STEM Kids Learning Management System Manual

Contents

1 Introduction

The basic foundation for the LMS for STEM Kids NYC has been implemented using Node.js. It is currently hosted for free on Heroku at the follow url: STEM Kids NYC LMS

2 Using the LMS to upload courses

The current LMS gives an administrator the capabilities of uploading new course information to the site. Students can enroll in these courses, watch video lectures, and take quizzes.

2.1 Loggin in

The index page of the LMS has a button 'Admin' that prompts a user for username and password information. To access the admin side of the site, use the following:

Name: yvonne
Password: Stemkids1234

2.2 Creating a Course

Once an administrator has entered their login information correctly, they will be prompted with a page with the option to 'Create a Course'. After clicking this button, our application will generate a random, and unique, 6-character course code, and prompt the user for the desired course title and course summary. After entering this information, the new course will automatically be entered into the database with the corresponding 6-character course code, course title and course summary.

2.3 Adding Lectures

After adding a new course, the admin will be redirected to a page that says "Add Lecture 1 to Course XXXXXX". This is where an admin can add a new information for the first lecture in the new course they have created. A lecture can have the following information:

1. Lecture title: Title for the lecture in the course
2. Lecture summary: Summary for the lecture in the course
3. Lecture video: Video for the lecture in the course **Note: the video must be hosted on another site (i.e. youtube) in order to be uploaded properly
4. Lecture Handout: A file that is a handout for the current lecture. **Note: This has not been implemented. See Section 9 subsections for more details
5. Quiz Questions and answers: Each lecture can have up to 10 quiz questions to test students on the lecture information. Each question has 4 answers, any of which can be marked as correct using the checkbox to the right of each answer. Remember to mark at least one question as correct. To add a new quiz question, use the 'Add Question' button.

Once the lecture information has been entered fully and accurately, an subsequent lecture can be added to the course, or the course can be uploaded if all the lectures have been added.

2.4 Adding Additional Lectures

After adding the first lecture, additional lectures can be continually added using the "Submit & Add New Lecture" button. This process can continue (in the same way detailed in Section 2.3. The last lecture should be uploaded using the "Upload Course" button.

2.5 Uploading a Course

After entering information for the last lecture, the course can but uploaded and activated on the side using "Upload Course". On success, the page will reload to the initial admin index page, and a message conveying the success can be seen. This means the course is now available for students to take online in the LMS. Uploading the course automatically activates the course, which means students will be allowed to take the course immediately. See Section 9 subsections for more details.

3 Using the LMS to Update a course

The current implementation has an additional button 'Update Course', which should allow an administrator to update a course. This has not been implemented. See Section 9 subsections for more details.

4 Creating an Account on the LMS

A new user can create an account from the index page of the LMS. A user can click the 'Create an Account button' and enter the following information (*required):

1. First Name* (text input)
2. Last Name* (text input)
3. Age* (number input)
4. Gender* (drop down options)
5. Grade* (text input) - might want to consider changing this to a dropdown or number input depending on the grade-range of students using the LMS
6. School* (text input) - might want to consider making this a dropdown with schools, if there is a concrete list of schools that will be using the LMS. This is suggested because there could be a lot of variation/misspellings (from young children) with text input.

7. Email* (text input with validity checking)
8. Phone number (optional)
9. Username* - availability of the username is checked in real-time against the database. A green background means the username is available, red means the username is already taken
10. Password*

The password is the only information that is encrypted in our database. We recommend additional protection of user information. See section **??** for details. After entering the information above accurately and fully, the user can submit. If there are no issues with their information, the user will be brought to the log in page, and can user their new log in information to enter the site.

5 Loggin in as a Student

A user can log in from the index page using the 'Log In' button. The user must enter the correct username and password combination to successfully enter the site.

6 Viewing and Enrolling in Courses

Once the user has logged in, they are brought to their profile page, where they can view available courses, current courses, completed courses, and future courses. Each course is shown with their corresponding course title and description.

6.1 Current Enrolled Courses

The first section on the profile page "My Current Enrolled Courses" lists all courses that a student has enrolled in. The student can then press "Continue" for a specific course to continue learning from the displayed courses.

6.2 Available Courses

A user can see all available, active courses in this section. This displays all the courses that are available for the student to enroll in. If a student is interested in enrolling, they can click 'Enroll' and the class will be added to their 'Current Enrolled Courses'

6.3 Course History

This section displays all the courses a student has finished. They can click "Do Again" to see the lecture information from the class they have completed. To see more details about how/when a course is added to the course history menu, see subsection in section 9

6.4 Future courses

This section allows students to see a preview of courses that have not been activated. This means students can view the course title and description but cannot click the button to view lectures or other content of the course.

7 Taking a course

After choosing a course to take, the student will be prompted with a list of all the lectures that exist for the course.

7.1 Modular Lectures

A newly enrolled student will only be allowed to view lectures chronologically, based on completing the quizzes. This means a user can only click and view lecture for the lectures for which they have completed classes and the currently lecture they have yet to do the quiz for. All subsequent lectures are displayed on the lecture page, but a student cannot click them until they finish the quiz for the current lecture they are on.

7.2 Lecture Page

Once a student has chosen an available lecture, they will be presented with all the information for that lecture. This information includes:

1. Lecture Video: This is the video that was uploaded as the informational video for the class

2. Lecture Handout: This is a tab where students should be able to view the corresponding handout for the current lecture. See section 9 for details.

3. Lecture Quiz: This tab displays the current quiz questions and answer options for the current lecture. See section 7.2.1 for details.

7.2.1 Quizzes

A prepared student can take a quiz for a certain lecture from clicking the "Quiz" tab on the specific lecture page. This allows students to fill in a single answer for each question. Once they have entered information for each question, they can click "Submit". If their answer was correct it will be highlighted in green. If not, their answer will be highlighted in red, and the correct answer will be highlighted in green. All the quiz information for a user is entered in the 'quiz history' table of our database.

After completing a quiz, this allows the student move on to the next lecture. There are a couple ideas that have not been implemented for this section that might be interesting functionality options to consider for an enhanced LMS experience. See Section 9 for more details.

8 Finishing a Course

A student has successfully completed a course when they finish the last quiz for the last lecture of the course. After completing the last quiz, a student can return to their profile and see the course is now part of their course history, and now longer in their currently enrolled course list.

9 Unimplemented Functionality

Due to time constraints and the depth of this large project, there are many functionality options that have are not currently implemented on the LMS. We detail suggestions here

9.1 Uploading and viewing handouts (Admin)

The admin side of the LMS has the option to upload a handout for each lecture in a course. That being said, the actual storing of the uploaded handouts is not functional. There is a column in the 'classes' database table that can be used to store relative paths to handouts stored in the file system.

Once the storing of the handout onto the file system has been implemented, the 'Handout' tab for a specific lecture must also be implemented to display the handout corresponding to the lecture. Essentially, in order to implement the handout functionality for the LMS, the following functionality must be added: store the uploaded handout for submitting each lecture on the admin side while creating a course on the host file system, insert the relative path of the handout, with a unique .pdf name, into the database table, and finally use the unique identifier for the handout for a specific lecture to render the handout in the handouts tab for the corresponding lecture.

9.2 Editing a Course (Admin)

A large part of the admin side has already been implemented, where an administrator can create, add, and active new courses. That being said, there are not editing capabilities for any of the current courses on the site. We recommend that the 'Edit Course' button on the admin side is implemented so that when an admin click on the button to edit a course, all the current courses in the 'courses' database table are displayed. An admin should then be able to click on a specific course to view and edit the corresponding course title, course description, and all lecture and quiz information. They should additionally be able delete the course, activate/inactivate the course, and delete individual lectures here.

9.2.1 Course Activation (Admin)

Currently, when an admin uploads a course from the admin side, the course is automatically activated so students can immediately enroll and view lectures for the course. An easy work around for uploading a course for the time being, without activating it, is to add all the lecture information for a specific when creating it, and never clicking the 'Upload Course' button on completion. Instead, an admin can click "Submit & Add New Lecture" for each lecture they want to upload and return to the admin index page. This way, all the lecture information will be added to the course without activating it immediately. This is obviously not the ideal functionality for uploading an inactive class, but it will work until the editing course capabilities have been fully implemented.

9.3 Edit Icon Button (student)

We thought it would be nice for students to have a choice in icon for their profiles. Ideally, when a user clicks the 'Edit Icon' button from their profile page, it should open up a pop-up window of an array of fun animal icons. The choice of icon should be saved in a database table. None of this implementation has been done, except for the button itself on the profile page.

9.4 Edit Profile Button (student)

In order to update user information, there is also a 'Edit Profile' button on the profile page for a user. We recommend that when a user clicks this, a separate page will be opened, and displays all user information. This should allow users access to change this information and update it in the database.

9.5 Reward System, Enhancing Functionality and UX/UI

While we currently store all the quiz information once a student completes a quiz for a given course, we do not have a reward system implemented for students to see their progress in a fun and interactive way. We recommend that the information in the quiz_history database table be used to begin the implementation for a reward system for students.

Additionally, when a student submits a quiz, the program currently grades the student's submission by displaying the correct answers in green. That being said, there is no easy way for a user to navigate back to the page displaying all the lectures for the current course. Currently, they must use the arrow in the right-hand side of the menu bar to navigate back to their profile/home page and click "Continue" for the course to continue on to the next lecture. We recommend that additional functionality be implemented to a user can easily return to the list of lectures for a current course from a specific lecture page.

10 Security

While for the users are encrypted in our database, very little other information is safe in our current implementation. We heavily recommend reviewing and revising how users information is sent back and forward between the front and back end. This means minimizing malicious attacks by safely maintaining user information in the database, as well as maintaining user sessions for currently logged on users.

11 Database Tables

All the database table schemas can be seen in the loader.js file. This file has comments that detail all the information stored in the database tables.

12 Source code files

Here, we explain all the source files in our directory.

All the html pages we are currently using for the LMS can be found in the 'updated html' file:

- account.html: create account page (student)
- addLecture.html: add a lecture to a course (admin)
- adminhome.html: home admin page with Create and Edit Course buttons (admin)
- createcourse.html: initial create course page where an admin can add a course title and summary (admin)
- index.html: landing page for all of LMS (all)
- lecturelist.html: Page that displays all lectures for a current course (student)
- login.html: login page to end LMS (student)
- profile.html: profile page for a student user, also displays course information for available courses, enrolled courses, etc.

- updated_course.html: page that displays specific information for a certain lecture, including video and quiz information

**Note: course.html, lecturepage.html, and all files in the html/ directory are not being used at the moment.

The corresponding css pages, customized fonts, images and javascript files for these pages can also be found in this directory.

The server is in the main directory named server.js.

13 Node Modules

Our program is depending on the following node modules:

- any-db
- any-db-sqlite3
- basic-auth
- basicauth-middleware
- body-parser
- consolidate
- express
- hogan
- semantic
- socket.io
- sqlite3

14 Current setup of LMS on Heroku

Hosting was done by deploying our github information to Heroku. We used this set up guide. There are a couple of important things to note about the current state of the site as it stands.

14.1 Database management

Heroku does not support SQLite, which is the only database management system we learned in class. Since the database, as it stands, is not that large, it is hosted on github and can be used directly on our app. That being said, using a free account for hosting on Heroku only allows for an active server for 18/24 hours in a day. Therefore, when someone attempts to access the site after the server has been idol, we have reason to believe that the database will be reset to its previous state as it stands on Github. In other words, any changes to database entries that are done from the site on Heroku will not be saved. We realize that DreamHost has been chosen as the webhosting partner for the current STEM Kids NYC. Unfortunately, we were not able to create a web application that could be hosted on DreamHost in any of the languages it supports because our class did not teach Rails, PHP, or Python, and many students in our group were unfamiliar with all of these languages.

If it is possible, we suggest researching a host that can supposed SQLite for the databases (perhaps it will be possible with DreamHost), while hosting the website itself on Heroku. The code will need to be modified to connect the the database add-on instead of connecting directly to the stemkids.sqlite3, as it is now. Specifically, this is done on line 11 of our current code in server.js:

```
var conn = anyDB.createConnection('sqlite3://stemkids.sqlite3');
```

Given the time constraints and the lack on knowledge in the subject, we were unable to find a solution to the database hosting. We hope that the above suggestion will help.