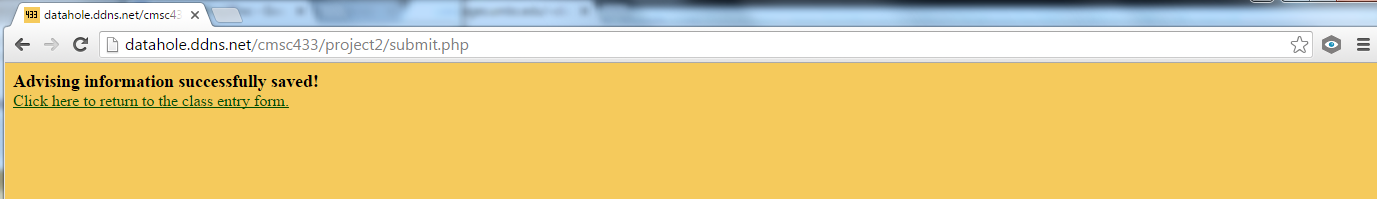


*Clicking the “Toggle Student Info” button again*

*-The form is hidden and the class type tabs and trees are visible again.*



*Filling out the form then clicking “Save Information”*



*DATA SUCCESSFULLY SAVED!!!*