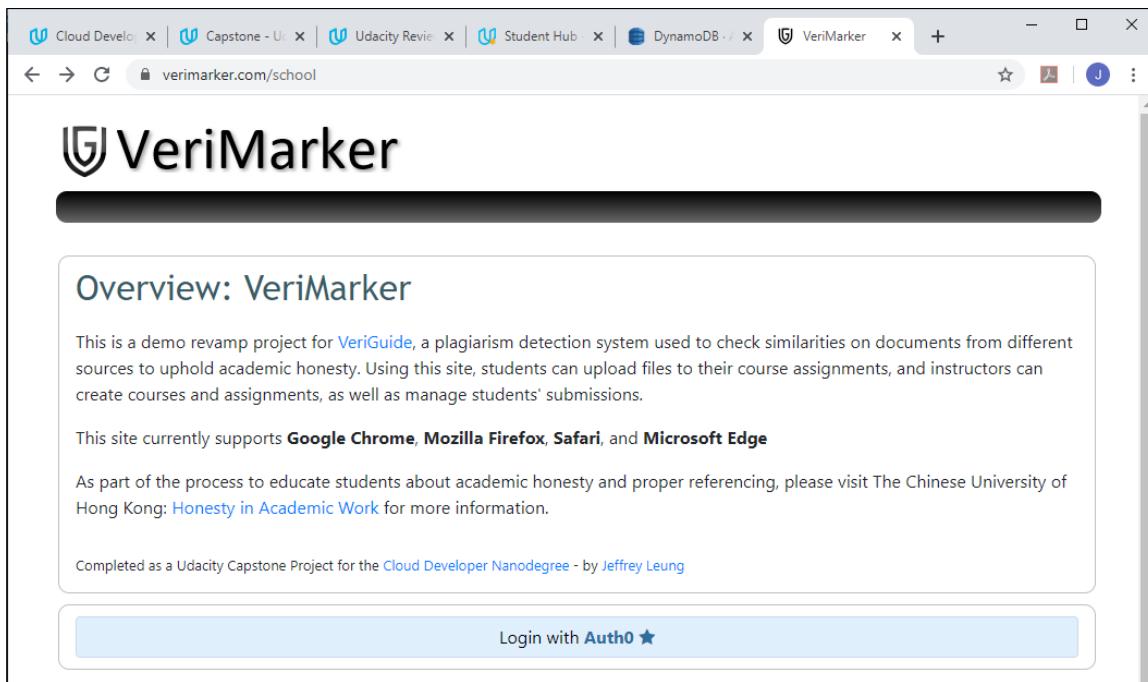


Contents

Cloud Developer Capstone Project - VeriMarker	3
1. Introduction.....	3
1.1 Implementation.....	3
1.2 Demo	3
1.4 Trouble shooting	4
2. Summary of User Functions	4
2.1 Summary of user functions for the Student.....	4
2.2 Summary of user functions for the Instructor.....	5
3. Unit Tests – Primary flow	6
Section 3.1: Register new user as an Instructor:.....	9
Section 3.2: Instructor create course / update course description	11
Section 3.3: Instructor delete a course	13
Section 3.4: Instructor create assignments.....	13
Section 3.5: Instructor delete assignment	17
Section 3.6: Register two new users as Students.....	17
Section 3.7: Student “Mary Lee” to upload submission to the course assignment and check her submission history.....	19
Section 3.8: Student “Mary Lee” upload the second submission to the same course and assignment.....	24
Section 3.9: Student “Mary Lee” deleted the submission uploaded in previous section (3.8).....	26
Section 3.10: Student “David Liu” to upload submission to Assignment 1 for Course “2019-NURS-1151” and check submission history.....	27
Section 3.11: Instructor to check on student submissions uploaded to assignment 1 in section 3.7 to 3.10.....	29
Section 3.12: Student update submission references.....	31
Section 3.13: Instructor add comments and score to the student’s submissions.....	34
4. Project Rubrics.....	37
4.1 Functionality.....	38
4.1.1 The application allows users to create, update, delete items	38
4.1.2 The application allows users to upload a file.	39
4.1.3 The application only displays items for a logged in user.....	39
4.1.4 Authentication is implemented and does not allow unauthenticated access.....	40
4.2. Code Base	41
4.2.1 The code is split into multiple layers separating business logic from I/O related code.....	41
4.2.2 Code is implemented using async/await and Promises without using callbacks.....	43
4.3. Best Practices	43
4.3.1 All resources in the application are defined in the "serverless.yml" file	43
4.3.2 Each function has its own set of permissions.	43
4.3.3 Application has sufficient monitoring.	44
4.3.4 HTTP requests are validated.....	49

4.4 Architecture.....	54
4.4.1 Data is stored in a table with a composite key.	54
4.4.2 Scan operation is not used to read data from a database.....	55
5. Appendix.....	56
5.1 Implementation of the Client.....	56
5.2 Linking Auth0 JWT token to VeriMarker User Account.....	57
5.2 Deployed Angular Client to AWS S3 and CloudFront using route 53.....	60

Cloud Developer Capstone Project - VeriMarker



1. Introduction

VeriMarker is a plagiarism detection system used by both students and instructors to manage submissions, assignments and courses, with the primary goal to ensure students did not plagiarized their work.

Using VeriMarker, student can submit files to the course assignments, view the similarity result, grades, and instructor's comments of their own submissions, and download the files of their own past submissions. Instructor can create and manage their own courses, assignments, and comment / grade the student's submissions uploaded to their own assignments. For the purpose of the capstone project, the similarity result of the student's submission is only a simulated number created by Math.random.

The Github repository of the project can be found in: https://github.com/jsleung1/project_capstone_verimarker and it also contains the instructors on how to setup the project.

1.1 Implementation

"Option 2" was selected for the Udacity Cloud Developer Capstone project. The backend was implemented using Serverless and AWS Lambda functions as taught in the project under Course 5 "Develop & Deploy Serverless App". The client is a Single Page Application written in Angular 8. The Angular client was deployed to AWS S3 / CloudFront with the help of Route 53. The deployed URL is <https://www.verimarker.com>

1.2 Demo

For demonstration purpose without setting up the Angular project, please go to <https://www.verimarker.com> and login via Auth0 using your Gmail account. You can also use the following Gmail accounts that are already registered in VeriMarker for a quick demo:

Gmail Login:	VeriMarker Role
udacitystudent106@gmail.com	Student
udacitystudent206@gmail.com	Student
udacityteacher306@gmail.com	Instructor

If you decide to use your own Gmail account, you will need to register in VeriMarker once you logged in from Auth0, in order to identify you as a Student or Instructor in VeriMarker. Subsequent logins will link your Auth0 token to your VeriMarker user account (stored in DynamoDB) in order to display the suitable menu functions for Student or Instructor. You can always change your user role to Student or Instructor by going to the “User Settings” in the main menu.

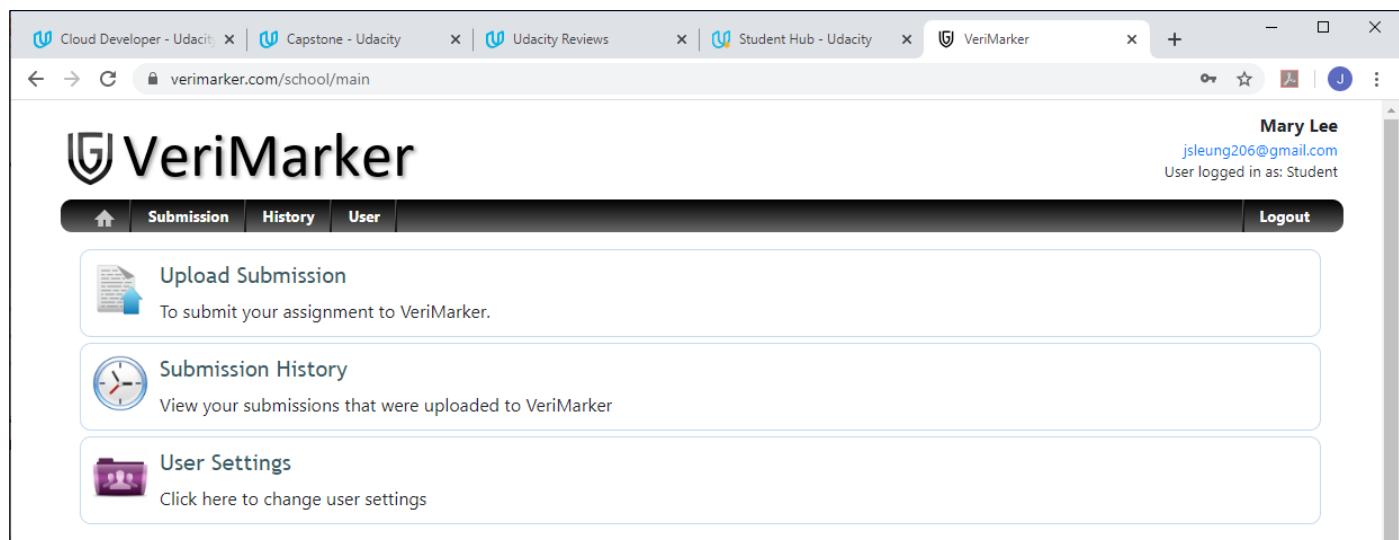
1.4 Trouble shooting

In the rare occasion that you receive an alert with message “invalid token error”, please logout from VeriMarker. Another option is to delete the cookies in your Web Browser (it is caused when you logged in VeriMarker without logout the previous user).

2. Summary of User Functions

Before talking about VeriMarker is able to meet the project Rubrics in section 4, the following is a summary of the user functions of Instructor and Student. Please note that if this is the first time you logged in from Auth0 for the Gmail account, you will need to register as a new VeriMarker user as Student or Instructor.

2.1 Summary of user functions for the Student



The screenshot shows a web browser window with multiple tabs open at the top. The active tab is 'VeriMarker'. In the top right corner of the browser, there is a user profile for 'Mary Lee' with the email 'jsleung206@gmail.com' and the note 'User logged in as: Student'. Below the browser, the VeriMarker interface is visible. It features a header with a logo, a navigation bar with links for 'Submission', 'History', and 'User', and a 'Logout' button. The main content area contains three boxes: 'Upload Submission' (with a file icon), 'Submission History' (with a clock icon), and 'User Settings' (with a folder icon). Each box has a brief description below it.

Upload Submission	Student upload a file to the assignment, by select (in this order) the course, assignment and the instructor. Student must enter the Submission References (to indicate what references that they used to complete the assignment) before able to click on the “Upload Submission” button.
Submission History	Student can view their own submissions uploaded to the course assignments. Student can update the submission references which can be read by the instructor, and able to delete the submission (by click on the ‘Withdraw button’). Once the submission is deleted, the instructor will not able to view the student’s submission. Finally, in each submission, the student can view the similarity result, and the comments and score assigned by the instructor. The student can also download the file of the submission.
User Settings	Student can update the Email and change the user role (Student or Instructor) of their VeriMarker user account.

2.2 Summary of user functions for the Instructor

The screenshot shows the VeriMarker web application interface. At the top, there are four browser tabs: 'Amazon Web Services Sign-In', 'VeriMarker', 'VeriMarker', and 'Settings'. The main window displays the 'VeriMarker' logo and the name 'Monica Tang' with the email 'parispurchase@gmail.com'. It also indicates 'User logged in as: Instructor'. Below the header, there is a navigation bar with links for 'Courses', 'Assignments', 'Submissions', 'User', and 'Logout'. The main content area contains four cards: 'My Courses' (Teaching Courses by Academic Year), 'My Assignments' (Assignments in My Teaching Course), 'My Submissions' (Student Submissions uploaded to me for My Assignments), and 'User Settings' (Click here to change user settings).

My Courses	Instructor can create courses (Student will able to view courses created by all the instructors).
	Instructor can view their own courses (Instructor cannot view courses created by other instructors.)
	Instructor can update the description of their own courses (Instructor cannot update courses created by other instructors.)
	Instructor can delete their own courses if it does not contain any assignments. (Instructor cannot delete courses created by other instructors.)
My Assignments	Instructor create assignment (by first selecting their own course).
	Instructor can view the assignments of their own course.
	Instructor can update the assignment description and assignment due date.
	Instructor can delete the assignment if the assignment does not contain any submissions.
My Submissions	Instructor can only view the students' submissions uploaded to their own assignments (Instructor cannot view the students' submissions uploaded to other instructors)
	Instructor can add comment and assignment score to the student submission.
	Instructor can download the file of the student submission.
	Instructor cannot delete the student submission (only student can delete their own submission by the 'Withdraw Submission' button).

	Instructor cannot update the Submission References of the student submission (only student can update the Submission References which is used by the Similarity Engine in the future).
User Settings	Instructor can update the Email and change the user role (Student or Instructor) of their VeriMarker user account. Please note the student will be unable to select the instructor from the dropdown in the submission upload screen if the instructor's user role was changed to a Student.

3. Unit Tests – Primary flow

The focus of this section is to present the Unit Tests of the primary flow described in the following table below. You may go directly to [Section 4: Project Rubrics](#) for meeting the project specifications. You may also find the Unit Tests already cover some of the requirements of the project Rubrics. Please click on the section link to go to the details of each unit test.

In DynamoDB, ensure Users, Courses, Assignments and Submissions Table are all empty.			
Section	Unit Test Name	Unit Test Description	Expected Result
3.1	Register new user as an Instructor	Register Gmail account (parispurchase@gmail.com) as Instructor, name = Monica Tang	Successfully registered the Gmail account as an Instructor in VeriMarker. Only Instructor functions are accessible by the user.
3.2	Instructor create course / update course description	Create following courses for Academic Year 2019: 2019-NURS-1151 Development of Nursing 2019-NURS-2112: Disaster Nursing 2019-WRONGCODE: Course to be deleted	Successfully created three courses under Academic Year 2019.
3.3	Instructor delete a course	Instructor delete course “2019-WRONGCODE: Course to be deleted”	Successfully deleted the course from the instructor.
3.4	Instructor create assignments	Create following assignments for 2019-NURS-1151: Assignment 1 Assignment 2 Create following assignments for 2019-NURS-2112: Assignment 1 Assignment X	Successfully created the assignments under each course.
3.5	Instructor delete assignment	Instructor delete “Assignment X” in course 2019-NURS-2112.	Instructor able to delete own assignment.
3.6	Register two new users as Students	Register Gmail account (jleung206@gmail.com) as Student, name = Mary Lee Register Gmail account (jleung106@gmail.com) as Student, name = David Liu	Successfully registered two Student accounts in VeriMarker. Only student functions are accessible by the user.
3.7	Student “Mary Lee” upload submission to the	Upload file to the following course / assignment: Course: 2019-NURS-1151: Disaster Nursing	File should be correctly uploaded to Assignment

	course assignment and check her submission history.	Assignment 1 Instructor: Monica Tang	1 in Course "2019-NURS-1151", to instructor Monica Tang.
<u>3.8</u>	Student "Mary Lee" upload the second submission to the same course and assignment.	Upload file to the following course / assignment: Course: 2019-NURS-1151: Disaster Nursing Assignment 1 Instructor: Monica Tang	Student's Submission History will show two submissions in Assignment 1 under Course "2019-NURS-1151"
<u>3.9</u>	Student "Mary Lee" deleted the second submission uploaded in previous section (3.8)	In submission History, student select the second submission, click on "Withdraw" submission to delete the submission.	The submission is deleted from the system (Student Mary Lee and Instructor Monica Tang will not able to view the deleted submission).
<u>3.10</u>	Student "David Liu" to upload submission to Assignment 1 for Course "2019-NURS-1151" and check submission history.	David Liu upload submission to the same course and assignment as in section 3.7 to 3.9.	The submission is successfully uploaded. Only one submission is listed in the Submission History of David Liu.
<u>3.11</u>	Instructor to check on student submissions uploaded to assignment 1 in section 3.7 to 3.10.	Instructor "Monica Tang" to check submissions uploaded by student "Mary Lee" and "David Liu" under Assignment of	Two submissions (one from each student) are found in Assignment 1 in Course 2019-NURS-1151. Instructor should able to download the file from both submissions.
<u>3.12</u>	Student update the submission references.	Student "Mary Lee" updates the references of her submission	Both Mary Lee and Instructor Monica Tang should able to view the updated Submission References.
<u>3.13</u>	Instructor add comments and score to the student's submissions.	Instructor Monica Tang add comments and assignment score to submissions of Mary Lee and David Liu.	Instructor able to update the submission with score and assignment comments. Students should able to view the instructor's score and comments.

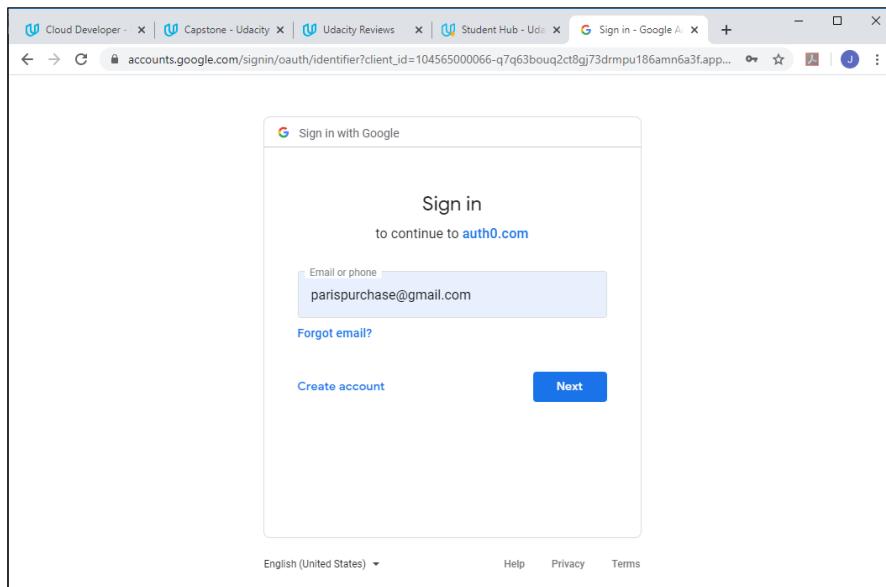
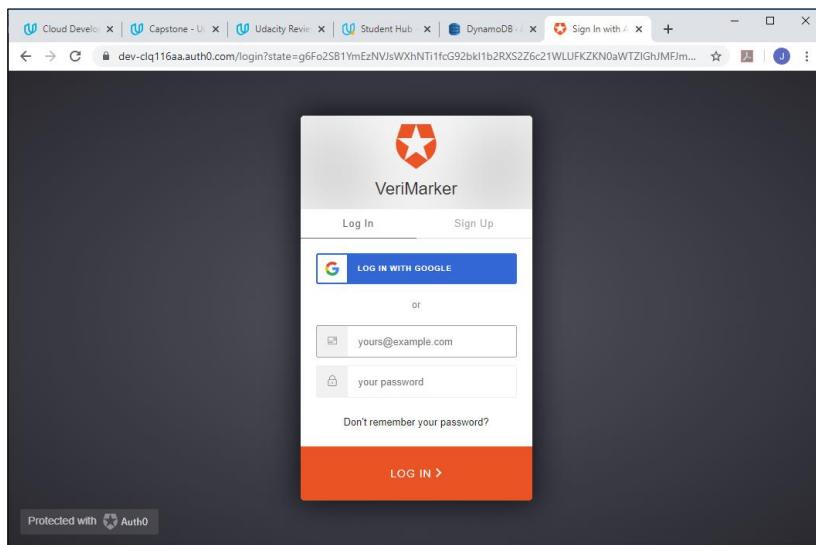
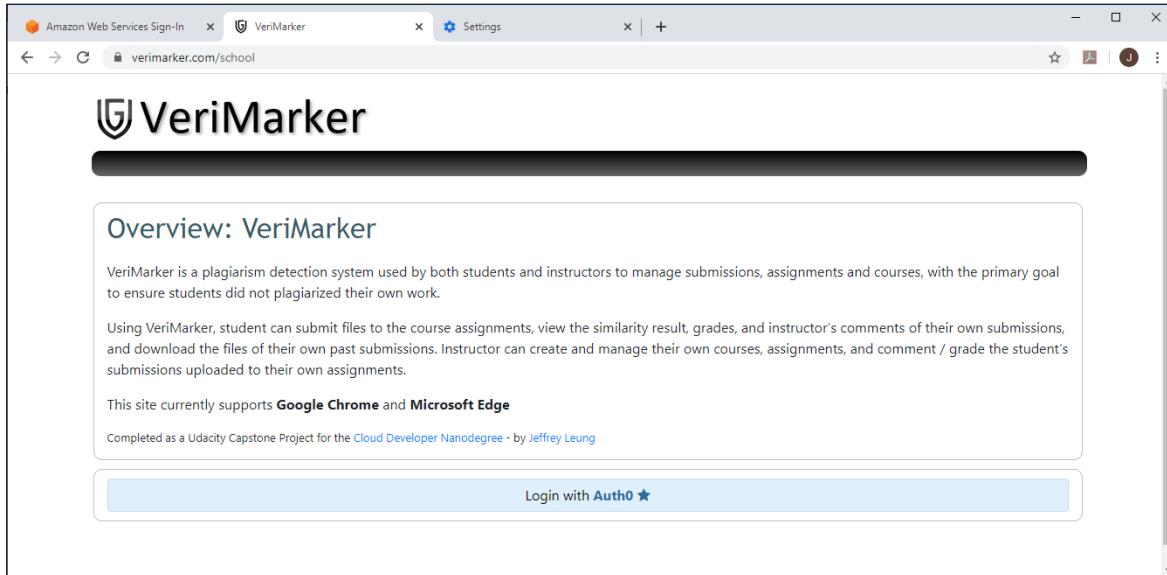
Initial data setup before the execution of the Unit Test:

Professor Jeffrey Leung jleung506@gmail.com Instructor	2019 Courses: 2019-ELTU-1001: Foundation English for University Studies Assignment 1 Assignment 2 Assignment 3 2019-ELTU-1002: English Communication for University Studies Assignment 1 Assignment 2 Assignment 3
---	--

	<p>2019-ELTU-2003: English Through Popular Culture Assignment 1</p> <p>2019-ELTU-3011: English for Arts Students II Assignment 1 Assignment 2 Assignment 3 Assignment 4</p> <p>2018 Courses: 2018-ELTU-1001: Foundation English for University Studies Assignment 1 Assignment 2 Assignment 3</p> <p>2018-ELTU-1002: English Communication for University Studies Assignment 1 Assignment 2 Assignment 3</p> <p>2018-ELTU-2202: Language Awareness for Teachers 1: Listening and Speaking Assignment 1 Assignment 2 Assignment 3</p>
Dr. Karen Li jisleung406@gmail.com Instructor	<p>2019 Courses:</p> <p>2019-ACCT-3004: Accounting Practicum and Experiential Learning Assignment 1 Assignment 2 Assignment 3 Assignment 4</p> <p>2019-ACCT-2111: Introductory Financial Accounting Assignment 1 Assignment 2 Assignment 3</p> <p>2019-ACCT-1111 Foundations in Financial Accounting Assignment 1 Assignment 2</p>

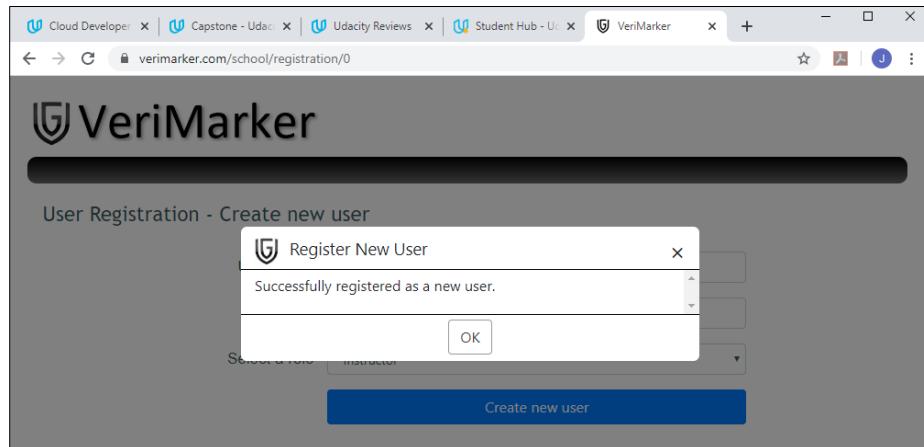
Section 3.1: Register new user as an Instructor:

1. Logged in to VeriMarker using Auth0:



2. After signed in to Google, the user was redirected to the VeriMarker User Registration page. Enter the User name as Monica Tang, Email as parispurchase@gmail.com and select the role as Instructor, and click Create new user:

The screenshot shows a web browser window with the URL verimarker.com/school/registration/. The page title is "User Registration - Create new user". It contains three input fields: "User name" with the value "Monica Tang", "Email" with the value "parispurchase@gmail.com", and a dropdown menu "Select a role" set to "Instructor". Below these fields is a blue "Create new user" button.



3. After the user is successfully registered, the user is navigated to the main menu of VeriMarker:

(continued to next page ...)

The screenshot shows the VeriMarker application interface. At the top, there are four tabs: "Amazon Web Services Sign-In", "VeriMarker", "VeriMarker", and "Settings". The current tab is "VeriMarker". The URL in the address bar is "verimarker.com/school/main". On the right side, a user profile is displayed for "Monica Tang" (parispurchase@gmail.com), indicating they are logged in as an Instructor. The main content area features a navigation bar with links: Home, Courses, Assignments, Submissions, User, and Logout. Below the navigation bar is a sidebar with four items: "My Courses" (Teaching Courses by Academic Year), "My Assignments" (Assignments in My Teaching Course), "My Submissions" (Student Submissions uploaded to me for My Assignments), and "User Settings" (Click here to change user settings).

4. User parispurchase@gmail.com was successfully registered as an Instructor with name Monica Tang. User is logged in to VeriMarker and the user can only access the Instructor functions from the main menu.

Section 3.2: Instructor create course / update course description

1. Click on the My Courses menu. It returns an empty list of courses. Click on the button to create a new course

The screenshot shows the "Courses" page of the VeriMarker application. The top navigation bar includes "Home", "Courses", "Assignments", "Submissions", "User", and "Logout". A dropdown menu labeled "Filter Course by Academic Year" is set to "All". A prominent blue button on the right says "+ Click [here] to create a New Course". The user profile at the top right is for "Monica Tang" (parispurchase@gmail.com) as an Instructor.

2. Enter the Course Code (must be unique for the selected Academic Year) and course description. Select the academic year (2019) for the course and click “Create New Course”:

(continued to next page ...)

Cloud Developer - Udacity | Capstone - Udacity | Udacity Reviews | Student Hub - Udacity | VeriMarker

verimarker.com/school/main/courses/createCourse/2019

VeriMarker

Logout

Create a New Course

Course Code: 2019-NURS-2112

Course description: Disaster Nursing

Academic Year: 2019

Create New Course

- Instructor can now see the new course, and able to update the course description (by click on the update button):

Cloud Developer - Udacity | Capstone - Udacity | Udacity Reviews | Student Hub - Udacity | VeriMarker

verimarker.com/school/main/courses

VeriMarker

Logout

Filter Course by Academic Year: All

+ Click [here] to create a New Course

2019-NURS-1151

Course Description: Development of Nursing revised description

Open **Update** **Delete**

Cloud Developer - Udacity | Capstone - Udacity | Udacity Reviews | Student Hub - Udacity | VeriMarker

verimarker.com/school/main/courses

VeriMarker

Logout

Filter Course by Academic Year: All

+ Click [here] to create a New Course

2019-NURS-1151

Course Description: Development of Nursing revised description

Update Course

Successfully updated the course.

OK

Open **Update** **Delete**

- Repeat the above steps to create courses for “2019-NURS-2112: Disaster Nursing” and “2019-WRONGCODE: Course to be deleted”:

The screenshot shows the VeriMarker web application interface. At the top right, the user is logged in as 'Monica Tang' (parispurchase@gmail.com). The main content area displays three course cards:

- 2019-WRONGCODE**: Course Description: "Course to be deleted". Buttons: Open, Update, Delete.
- 2019-NURS-2112**: Course Description: "Disaster Nursing". Buttons: Open, Update, Delete.
- 2019-NURS-1151**: Course Description: "Development of Nursing revised description". Buttons: Open, Update, Delete.

A blue button at the top right says "Click [here] to create a New Course".

Section 3.3: Instructor delete a course

1. Refresh the Web browser. Click on the Delete button of the course “2019-WRONGCODE”.

The screenshot shows the VeriMarker web application interface after a course deletion. A confirmation dialog box is displayed in the center of the screen:

Delete Course
Successfully deleted the course.
OK

The background shows the course list with the course "2019-NURS-2112" still present but dimmed, indicating it has been deleted.

2. The course was successfully deleted and the instructor no longer able to view the deleted course.

Section 3.4: Instructor create assignments

1. Instructor click on “Open” button of course 2019-NURS-1151. The page returns an empty list of assignments under the course as indicated below:

(continued to next page ...)

Monica Tang
parispurchase@gmail.com
User logged in as: Instructor

Viewing Assignments for Course **2019-NURS-1151**

Click [here] to create a New Assignment

- Instructor clicked on “Click [here] to create a New assignment”. Instructor enters “Assignment 1” as the assignment name and enter the description “Assignment 1 for 2019-NURS-1151’. Instructor should also specify the assignment due date (using “2020 January 6, 09:30” in the below example):

Course Code: 2019-NURS-1151

Course description: Development of Nursing revised description

Instructor name: Monica Tang

Assignment name: Assignment 1

Assignment description: Assignment 1 for 2019-NURS-1151

Assignment due date: 2020-01-06 09 : 30

Create New Assignment

- Instructor click “Create New Assignment”. The user successfully created the course. The user is redirect back to the list of assignments under course “2019-NURS-1151”, with the one assignment that was just created:

The screenshot shows the VeriMarker interface for viewing assignments. At the top, there are tabs for Cloud Developer - Udacity, Capstone - Udacity, Udacity Reviews, Student Hub - Udacity, and VeriMarker. The VeriMarker tab is active. The URL in the address bar is verimarker.com/school/main/courses/c147ca86-e127-4fe3-8e65-8e744252571d/assignments. On the right, user information is displayed: Monica Tang, parispurchase@gmail.com, User logged in as: Instructor, with a Logout button. A navigation bar below has links for Home, Courses, Assignments, Submissions, User, and Logout. A blue button on the right says "Click [here] to create a New Assignment". The main content area shows "Viewing Assignments for Course 2019-NURS-1151". Assignment 1 is listed with a clock and calendar icon. The assignment description is "Assignment 1 for 2019-NURS-1151". Below it, the assignment due date is set to 2020-01-06 at 09:30. Buttons for View Student Submissions, Update Assignment, and Delete Assignment are present.

4. Repeat the above steps to create Assignment 2 under course “2019-NURS-1151”:

The screenshot shows the VeriMarker interface for creating a new assignment. The top navigation and user info are identical to the previous screenshot. The main content area is titled "Create a New Assignment". The form fields are as follows: Course Code (2019-NURS-1151), Course description (Development of Nursing revised description), Instructor name (Monica Tang), Assignment name (Assignment 2), Assignment description (Assignment 2 for 2019-NURS-1151), and Assignment due date (2020-01-24 at 13:00). A blue "Create New Assignment" button is at the bottom.

5. The list of assignments under course “2019-NURS-1151” will display Assignment 2 and Assignment 1:

(continued to next page ...)

The screenshot shows the VeriMarker application interface. At the top, there are several browser tabs: "Cloud Developer - Udacity", "Capstone - Udacity", "Udacity Reviews", "Student Hub - Udacity", and "VeriMarker". The main content area displays two assignments for the course "2019-NURS-1151".

Assignment 2:
Assignment Description: Assignment 2 for 2019-NURS-1151
Assignment Due Date: 2020-01-24 13:00
Buttons: View Student Submissions, Update Assignment, Delete Assignment

Assignment 1:
Assignment Description: Assignment 1 for 2019-NURS-1151
Assignment Due Date: 2020-01-06 09:30
Buttons: View Student Submissions, Update Assignment, Delete Assignment

6. Repeat the above steps to create “Assignment 1” and “Assignment X” under course “**2019-NURS-2112**”:

The screenshot shows the VeriMarker application interface. At the top, there are several browser tabs: "Amazon Web Services Sign-In", "VeriMarker", "Settings", and "VeriMarker". The main content area displays two assignments for the course "2019-NURS-2112".

Assignment X:
Assignment Description: Assignment X (to be deleted)
Assignment Due Date: 2029-07-10 09:00
Buttons: View Student Submissions, Update Assignment, Delete Assignment

Assignment 1:
Assignment Description: Assignment 1 for 2019-NURS-2112
Assignment Due Date: 2020-02-03 12:30
Buttons: View Student Submissions, Update Assignment, Delete Assignment

7. The list of assignments under course “2019-NURS-2112” will display Assignment X and Assignment 1.

Section 3.5: Instructor delete assignment

1. Click on the “Delete Assignment” button of “Assignment X” under course 2019-NURS-2112.

The screenshot shows the VeriMarker interface. At the top, there are tabs for Courses, Assignments, Submissions, and User. The user is logged in as Instructor Monica Tang. A modal window titled "Delete Assignment" is open, displaying the message "Successfully deleted the assignment." with an "OK" button. In the background, there is a list of assignments for the course 2019-NURS-2112, including "Assignment 1". Below the assignment list, there is a date input field set to "2020-02-03" and a "Delete Assignment" button.

2. “Assignment X” was successfully deleted in Course 2019-NURS-2112. The instructor will no longer able to view the assignment “Assignment X”.

Section 3.6: Register two new users as Students

1. In VeriMarker, logout the instructor Monica Tang. Login with jsleung206@gmail.com to register the Google account as a Student in VeriMarker:

The screenshot shows a browser window with multiple tabs at the top. One tab is for "Sign in - Google Account". The main content is a "Sign in with Google" page. It displays a "Sign in" form with an "Email or phone" field containing "jsleung206@gmail.com". Below the form are links for "Forgot email?" and "Create account". To the right of the form is a blue "Next" button. At the bottom of the page are links for "English (United States)", "Help", "Privacy", and "Terms".

2. Enter the user name as Mary Lee, Email as jsleung206@gmail.com and select the role as Student, and click Create new user:

User Registration - Create new user

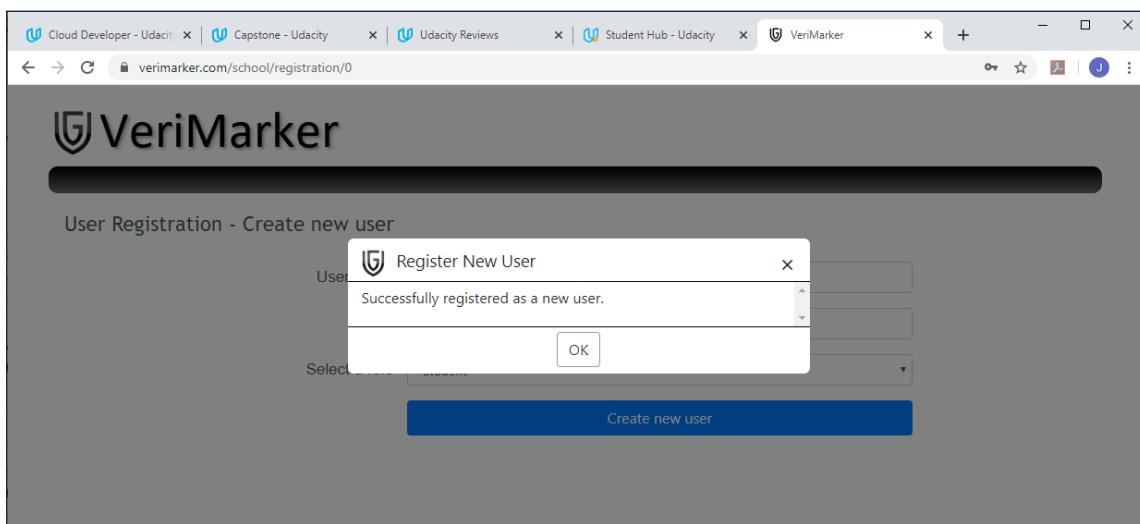
User name: Mary Lee

Email: jsleung206@gmail.com

Select a role: Student

Create new user

3. Enter the user name as Mary Lee, Email as jsleung206@gmail.com and select the role as Student, and click Create new user:



4. After user is successfully registered, the student is navigated to the main menu of VeriMarker:

(continued to next page ...)

The screenshot shows a browser window with five tabs open: "Cloud Developer - Udacity", "Capstone - Udacity", "Udacity Reviews", "Student Hub - Udacity", and "VeriMarker". The "VeriMarker" tab is active, displaying the "verimarker.com/school/main" page. The page features a header with the VeriMarker logo and the user's name, "Mary Lee", along with their email address, "jsleung206@gmail.com", and a note indicating they are logged in as a Student. Below the header is a navigation bar with links for "Submission", "History", and "User", and a "Logout" button. The main content area contains three buttons: "Upload Submission" (with an icon of a document and arrow), "Submission History" (with an icon of a clock), and "User Settings" (with an icon of a person). Each button has a corresponding descriptive text below it.

5. User jsleung206@gmail.com was successfully registered as a student with name Mary Lee. Mary Lee can only access the Student functions in the main menu.
6. In VeriMarker, logout the student Mary Lee. Repeat the above steps to register jsleung106@gmail.com as a new Student account with name “David Liu” in VeriMarker.
7. User jsleung106@gmail.com was successfully registered as a student with name David Liu. David Liu can only access the Student functions in the main menu.
8. Finish this section by logout David Liu in VeriMarker.

Section 3.7: Student “Mary Lee” to upload submission to the course assignment and check her submission history.

1. Login as Mary Lee using jsleung206@gmail.com.
2. Select Upload Submission in the main menu.
3. In the Upload New Submission page shown below, the student selects Instructor “Monica Tang”, Academic Year = 2019, and choose a course “2019-NURS-1151” in the selection drop down:

(continued to next page ...)

AWS Management Console VeriMarker

verimarker.com/school/main/uploadSubmission

Mary Lee
jsleung206@gmail.com
User logged in as: Student

Home Submission History User Logout

Upload New Submission

Select Instructor: Monica Tang

Academic Year: 2019

Choose a course:

- 2019-NURS-2112
- 2019-NURS-1151**

Course description:

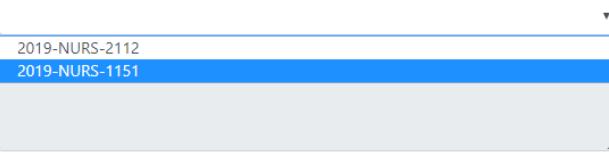
Choose assignment:

Assignment description:

Assignment due date:

Select a file: Choose File No file chosen

Submission References:



4. The correct course description should be displayed after the Course (“2019-NURS-1151”) is selected. Student select Assignment 1 in the Choose Assignment drop down:

(continued to next page ...)

AWS Management Console VeriMarker verimarker.com/school/main/uploadSubmission

VeriMarker

Mary Lee
jsleung206@gmail.com
User logged in as: Student

Submission History User Logout

Upload New Submission

Select Instructor: Monica Tang

Academic Year: 2019

Choose a course: 2019-NURS-1151

Course description: Development of Nursing revised description

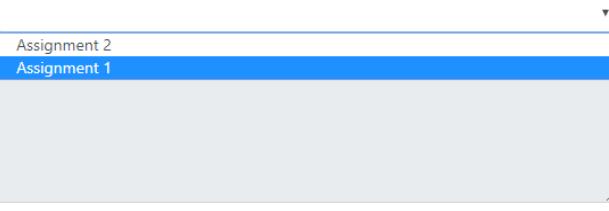
Choose assignment: Assignment 1

Assignment description: Assignment 1

Assignment due date: [Calendar icon] HH : MM

Select a file: Choose File No file chosen

Submission References



- After the student selected “Assignment 1”, the correct due date “2020 January 6, 09:30” should be displayed. Student click on “Choose a file” to select the file “Project2_udagram_jeffrey.docx”. Student must enter the Submission References before able to click on the “Upload Submission” button:

(continued to next page ...)

VeriMarker

Submission History User Logout

Upload New Submission

Select Instructor: Monica Tang

Academic Year: 2019

Choose a course: 2019-NURS-1151

Course description: Development of Nursing revised description

Choose assignment: Assignment 1

Assignment description: Assignment 1 for 2019-NURS-1151

Assignment due date: 2020-01-06 09 : 30

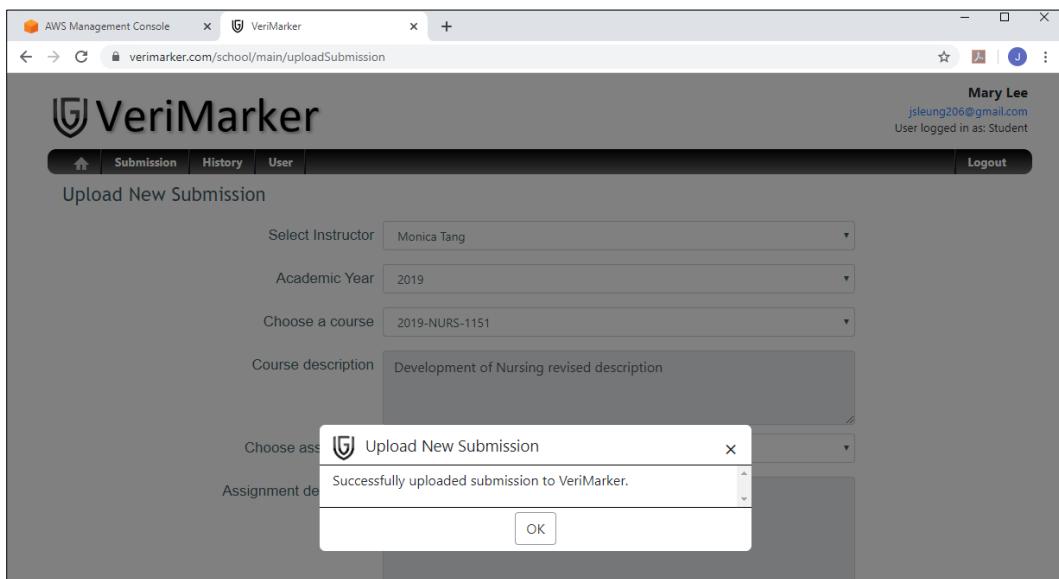
Select a file: Choose File Project2_udagram_jeffrey.docx

Submission References:

1. www.refexample.com: AAWWZZD
2. Book 1st edition, by author APPWSSZD
3. Article from Internet https://www.example.com XXXYZZZ

Upload Submission

6. Student Mary Lee successfully uploaded the submission to VeriMarker.



7. The student will be navigated back to the main menu. Select Submission History in the main menu to verify the submission was uploaded correctly

The screenshot shows the VeriMarker web application. At the top right, it displays the user's name, email (jsleung206@gmail.com), and a message indicating they are logged in as a Student. Below the header is a navigation bar with three tabs: 'Submission' (selected), 'History', and 'User'. Underneath the navigation bar are three main sections: 'Upload Submission' (with a file icon and a placeholder 'To submit your assignment to VeriMarker.'), 'Submission History' (with a clock icon and a placeholder 'View your submissions that were uploaded to VeriMarker.'), and 'User Settings' (with a user icon and a placeholder 'Click here to change user settings').

8. After click on Submission History, it will list the submission that was just uploaded:

The screenshot shows the 'Submission History' page for the student. It features a title 'Viewing My Submissions Upload History'. Below this, there is a card for 'Assignment Name: Assignment 1'. The card contains the following information: 'Uploaded time: 12/22/2019, 6:15:57 PM', 'Student Name: Mary Lee', 'Instructor Name: Monica Tang', 'Similarity Percentage: 12.5 %', 'Submission Score: --%', and a 'Download File' link to 'Project2_udagram_jeffrey.docx'. At the bottom of the card are three buttons: 'Instructor Comments' (blue), 'Submission References' (green), and 'Withdraw Submission' (red).

9. Student click on the file name to start downloading the file. Open the file to verify the file contents are correct:

The screenshot shows the same 'Submission History' page as before, but now with a download progress bar at the bottom of the assignment card. The progress bar indicates the file '89a38731-ad70-4eba-9029-8374877ef381' is being downloaded. The status bar at the bottom of the browser window also shows the download path: '...amazonaws.com/89a38731-ad70-4eba-9029-8374877ef381'. The rest of the page content remains the same, including the assignment details and action buttons.

The screenshot shows a Microsoft Word document with the title "Project 2 – Udagram – by Jeffrey Leung". Below the title, there is a section heading "1. Engineering Process and Quality". Underneath this heading, there is a paragraph of text. At the bottom of the Word window, there is a screenshot of a GitHub repository page for "jsleung1 / cloud-developer". The GitHub page shows basic repository statistics: 33 commits, 2 branches, 0 releases, and 6 contributors. It also shows a list of recent commits, including one from "udacity:master" and another from "jsleung1 / cloud-developer".

Section 3.8: Student “Mary Lee” upload the second submission to the same course and assignment.

1. Repeat step 3.7 with a different file (project_4_rubrics.pdf):

(continued to next page ...)

The screenshot shows a web browser window with the title bar "AWS Management Console" and "VeriMarker". The address bar displays "verimarker.com/school/main/uploadSubmission". On the right side of the header, there is a user profile for "Mary Lee" (jsleung206@gmail.com) with the message "User logged in as: Student" and a "Logout" button.

The main content area is titled "Upload New Submission". It contains the following fields:

- Select Instructor: Monica Tang
- Academic Year: 2019
- Choose a course: 2019-NURS-1151
- Course description: Development of Nursing revised description
- Choose assignment: Assignment 1
- Assignment description: Assignment 1 for 2019-NURS-1151
- Assignment due date: 2020-01-06 09 : 30
- Select a file: Choose File project_4_rubrics.pdf
- Submission References: Ref 1: Internet by cw, part 1... The Thing

A large blue "Upload Submission" button is located at the bottom of the form.

2. The submission history of student Mary Lee will display two submissions uploaded to Assignment 1:

(continued to next page ...)

Mary Lee
jsleung206@gmail.com
User logged in as: Student

Submission History User Logout

Viewing My Submissions Upload History

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:22:03 PM Student Name: Mary Lee Instructor Name: Monica Tang

Similarity Percentage: 41.9 % Submission Score: --% Download File: project_4_rubrics.pdf

+ Instructor Comments + Submission References Withdraw Submission

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:15:57 PM Student Name: Mary Lee Instructor Name: Monica Tang

Similarity Percentage: 12.5 % Submission Score: --% Download File: Project2_udagram_jeffrey.docx

+ Instructor Comments + Submission References Withdraw Submission

Section 3.9: Student “Mary Lee” deleted the submission uploaded in previous section (3.8)

1. In the submission history page, Mary Lee clicks on “Withdraw” submission button of the second submission (which is displayed first as the most recent submission):

Mary Lee
jsleung206@gmail.com
User logged in as: Student

Submission History User Logout

Viewing My Submissions Upload History

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:22:03 PM Student Name: Mary Lee Instructor Name: Monica Tang

Similarity Percentage: 41.9 % Submission Score: --% Download File: project_4_rubrics.pdf

+ Instructor Comments + Submission References Withdraw Submission

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:15:57 PM Student Name: Mary Lee Instructor Name: Monica Tang

Similarity Percentage: 12.5 % Submission Score: --% Download File: Project2_udagram_jeffrey.docx

+ Instructor Comments + Submission References Withdraw Submission

2. Click Yes to delete (withdraw) the submission. The second submission was deleted in VeriMarker, and the student is left with the first submission uploaded to Assignment 1:

The screenshot shows a browser window with the VeriMarker logo at the top. The user is logged in as 'Mary Lee' (jsleung206@gmail.com). In the main content area, there is a modal dialog titled 'Withdraw Submission' with the message 'Successfully withdraw the submission.' An 'OK' button is visible at the bottom right of the dialog. The background shows a submission history page for 'Assignment Name: Assignment 1'.

Section 3.10: Student “David Liu” to upload submission to Assignment 1 for Course “2019-NURS-1151” and check submission history

1. Login as David Liu. Initially, the submission history of David Liu should be empty:

The screenshot shows a browser window with the VeriMarker logo at the top. The user is logged in as 'David Liu' (jsleung106@gmail.com). In the main content area, there is a modal dialog titled 'Withdraw Submission' with the message 'Successfully withdraw the submission.' An 'OK' button is visible at the bottom right of the dialog. The background shows a submission history page for 'Assignment Name: Assignment 1'.

2. Click on Submission in the main menu. In the Upload New Submission page, select Instructor = Monica Tang, Academic Year = 2019, select course = 2019-NURS-1151, choose assignment = Assignment 1, select file = project5_rubrics.docx. Enter the appropriate Submission References and click Upload Submission:

(continued to next page ...)

VeriMarker

Submission History User Logout

Upload New Submission

Select Instructor: Monica Tang

Academic Year: 2019

Choose a course: 2019-NURS-1151

Course description: Development of Nursing revised description

Choose assignment: Assignment 1

Assignment description: Assignment 1 for 2019-NURS-1151

Assignment due date: 2020-01-06 09:30

Select a file: Choose File project5_rubrics.docx

Submission References: Ref 1: stackoverflow, please refer to XXYYZZ using method...

Upload Submission

- After the submission was successfully uploaded to VeriMarker, the submission history of David Liu will show the uploaded submission:

Viewing My Submissions Upload History

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:31:03 PM **Student Name:** David Liu **Instructor Name:** Monica Tang

Similarity Percentage: 46.64 % **Submission Score:** --% **Download File:** project5_rubrics.docx

+ Instructor Comments + Submission References - Withdraw Submission

- Click on the file name to download the file and verify the contents are correct:

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:31:03 PM Student Name: David Liu Instructor Name: Monica Tang

Similarity Percentage: 46.64 % Submission Score: 89% Download File: [project5_rubrics.docx](#)

[Instructor Comments](#) [Submission References](#) [Withdraw Submission](#)

Serverless Application

Rubrics:

<https://review.udacity.com/#!/rubrics/2574/view>

Please use the following settings in the config of the client in order to test our serverless application:

```
const apiId = 'qbasdfm88e'

export const apiEndpoint = `https://${apiId}.execute-api.us-east-1.amazonaws.com/dev`

export const authConfig = {
  domain: 'dev-clq116aa.auth0.com',
  clientId: 'j89rVwvRORYSRJbxWAGRCWk25TeB8Fd0',
  callbackUrl: 'http://localhost:3000/callback'
}
```

Path to download the certification for authorization using Auth0 and JWT:

<https://dev-clq116aa.auth0.com/.well-known/jwks.json>

1. Functionality

1.1 The application allows users to create, update, delete TODO items

1. Please git clone https://github.com/jleung1/serverless_application.git
2. In the client folder, execute: `npm run start`
3. Logged in using Gmail account (supported by <https://auth0.com/>)
4. User should able to create, update (by placing the "checkmark" of the TODO item to mark the item as Done) and delete TODO items.

1.2 The application allows users to upload a file.

Section 3.11: Instructor to check on student submissions uploaded to assignment 1 in section 3.7 to 3.10.

1. Login instructor “Monica Tang” (parispurchase@gmail.com) and Select My Courses in the main menu.
2. This will bring up the courses main page and display two courses that were created by Monica Tang. Select the course 2019-NURS-1151 and click Open:

(continued to next page ...)

VeriMarker

Monica Tang
parispurchase@gmail.com
User logged in as: Instructor

Courses | Assignments | Submissions | User | Logout

Filter Course by Academic Year All

+ Click [here] to create a New Course

2019-NURS-2112

Course Description:
Disaster Nursing

Open Update Delete

2019-NURS-1151

Course Description:
Development of Nursing revised description

Open Update Delete

3. This will bring up the assignments page for Course 2019-NURS-1151. Select Assignment 1 and click View Student Submissions:

VeriMarker

Monica Tang
parispurchase@gmail.com
User logged in as: Instructor

Courses | Assignments | Submissions | User | Logout

Viewing Assignments for Course 2019-NURS-1151

+ Click [here] to create a New Assignment

Assignment 2

Assignment Description:
Assignment 2 for 2019-NURS-1151

Assignment Due Date:
2020-01-24

View Student Submissions | Update Assignment | Delete Assignment

Assignment 1

Assignment Description:
Assignment 1 for 2019-NURS-1151

Assignment Due Date:
2020-01-06

View Student Submissions | Update Assignment | Delete Assignment

4. This will bring up the student submissions for Assignment 1 (in course 2019-NURS-1151). The following verifies that the submission of Mary Lee and David Liu were uploaded correctly to Instructor Monica Tang.

Viewing Student Submissions uploaded to Assignment 1

Assignment Name: Assignment 1

Student Name: David Liu **Instructor Name:** Monica Tang

Similarity Percentage: 46.64 % **Submission Score:**

Download File: project5_rubrics.docx

Instructor Comments | **Submission References** | **Update Student Score**

Assignment Name: Assignment 1

Student Name: Mary Lee **Instructor Name:** Monica Tang

Similarity Percentage: 12.5 % **Submission Score:**

Download File: Project2_udagram_jeffrey.docx

Instructor Comments | **Submission References** | **Update Student Score**

- Click on the file name on both submissions should download the files uploaded by Mary Lee and David Liu. Verify the content of the files from both submissions are correct:

Viewing Student Submissions uploaded to Assignment 1

Assignment Name: Assignment 1

Student Name: David Liu **Instructor Name:** Monica Tang

Similarity Percentage: 46.64 % **Submission Score:**

Download File: project5_rubrics.docx

Instructor Comments | **Submission References** | **Update Student Score**

Assignment Name: Assignment 1

Student Name: Mary Lee **Instructor Name:** Monica Tang

Similarity Percentage: 12.5 % **Submission Score:**

Download File: Project2_udagram_jeffrey.docx

Instructor Comments | **Submission References** | **Update Student Score**

https://verimarker-submissions-files-v2.s3.amazonaws.com/3a4f18eb-07e1-4a18-a5c2-bc390a2bbe60 | 89a38731-ad70-....docx

Show all X

Section 3.12: Student update submission references

- Login to VeriMarker using student Mary Lee (isleung206@gmail.com). Click on Submission History in the main menu. Click on the “Submission References” button:

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:15:57 PM **Student Name:** Mary Lee **Instructor Name:** Monica Tang

Similarity Percentage: 12.5 % **Submission Score:** --% **Download File:** Project2_udagram.jeffrey.docx

References:

1. [www.refexample.com: AAWWZZD](#)
2. Book 1st edition, by author APPWSSZD
3. Article from Internet [https://www.example.com XXXXYZDD](#)
4. Add another reference

Update References

- Add the last line ("4. Add another reference") and click on Update References button. The student's submission is updated with the latest References.

Assignment Name: Assignment 1

Uploaded time: 12/22/2019, 6:15:57 PM **Similarity Percentage:** 12.5 %

References:

1. [www.refexample.com: AAWWZZD](#)
2. Book 1st edition, by author APPWSSZD
3. Article from Internet [https://www.example.com XXXXYZDD](#)
4. Add another reference

- Login as Instructor Monica Tang, go to My Courses, select 2019-NURS-1151. Click on the View Student Submissions button of Assignment 1.

The screenshot shows the VeriMarker interface. At the top right, the user is logged in as 'Monica Tang' (parispurchase@gmail.com) and is an 'Instructor'. The main content area displays two student submissions for 'Assignment 1'. Each submission card includes the assignment name, upload date, student name, instructor name, similarity percentage, submission score (input field), download link, and three buttons: 'Instructor Comments', 'Submission References', and 'Update Student Score'. The second submission card also lists 'References' with four entries: 1. www.refexample.com: AAWWZZD, 2. Book 1st edition, by author APPWSSZD, 3. Article from Internet https://www.example.com XXXYZZZ, and 4. Add another reference.

4. In the list of assignments, click on Submission References of Mary Lee. The instructor will be able to read the updated reference (last line as “4. Add another reference”).

(continued to next page ...)

Section 3.13: Instructor add comments and score to the student's submissions.

1. Continued from section 3.12, Instructor Monica Tang will add comments and assign score to the submissions of Mary Lee and David Liu.
2. To update David Liu's submission, instructor first assign 89 as the submission score and click on "Update Student Score". Then, Instructor click on the "Instructor Comments" button and add the comments "Instructor comments for David Liu". Click on the "Update Comments" button to update the instructor comments for the submission:

The screenshot shows the VeriMarker interface with two assignment submissions listed. Both assignments are named "Assignment 1".

Submission 1 (David Liu):

- Uploaded time: 12/22/2019, 6:31:03 PM
- Student Name: David Liu
- Instructor Name: Monica Tang
- Similarity Percentage: **46.64 %**
- Submission Score:
- Download File: [project5_rubrics.docx](#)
- Buttons: Instructor Comments (highlighted), Submission References, Update Student Score
- Instructor comments for David Liu:
Instructor comments for David Liu
- Update Comments button

Submission 2 (Mary Lee):

- Uploaded time: 12/22/2019, 6:15:57 PM
- Student Name: Mary Lee
- Instructor Name: Monica Tang
- Similarity Percentage: **12.5 %**
- Submission Score:
- Download File: [Project2_udagram_jeffrey.docx](#)
- Buttons: Instructor Comments, Submission References, Update Student Score

(continued to next page ...)

3. Similarly, to update Mary Lee's submission, instructor first assign 62 as the submission score and click on "Update Student Score". Then, Instructor click on the "Instructor Comments" button and add the comments "Instructor comments for Mary Lee". Click on the "Update Comments" button to update the instructor comments for the submission:

The screenshot shows the VeriMarker interface. At the top, it displays the assignment name "Assignment 1". Below this, there are two entries for "Assignment Name: Assignment 1". The first entry is for "Student Name: David Liu" with "Similarity Percentage: 46.64 %", "Submission Score: 89", and "Instructor Name: Monica Tang". The second entry is for "Student Name: Mary Lee" with "Similarity Percentage: 12.5 %", "Submission Score: 62", and "Instructor Name: Monica Tang". Both entries have a "Download File" link. Below each entry are three buttons: "Instructor Comments", "Submission References", and "Update Student Score". In the "Mary Lee" entry, the "Instructor Comments" button is highlighted. A text area below it contains the comment "Instructor comments for Mary Lee". At the bottom of this section is a blue "Update Comments" button.

4. Logged in as student David Liu. In the submission history of David Liu, the only one submission in Submission History should display submission score 89 and instructor comments "Instructor comments for David Liu":

(continued to next page ...)

The screenshot shows the VeriMarker web application interface. At the top right, the user is identified as "David Liu" with the email "jsleung106@gmail.com" and the note "User logged in as: Student". The main navigation bar includes links for Home, Submission, History, User, and Logout. Below the navigation, a section titled "Viewing My Submissions Upload History" displays a single submission entry for "Assignment Name: Assignment 1". The submission details are: Uploaded time: 12/22/2019, 6:31:03 PM; Student Name: David Liu; Instructor Name: Monica Tang; Similarity Percentage: 46.64 %; Submission Score: 89%; and a Download File link for "project5_rubrics.docx". Below these details are three buttons: "Instructor Comments" (blue), "Submission References" (green), and "Withdraw Submission" (red). A large text area labeled "Instructor comments for David Liu" contains the placeholder text "Instructor comments for David Liu".

5. Similarly, logged in as student Mary Lee. In the submission history of Mary Lee, the only one submission in Submission History should display submission score 62 and instructor comments “Instructor comments for Mary Lee”:

The screenshot shows the VeriMarker web application interface. At the top right, the user is identified as "Mary Lee" with the email "jsleung206@gmail.com" and the note "User logged in as: Student". The main navigation bar includes links for Home, Submission, History, User, and Logout. Below the navigation, a section titled "Viewing My Submissions Upload History" displays a single submission entry for "Assignment Name: Assignment 1". The submission details are: Uploaded time: 12/22/2019, 6:15:57 PM; Student Name: Mary Lee; Instructor Name: Monica Tang; Similarity Percentage: 12.5 %; Submission Score: 62%; and a Download File link for "Project2_udagram_jeffrey.docx". Below these details are three buttons: "Instructor Comments" (blue), "Submission References" (green), and "Withdraw Submission" (red). A large text area labeled "Instructor comments for Mary Lee" contains the placeholder text "Instructor comments for Mary Lee".

4. Project Rubrics

Project Setup:

Refer to **Option 2** of the Cloud Rubrics for the Capstone Project:

<https://review.udacity.com/#!/rubrics/2578/view>

The project structure of VeriMarker is very similar to the one in Serverless project:

1. “backend” folder – contains serverless and lambda function implementations.
2. “client” folder – contains the code for the Angular front-end.

One particular file is the config.ts (located in client/src/app/config.ts) in the client project. This file contains the parent URL request path to the AWS Lambda functions, and the Auth0 settings such as clientId and callbackUrl. Please note Auth0 settings (authConfig_prod) is commented out, since starting the local client will use a different set of Auth0 settings (authConfig_dev) as indicated in the following:

```
const apiKey = 'lh756iw3bi';
const env = 'v2';

export const apiEndpoint = `https://${apiKey}.execute-api.us-east-1.amazonaws.com/${env}`;
/*
export const authConfig_prod = {
  domain: 'dev-clq116aa.auth0.com',
  clientId: 'QLaMP23u619whbZWhxF6PTF9eKxMSVU8',
  callbackUrl: 'https://www.verimarker.com/school/auth0',
  mode: 'prod'
}
*/
export const authConfig_dev = {
  domain: 'dev-clq116aa.auth0.com',
  clientId: 'Z52r8N5nV8h3XvWr3jqZJm9wE3JWi6yq',
  callbackUrl: 'http://localhost:4200/school/auth0',
  mode: 'dev'
}

export const authConfig = authConfig_dev
```

Path to download the certification for authorization using Auth0 and JWT:

<https://dev-clq116aa.auth0.com/.well-known/jwks.json>

The Postman REST API for VeriMarker can be imported from the file: capstone_project.postman_collection.json (located in the project root directory).

To run the Angular client, please perform the following steps:

1. git clone https://github.com/isleung1/project_capstone_verimarker.git
2. In Visual Studio Code, open the project folder “project_capstone_verimarker”.
3. Open a terminal in Visual Studio code, and cd to the client folder (project_capstone_verimarker/client).
4. In the client folder, install latest Angular by execute: npm install -g @angular/cli (you can try install Angular locally to the project by: npm install @angular/cli , but you need additional setup for this to work).

5. Verify Angular install correctly by Execute: `ng -version` (it should return Angular 8.0.6).
6. Start the Angular 8, execute: `npm run start`
7. In Chrome Browser, go to the Url: <http://localhost:4200>
8. Please use your Gmail account to login VeriMarker:

4.1 Functionality

4.1.1 The application allows users to create, update, delete items

Click on the link in the following table to go through the details of the Unit Test that shows how VeriMarker allows both Instructors and Students to create, update and delete items:

Section	Unit Test Name	User Type
3.2	Instructor create course / update course description	Instructor
3.3	Instructor delete a course	Instructor
3.4	Instructor create assignments	Instructor
3.5	Instructor delete assignment	Instructor
3.9	Student "Mary Lee" deleted the second submission uploaded in previous section (3.8)	Student
3.12	Student update the submission references (submission update)	Student
3.13	Instructor add comments and score to the student's submissions (submission update)	Instructor

4.1.2 The application allows users to upload a file.

Click on the link in the following table to go through the details of the Unit Test that shows how VeriMarker allows Students to upload files and download the file with the correct content:

Section	Unit Test Name	User Type
3.7	Student “Mary Lee” upload submission to the course assignment and check her submission history.	Student
3.8	Student “Mary Lee” upload the second submission to the same course and assignment.	Student
3.10	Student “David Liu” to upload submission to Assignment 1 for Course “2019-NURS-1151”.	Student
3.11	Instructor to check on student submissions uploaded to assignment 1 in section 3.7 to 3.10.	Instructor

4.1.3 The application only displays items for a logged in user.

Click on the link in the following table to go through the details of the Unit Test that shows how VeriMarker only display instructor own courses and assignments, and student only view their own submissions:

Section	Unit Test Name	User Type
3.2	Instructor create course / update course description (only display list of courses created by the instructor)	Instructor
3.4	Instructor create assignments (only display list of assignments created by the instructor)	Instructor
3.7	Student “Mary Lee” upload submission to the course assignment and check her submission history (submission history only showed her own uploaded submission).	Student
3.8	Student “Mary Lee” upload the second submission to the same course and assignment (submission history only showed her own uploaded submissions).	Student
3.10	Student “David Liu” to upload submission to Assignment 1 for Course “2019-NURS-1151” and check submission history (submission history only showed his own uploaded submission).	Student

Please also refer to one of the data access layers (located in backend/src/dataLayer) to query user items.

In the follow example, when the student clicks on Submission History, submissionAccess.ts performs a query of Submission items using the student Id (which is the value of the decodedJwt.sub in the JwtToken):

```
async getAllSubmissionsByStudentId(studentId: string): Promise<Submission[]> {

    const result = await this.docClient.query({
        TableName: this.submissionsTable,
        IndexName: this.submissionsStudentIdIndex,
        KeyConditionExpression: 'studentId = :studentId',
        ExpressionAttributeValues: {
            ':studentId': studentId
        },
    },
```

```

        ScanIndexForward: false
    }).promise()

    const items = result.Items
    return items as Submission[]
}

```

Another example is when the Instructor clicks on “My Courses” in the main menu. The courseAccess.ts performs a query of Courses items using the instructor Id (which is the value of the decodedJwt.sub in the JwtToken):

```

async getAllCoursesByInstructorId(instructorId: string): Promise<Course[]> {

    const result = await this.docClient.query({
        TableName: this.coursesTable,
        IndexName: this.coursesInstructorIdIndex,
        KeyConditionExpression: 'instructorId = :instructorId',
        ExpressionAttributeValues: {
            ':instructorId': instructorId
        },
        ScanIndexForward: false
    }).promise()

    const items = result.Items
    return items as Course[]
}

```

4.1.4 Authentication is implemented and does not allow unauthenticated access.

All of the AWS Lambda functions defined in serverless.yml are declared with the authorizer Auth0, which is the auth0Authorizer.ts (located in backend/src/lambda/auth). The bulk of the work in auth0Authorizer.ts is call the verifyToken function:

```

async function verifyToken(authHeader: string): Promise<JwtPayload> {
    const token = getToken(authHeader)
    const cert = await getJksCert( jwksUrl )
    return verify(token, cert, { algorithms: ['RS256'] }) as JwtPayload
}

```

First, it gets the token from the authorization header. Then, it calls async method getJksCert with url path <https://dev-clq116aa.auth0.com/.well-known/jwks.json>. This will download the certificate to verify the incoming token from the request header, such as whether the token already expired, or simply an invalid token.

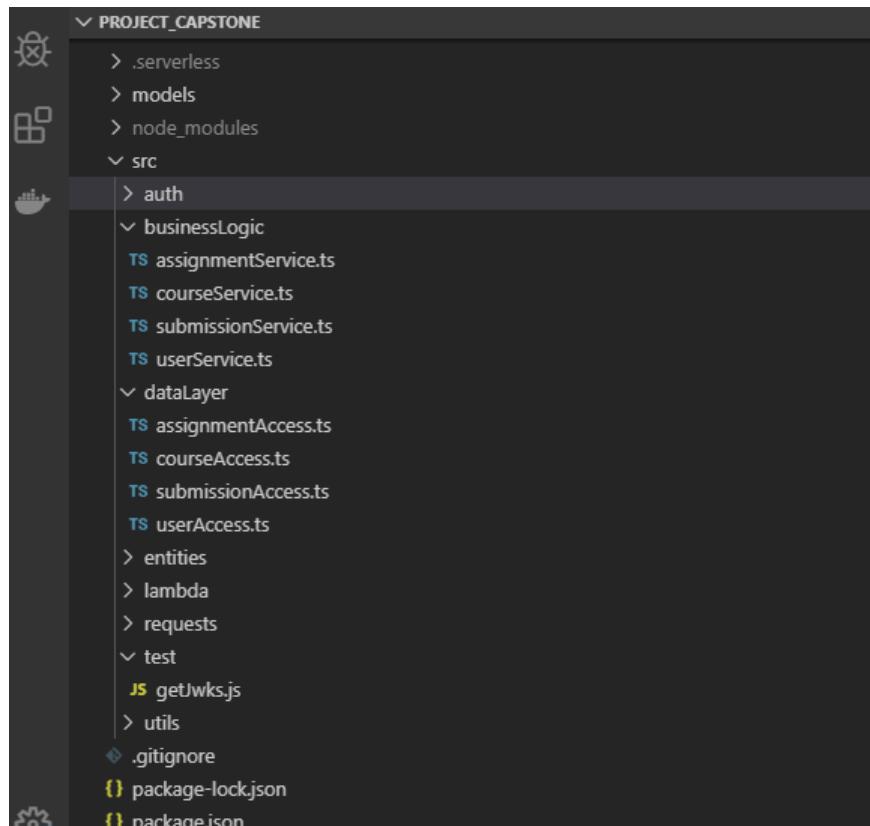
To try it out, in Postman, using the wrong jwtToken value or missing Authorization header will result in a HTTP respond with error code 403 Forbidden as shown in the following example:

The screenshot shows the Postman application interface. On the left, the 'Collections' tab is selected, displaying a list of collections including 'udacity-c2-restapi' (10 requests), 'Capstone Project: VeriMarker' (21 requests), 'Users', 'Courses', 'Test', and others. A specific request, 'Get courses (student)', is highlighted. The main panel shows the request details: method 'GET', URL 'https://{{apilid}}.execute-api.us-east-1.amazonaws.com/{{env}}/courses/2019', and the 'Headers' tab selected. Under 'Headers (8)', there is one entry: 'Authorization' with value 'Bearer invalidTokenValue'. Below this, 'Temporary Headers (7)' is listed. The 'Body' tab is selected, showing a JSON response with a single key 'message': "User is not authorized to access this resource with an explicit deny".

4.2. Code Base

4.2.1 The code is split into multiple layers separating business logic from I/O related code.

Refer to the code structure in the /backend folder, the code is split into businessLogic and dataLayer (I/O) layers:



The following table summarized the functions of the business logic and data layer (I/O) in VeriMarker:

Business Logic (src/businessLogic)	Description
courseService.ts	<ul style="list-style-type: none"> - Check the userType. If the user is an Instructor, get the courses created by that instructor for that academic year. If the user is a Student, get courses of ALL instructors for that academic year. - Check the userType is an Instructor before allowing to create, update and delete Course. Also, check if the instructor is the course owner before allowing to update and delete the course. - Check to ensure no assignments are created under the course before the Instructor can delete the course.
assignmentService.ts	<ul style="list-style-type: none"> - Get all the assignments of the course for the student. For the instructor, check if the instructor is the course creator before return all the assignments of the course. - Check the userType is an Instructor before allowing to create, update or delete the assignment. <p>Check to ensure no submissions are uploaded to the assignment before the Instructor can delete the assignment.</p>
submissionService.ts	<ul style="list-style-type: none"> - Get all submissions uploaded by the student. For the instructor, return only the submissions uploaded to the Instructor by the student. - Check the user is a student before allowing to delete the submission, or update the submission references. - Check the user is an instructor before allowing to update the submission score and instructor's comments.
userService.ts	<ul style="list-style-type: none"> - Check if the user is registered in VeriMarker by using the userId which is the value of the decodedJwt.sub in the JwtToken. If the user is not registered in VeriMarker, returns an empty JSON user to the Angular client. - Create a new registered user in VeriMarker. - Allows user to update the email and user Type (Student or Instructor).

DataLayer (I/O) (src/dataLayer)	Description
courseAccess.ts	<ul style="list-style-type: none"> - Enable AWS Ray X-Ray. - Performs creation of item in DynamoDb using docClient.put
assignmentAccess.ts	<ul style="list-style-type: none"> - Performs query of DynamoDb using docClient.query, using the appropriate Indexes. - Performs update of item in DynamoDb using docClient.update, using the appropriate UpdateExpression and ExpressionAttributeValues.
submissionAccess.ts	<ul style="list-style-type: none"> - Performs delete of Course item in DynamoDb using docClient.delete
userAccess.ts	<ul style="list-style-type: none"> - All queried items are sorted by returning the most recent created item first (using ScanIndexForward: false on the range key "CreatedAt").

4.2.2 Code is implemented using async/await and Promises without using callbacks.

Please check all the code in the backend folder. It contains code using async/await and Promises without using callbacks.

4.3. Best Practices

4.3.1 All resources in the application are defined in the "serverless.yml" file

All resources are defined in backend/serverless.yml . Please check it out.

4.3.2 Each function has its own set of permissions.

In the serverless.yml file, under plugins, the “serverless-iam-roles-per-function” was included. Please check the serverless.yml which shows all the Lambda functions has its own iamRoleStatements. The following is a snippet of the serverless.yml, which shows the iamRoleStatements for the GetUserSubmissions Lambda function:

```
 GetUserSubmissions:
  handler: src/lambda/http/submission/getUserSubmissions.handler
  events:
    - http:
        method: get
        path: submissions
        cors: true
        authorizer: Auth0
  iamRoleStatements:
    - Effect: Allow
      Action:
        - dynamodb:Query
      Resource: arn:aws:dynamodb:${self:provider.region}:*:table/${self:provider.environment.USERS_TABLE}
    - Effect: Allow
      Action:
        - dynamodb:Query
      Resource: arn:aws:dynamodb:${self:provider.region}:*:table/${self:provider.environment.USERS_TABLE}/index/${self:provider.environment.USERS_USERID_INDEX}
    - Effect: Allow
      Action:
        - dynamodb:Query
      Resource: arn:aws:dynamodb:${self:provider.region}:*:table/${self:provider.environment.USUBMISSIONS_TABLE}/index/${self:provider.environment.SUBMISSIONS_STUDENTID_INDEX}
    - Effect: Allow
      Action:
        - dynamodb:Query
      Resource: arn:aws:dynamodb:${self:provider.region}:*:table/${self:provider.environment.USUBMISSIONS_TABLE}/index/${self:provider.environment.SUBMISSIONS_INSTRUCTORID_INDEX}
    - Effect: Allow
      Action:
        - dynamodb:Query
```

```

Resource: arn:aws:dynamodb:${self:provider.region}::table/${self:provider.environment.SUBMISSIONS_TABLE}
- Effect: Allow
Action:
- xray:PutTraceSegments
- xray:PutTelemetryRecords
Resource: "*"

```

4.3.3 Application has sufficient monitoring.

- Distributed tracing is enabled

Using AWS X-Ray, distributed tracing is enabled by define the following in serverless.yml:

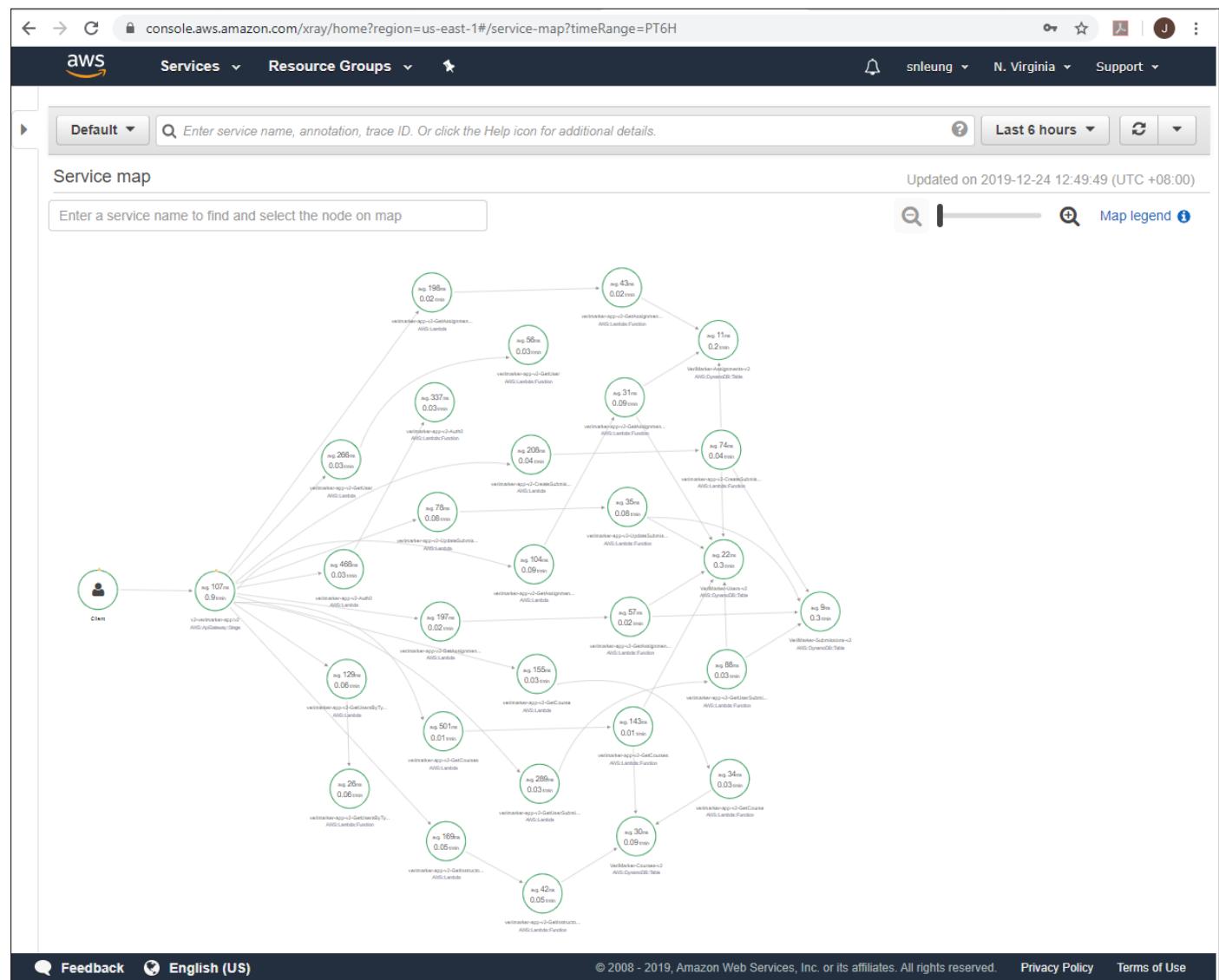
```
provider:
```

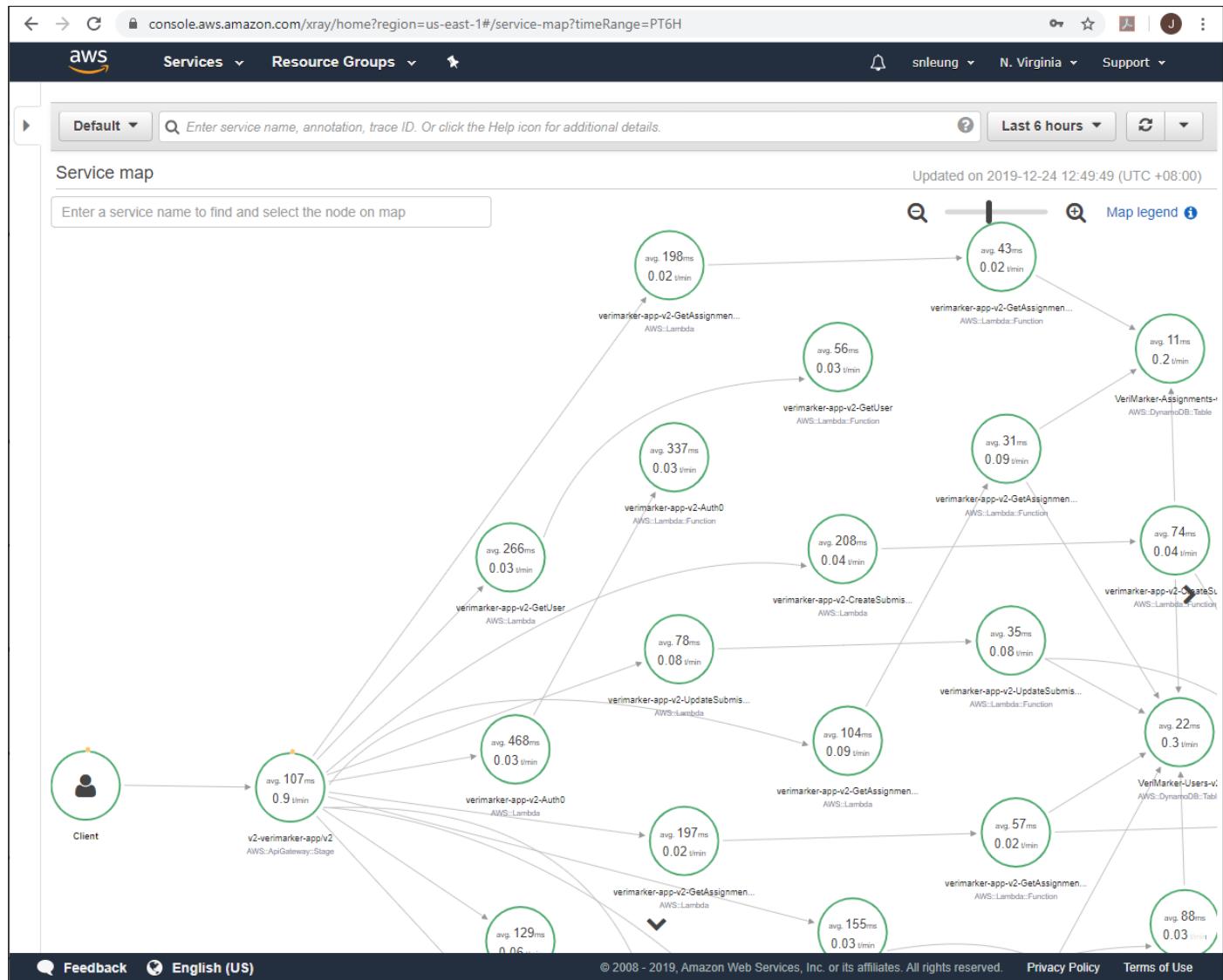
```

.....
tracing:
  lambda: true
  apiGateway: true

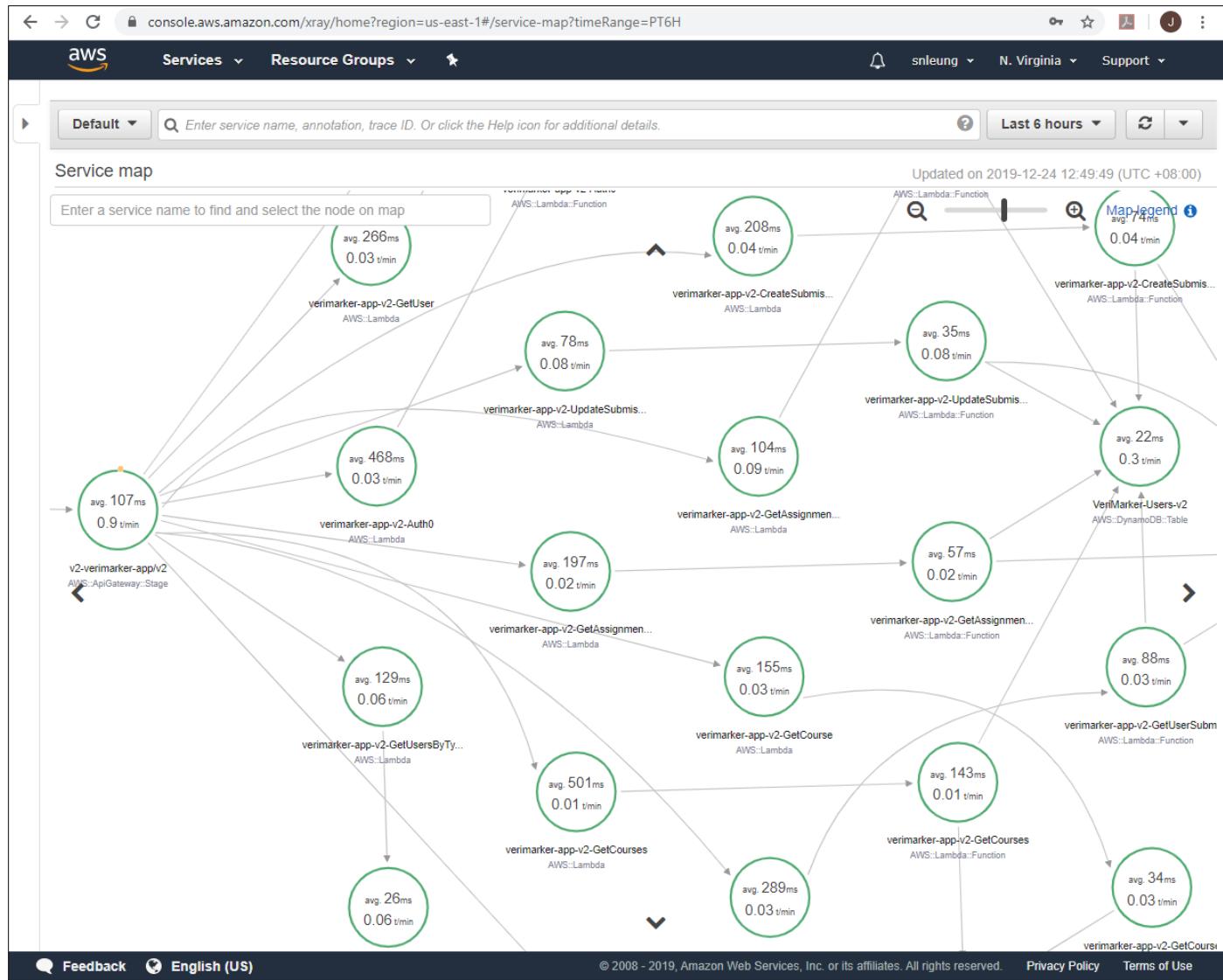
```

In the AWS Console, under AWS X-Ray, it shows the following trace in the following diagrams:





(continued to next page ...)



(continued to next page ...)

- It has a sufficient amount of log statements

Our application uses the Winston logger which outputs the logs to AWS CloudWatch:

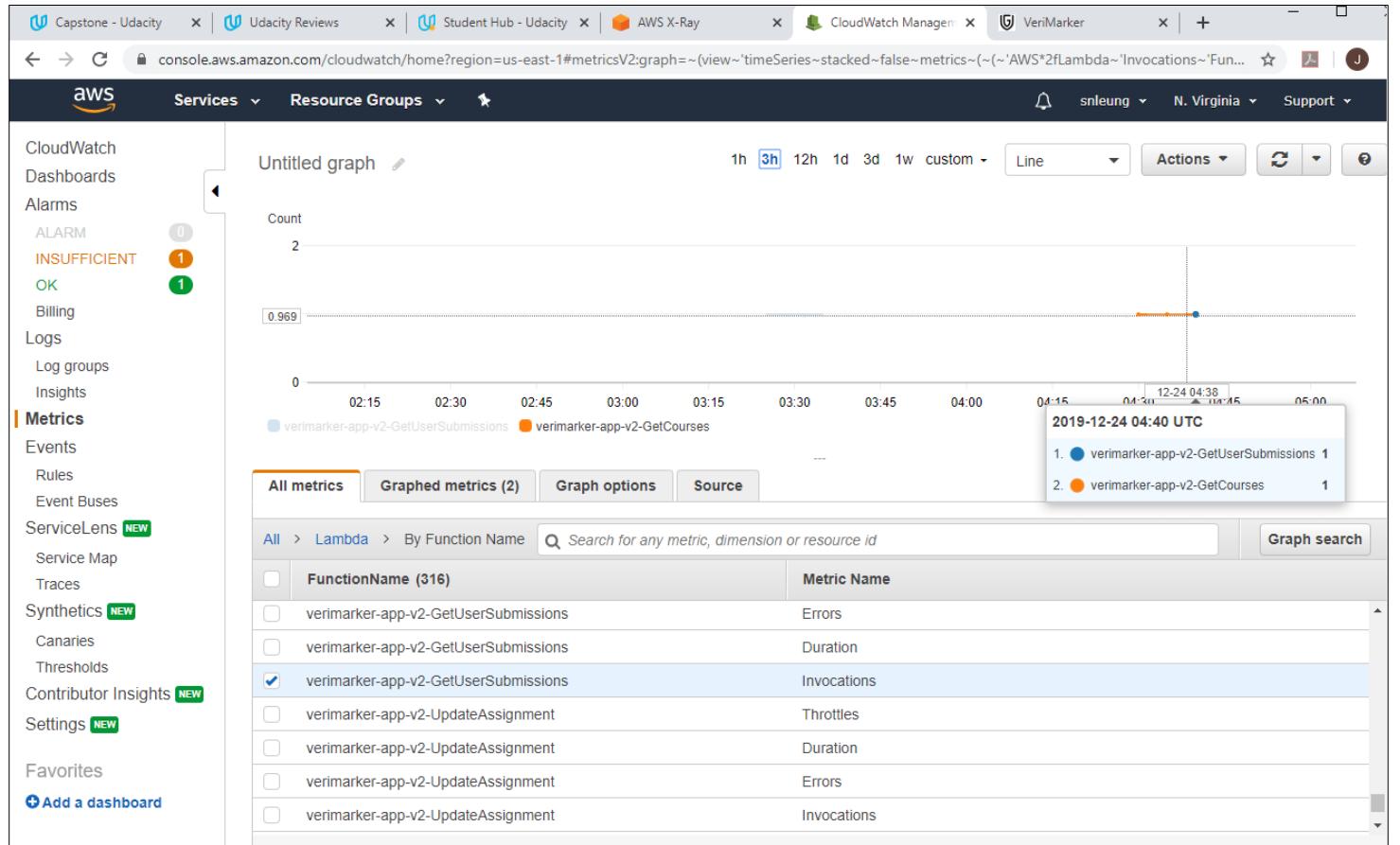
Screenshot of the AWS CloudWatch Logs console showing a list of log streams for the VeriMarker Lambda application. The logs are organized by path, and each entry includes options to 'Explore' or 'Edit' the log stream.

Log Stream Path	Action	Retention	Filters	Region
/aws/lambda/verimarker-app-v2-Auth0	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-CreateAssignment	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-CreateCourse	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-CreateSubmission	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-CreateUser	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-DeleteAssignment	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-DeleteCourse	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-DeleteSubmission	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetAssignment	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetAssignmentSubmissions	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetAssignments	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetCourse	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetCourses	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetInstructorCourses	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetSubmission	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetUser	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetUserSubmissions	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-GetUsersByType	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-UpdateAssignment	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-UpdateCourse	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-UpdateSubmission	Explore	Never Expire	0 filters	None
/aws/lambda/verimarker-app-v2-UpdateUser	Explore	Never Expire	0 filters	None

For example, for log group “/aws/lambda/verimarker-app-v2-CreateSubmission2”, the following indicate the submission was created successfully (logged by submissionService). Also, note the assignment Id of this submission record (the student must select the assignment first before able to upload the submission to VeriMarker).

- It generates application level metrics

Under AWS CloudWatch, the application metrics of VeriMarker can be found by going to Metrics → Lambda → By Function Name. The following example selects Metric Name “Invocations” for FunctionName “verimarker-app-v2-GetUserSubmissions” and “verimarker-app-v2-GetCourses”:



(continued to next page ...)

4.3.4 HTTP requests are validated

To reduce the number of unwanted Lambda function calls, incoming HTTP requests are validated using request validation in API Gateway. In serverless.yml:

```
custom:
  dynamodb:
    stages:
      - v2
    start:
      port: 8000
      inMemory: true
      migrate: true
  webpack:
    webpackConfig: ./webpack.config.js
    includeModules: true
  documentation:
    api:
      info:
        version: v1.0.0
        title: VeriMarker API
        description: Serverless application for VeriMarker, a plagiarism detection application for managing student's submissions
    models:
      - name: CreateUserRequest
        contentType: application/json
        schema: ${file(models/user/create-user-request.json)}
      - name: UpdateUserRequest
        contentType: application/json
        schema: ${file(models/user/update-user-request.json)}
      - name: CreateCourseRequest
        contentType: application/json
        schema: ${file(models/course/create-course-request.json)}
      - name: UpdateCourseRequest
        contentType: application/json
        schema: ${file(models/course/update-course-request.json)}
      - name: CreateAssignmentRequest
        contentType: application/json
        schema: ${file(models/assignment/create-assignment-request.json)}
      - name: UpdateAssignmentRequest
        contentType: application/json
        schema: ${file(models/assignment/update-assignment-request.json)}
      - name: CreateSubmissionRequest
        contentType: application/json
        schema: ${file(models/submission/create-submission-request.json)}
      - name: UpdateSubmissionRequest
        contentType: application/json
        schema: ${file(models/submission/update-submission-request.json)}
```

All of the lambda functions with method post or patch will have the RequestBodyValidator and requestModels defined. For example, in CreateSubmission (which is used by the student to upload submission):

```
CreateSubmission:  
  handler: src/lambda/http/submission/createSubmission.handler  
  events:  
    - http:  
        method: post  
        path: submissions  
        cors: true  
        authorizer: Auth0  
        reqValidatorName: RequestBodyValidator  
        documentation:  
          summary: Create a new submission for the student  
          description: Create a new submission for the student  
        requestModels:  
          'application/json': CreateSubmissionRequest
```

And in UpdateSubmission (which is used by the student to update the Submission References and Instructor to assign a score and add instructor comments to the submission):

```
UpdateSubmission:  
  handler: src/lambda/http/submission/updateSubmission.handler  
  events:  
    - http:  
        method: patch  
        path: submissions/{queryId}  
        cors: true  
        authorizer: Auth0  
        reqValidatorName: RequestBodyValidator  
        documentation:  
          summary: Update existing submission of the student  
          description: Update existing submission of the student  
        requestModels:  
          'application/json': UpdateSubmissionRequest
```

For the above two examples, let's take a look at the validation models of create-submission-request.json and update-submission-request.json:

(continued to next page ...)

For the create-submission-request.json, all properties are required (mandatory):

```
{  
  "$schema": "http://json-schema.org/draft-04/schema#",  
  "title": "createSubmissionRequest",  
  "type": "object",  
  "properties": {  
    "assignmentId": {  
      "type": "string"  
    },  
    "fileName": {  
      "type": "string"  
    },  
    "studentReferences": {  
      "type": "string"  
    }  
  },  
  "required": [  
    "assignmentId",  
    "fileName",  
    "studentReferences"  
  ],  
  "additionalProperties": false  
}
```

For the update-submission-request.json, only the studentReferences is required. This is because instructorComments and studentScore are not defined initially when the submission was uploaded by the student (submission score and comments will be added later by the Instructor when the Instructor evaluate the student's submission).

```
{  
  "$schema": "http://json-schema.org/draft-04/schema#",  
  "title": "updateSubmissionRequest",  
  "type": "object",  
  "properties": {  
    "instructorComments": {  
      "type": "string"  
    },  
    "studentScore": {  
      "type": "number"  
    },  
    "studentReferences": {  
      "type": "string"  
    }  
  },  
  "required": [  
    "studentReferences"  
  ],  
  "additionalProperties": false  
}
```

In the first example, creating a submission request without the assignment Id will result in a response with code 400 Bad Request and message "Invalid request body":

The screenshot shows the Postman application interface. On the left, the sidebar lists collections and APIs. Under the 'Submissions' collection, there is a POST request named 'Create submission'. The request URL is set to `https://{{apilid}}.execute-api.us-east-1.amazonaws.com/{{env}}/submissions`. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   "fileName": "sampleSubmission3.pdf",  
3   "studentReferences": "Reference 4.10"  
4 }  
5 }
```

Below the request details, the response section shows the status as '400 Bad Request' with a message: "message": "Invalid request body".

(continued to next page ...)

In the second example, sending a PATCH request with only the studentReferences will successfully update the submission item without errors:

The screenshot shows the Postman interface with the following details:

- Collection:** udacity-c2-restapi
- Request Type:** PATCH
- URL:** https://{{apild}}.execute-api.us-east-1.amazonaws.com/{{env}}/submissions/14b5bf5e-a800-43db-9ed1-3a81c4456a6c
- Body (JSON):**

```
1 {
2     "studentReferences": "updated reference for citation 3, 4"
3 }
```
- Response Status:** 200 OK
- Response Body (Pretty):**

```
1
2     "assignmentId": "9b0816b6-8831-471a-bd78-e721f494ba6d",
3     "fileName": "project_capstone_rubrics_verimarker.pdf",
4     "similarityPercentage": 51.49,
5     "studentId": "google-oauth2-104786078256352731553",
6     "instructorEmail": "udacityteacher306@gmail.com",
7     "studentName": "Udacity Student 1",
8     "submissionFileUrl": "https://verimarker-submissions-files-v2.s3.amazonaws.com/14b5bf5e-a800-43db-9ed1-3a81c4456a6c",
9     "createdAt": "2019-12-24T05:54:45.515Z",
10    "studentReferences": "updated reference for citation 3, 4",
11    "studentScore": 89,
12    "instructorName": "Udacity Teacher",
13    "studentEmail": "udacitystudent106@gmail.com",
14    "instructorId": "google-oauth2-115227087163046503604",
15    "assignmentName": "Assignment 3",
16    "submissionUploadUrl": "https://verimarker-submissions-files-v2.s3.amazonaws.com/14b5bf5e-a800-43db-9ed1-3a81c4456a6c?
X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIA4UR3N7PEG27UEW0I%2F20191224%2Fus-east-1%2Fs3%2Faws4_request&
X-Amz-Date=20191224T055445Z&X-Amz-Expires=3008
```

However, define a wrong attribute name in the patch update submission request will result in a response with code 400 Bad Request and message “Invalid request body”:

The screenshot shows the Postman interface with the following details:

- Collection:** udacity-c2-restapi
- Request Type:** PATCH
- URL:** https://{{apild}}.execute-api.us-east-1.amazonaws.com/{{env}}/submissions/14b5bf5e-a800-43db-9ed1-3a81c4456a6c
- Body (JSON):**

```
1 {
2     "wrongAttribute": "updated reference for citation 3, 4"
3 }
```
- Response Status:** 400 Bad Request
- Response Body (Pretty):**

```
1
2     "message": "Invalid request body"
3 
```

4.4 Architecture

4.4.1 Data is stored in a table with a composite key.

1:M (1 to many) relationship between users and items is modeled using a DynamoDB table that has a composite key with both partition and sort keys. VeriMarker is heavily built upon the one to many (1:M) relationship exists in DynamoDB:

- **One Assignment to Many Submissions**
- **One Student to Many submissions**
- **One Instructor to Many Courses**
- **One Course to Many Assignments**

Please checkout the table definitions at the end of the serverless file. As an example, in the SubmissionsDynamoDbTable shown below, the submissions can be queried by the assignment Id and the student Id which is defined as the GlobalSecondaryIndexes:

```
SubmissionsDynamoDBTable:  
  Type: "AWS::DynamoDB::Table"  
  Properties:  
    AttributeDefinitions:  
      - AttributeName: submissionId  
        AttributeType: S  
      - AttributeName: assignmentId  
        AttributeType: S  
      - AttributeName: studentId  
        AttributeType: S  
      - AttributeName: instructorId  
        AttributeType: S  
      - AttributeName: createdAt  
        AttributeType: S  
    KeySchema:  
      - AttributeName: submissionId  
        KeyType: HASH  
      - AttributeName: createdAt  
        KeyType: RANGE  
    GlobalSecondaryIndexes:  
      - IndexName: ${self:provider.environment.SUBMISSIONS_SUBMISSIONID_INDEX}  
        KeySchema:  
          - AttributeName: submissionId  
            KeyType: HASH  
          - AttributeName: createdAt  
            KeyType: RANGE  
        Projection:  
          ProjectionType: ALL  
      - IndexName: ${self:provider.environment.SUBMISSIONS_ASSIGNMENTID_INDEX}  
        KeySchema:  
          - AttributeName: assignmentId  
            KeyType: HASH  
          - AttributeName: createdAt
```

```
    KeyType: RANGE
    Projection:
        ProjectionType: ALL
    - IndexName: ${self:provider.environment.SUBMISSIONS_STUDENTID_INDEX}
        KeySchema:
            - AttributeName: studentId
                KeyType: HASH
            - AttributeName: createdAt
                KeyType: RANGE
        Projection:
            ProjectionType: ALL
    - IndexName: ${self:provider.environment.SUBMISSIONS_INSTRUCTORID_INDEX}
        KeySchema:
            - AttributeName: instructorId
                KeyType: HASH
            - AttributeName: createdAt
                KeyType: RANGE
        Projection:
            ProjectionType: ALL
    BillingMode: PAY_PER_REQUEST
    TableName: ${self:provider.environment.SUBMISSIONS_TABLE}
```

4.4.2 Scan operation is not used to read data from a database.

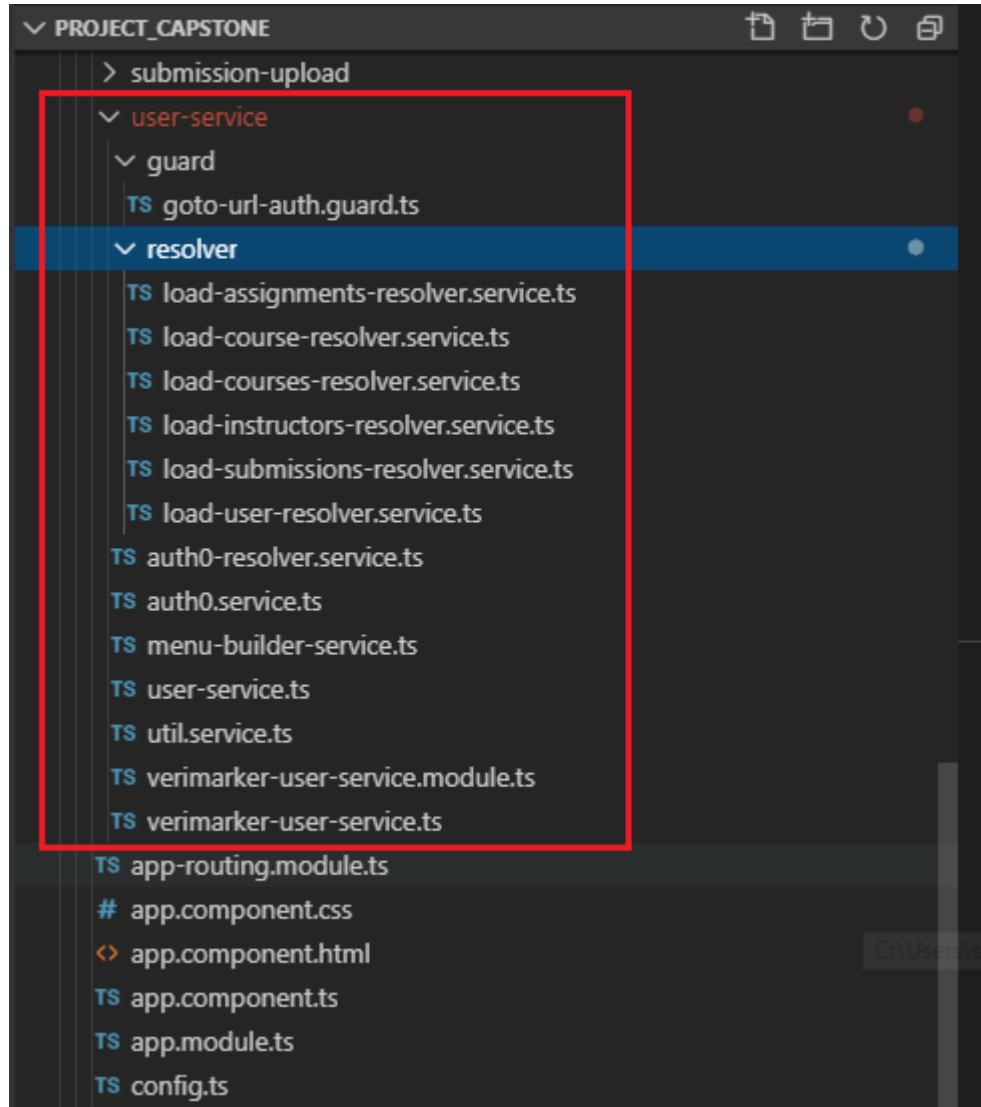
With the use of GlobalSecondaryIndexes , VeriMarker only uses the query method to retrieve records from DynamoDb. No scan method is ever used in the application.

(continued to next page ...)

5. Appendix

5.1 Implementation of the Client

The Client is a webapp developed by using Angular and make heavily use of modules. In particular, the VerimarkerUserServiceModule is particularly important, and the following shows the services included in this module:



As seen from the above, the VeriMarker client app made use of the Angular Resolver (under the resolver folder), which issue HTTP requests to the AWS Lambda functions to get the data before the page is displayed to the user. This prevents the problem of “Flash of unstyled content” (FOUC) commonly found in most SPA applications.

The resolver (auth0-resolver.service.ts) is also used to listen on the callback called by Auth0, in order to determine whether the user should go directly to the Main Page of VeriMarker, or if the user is not registered in VeriMarker, go to the user registration page. Finally, the route guard (goto-url.auth.guard.ts) is used to determine whether the user is logged in (by checking on the user cookie which stored the user session JSON data of LoggedInUser.ts). If the user is already logged in, the user is permitted to go directly to the URL address, and if not, the user is redirected to the Login Page.

5.2 Linking Auth0 JWT token to VeriMarker User Account

It is found that although the idToken (JWT Token) of Auth0 can be different each time the user logged using Auth0, the decrypted JWT token, JwtPayload.sub, is a constant unique value for each Google user. As a result, this piece of information can be used to allow the Auth0 JWT token to link with a VeriMarker user account.

As mentioned before, the auth0-resolver.service.ts (in verimarker-school-login-routing.module.ts) listens on the callback Url after user logged in from Auth0:

```
{  
  // listen path: for dev = http://localhost:4200/school/auth0,  
  // listen path: for prod = 'https://www.verimarker.com/school/auth0'  
  path: verimarkerInjectors.get(URL_PATH_CONFIG).userAuth0CallBackPath.relativePath,  
  resolve: {  
    auth0ResolverService: Auth0ResolverService  
  },  
},
```

In Auth0ResolverService, it delegates to the auth0.service.ts HandleAuthentication(), which obtains the idToken from auth0.parseHash. Using the idToken, the client queries the AWS Lambda function GetUser in order to get the VeriMarker registered user:

```
handleAuthentication() {  
  return this.auth0.parseHash( async (err, authResult) => {  
    if ( authResult && authResult.accessToken && authResult.idToken ) {  
      console.log('Access token: ', authResult.accessToken );  
      console.log('id token: ', authResult.idToken );  
      try {  
        this.spinner.show();  
        const headers = new HttpHeaders({  
          'Content-Type': 'application/json',  
          Authorization: `Bearer ${authResult.idToken}`  
        });  
  
        const user = await this.http.get<User>(`${apiEndpoint}/user`, { headers }).toPromise();  
        this.spinner.hide();  
  
        console.log(user);  
        // user is registered in the system, fetch the user details  
        // and store the user in the cookie as our session cookie  
        if ( user.userId != null ) {  
          this.userService.setLoggedInUser({  
            authenticationState: AuthenticationStateEnum.Authenticated,  
            userName: user.userName,  
            email: user.email,  
            userType: user.userType,  
            accessToken: authResult.accessToken,  
            idToken: authResult.idToken,  
            userId: user.userId  
          });  
          // navigate to the user main page  
        }  
      } catch (e) {  
        console.error(e);  
      }  
    }  
  }  
}
```

```

        this.router.navigate( [ verimarkerInjectors.get(URL_PATH_CONFIG).userMainPage.
fullPath ] );
        .....

```

A non-null value in the userId of the returned User indicates the user is already registered in VeriMarker. Thus, the Angular client will call UserService.setLoggedInUser, which stores the attributes of the returned user and the idToken/accessToken of Auth0 as an JSON object as a cookie:

```

setLoggedInUser(loggedInUser: LoggedInUser ) {

    // cookie duration same as the Auth0 JWT duration
    this.cookieService.set(` ${this.cookieName}-${authConfig.mode}` ,
        JSON.stringify(loggedInUser),
        new Date( new Date().getTime() + 36000 * 1000 ),
        '/');
}

const alreadyLoggedInUserStr = this.cookieService.get(` ${this.cookieName}-`+
${authConfig.mode}` );
console.log('alreadyLoggedInUserStr=' + alreadyLoggedInUserStr);

this.loggedInUserObservable.next( loggedInUser );
}

```

The LoggedInUser.idToken is used by VeriMarkerHttpClient, which is a wrapper of Angular HttpClient. Any HTTP requests with the AWS Lambda functions after user logged in will be done by the VeriMarkerHttpClient, whose primary purpose is to automatically set the idToken in the authorization header for each HTTP request.

Finally, the already registered user is navigated to the main menu.

If the user cannot be found (userId of the returned user from the AWS Get User method is null), the idToken and accessToken are passed to UserService.setRegisterNewUser. The application will redirect the user to the Registration Page which will use the idToken to POST the new user to the AWS Lambda function CreateUser:

```

.....
else {
    // navigate to register new user
    this.userService.setRegisterNewUser({
        authenticationState: AuthenticationStateEnum.NeedToCreate,
        idToken: authResult.idToken,
        accessToken: authResult.accessToken,
        userId: null,
        userType: '',
        email: '',
        userName: ''
    });

    let url = verimarkerInjectors.get(URL_PATH_CONFIG).userRegistrationPage.fullPa
th;
    url = url.replace(':userId', '0');
    this.router.navigate( [ url ] );
}

```

user-registration.component (located in src/app/common-ui/component/user-registration):

```
async createNewUser() {
  const createUserRequest: CreateUserRequest = {
    userType: this.registerUser.userType,
    email: this.registerUser.email,
    userName: this.registerUser.userName
  };

  const headers = new HttpHeaders({
    'Content-Type': 'application/json',
    Authorization: `Bearer ${ this.registerUser.idToken}`
  });

  this.spinner.show();
  try {
    const user = await this.http.post(`/${apiEndpoint}/users`, createUserRequest, { headers })
      .toPromise() as User;
    this.spinner.hide();

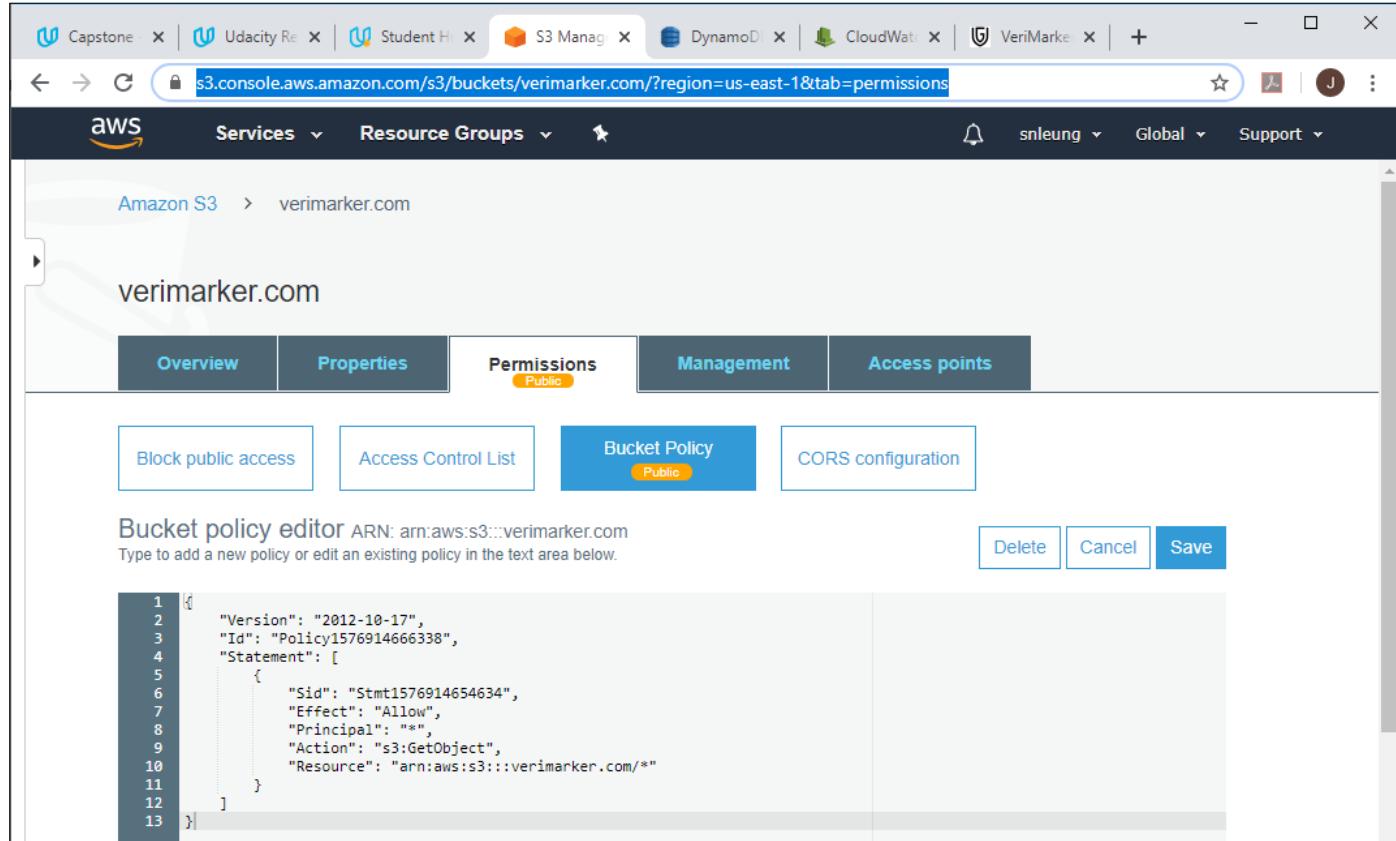
    // for create new user when login to auth0, set the newly created user as our logged in user and create the session cookie
    if ( this.registerUser.authenticationState === AuthenticationStateEnum.NeedToCreate ) {
      this.alertDialogService.openDialog({
        title: 'Register New User',
        message: 'Successfully registered as a new user.',
        dialogType: 'OKDialog'
      }).then( res => {
        this.userService.setLoggedInUser({
          authenticationState: AuthenticationStateEnum.Authenticated,
          userName: user.userName,
          email: user.email,
          userType: user.userType,
          accessToken: this.registerUser.accessToken,
          idToken: this.registerUser.idToken,
          userId: user.userId
        });
      });
    }
  } catch (e) {
    this.spinner.hide();
    console.log(e);
  }
}
```

5.2 Deployed Angular Client to AWS S3 and CloudFront using route 53

The Angular client was deployed to <https://www.verimarker.com>

Here are few important notes when deployed the Angular client to AWS S3 / CloudFront:

1. Create S3 bucket www.verimarker.com
2. In the S3 bucket, ensure to include the following bucket policy:



The screenshot shows the AWS S3 console with the bucket 'verimarker.com' selected. The 'Permissions' tab is active, specifically the 'Bucket Policy' section. A JSON policy is displayed in the editor:

```
1 {
2     "Version": "2012-10-17",
3     "Id": "Policy1576914666338",
4     "Statement": [
5         {
6             "Sid": "Stmt1576914654634",
7             "Effect": "Allow",
8             "Principal": "*",
9             "Action": "s3:GetObject",
10            "Resource": "arn:aws:s3:::verimarker.com/*"
11        }
12    ]
13 }
```

Below the policy, there are buttons for 'Delete', 'Cancel', and 'Save'.

3. In the S3 bucket, ensure to remove all check marks under Block public access:

(continued to next page ...)

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access

Off

- Block public access to buckets and objects granted through **new** access control lists (ACLs)
Off
- Block public access to buckets and objects granted through **any** access control lists (ACLs)
Off
- Block public access to buckets and objects granted through **new** public bucket or access point policies
Off
- Block public and cross-account access to buckets and objects through **any** public bucket or access point policies
Off

Edit

4. Enable CORS configuration in S3:

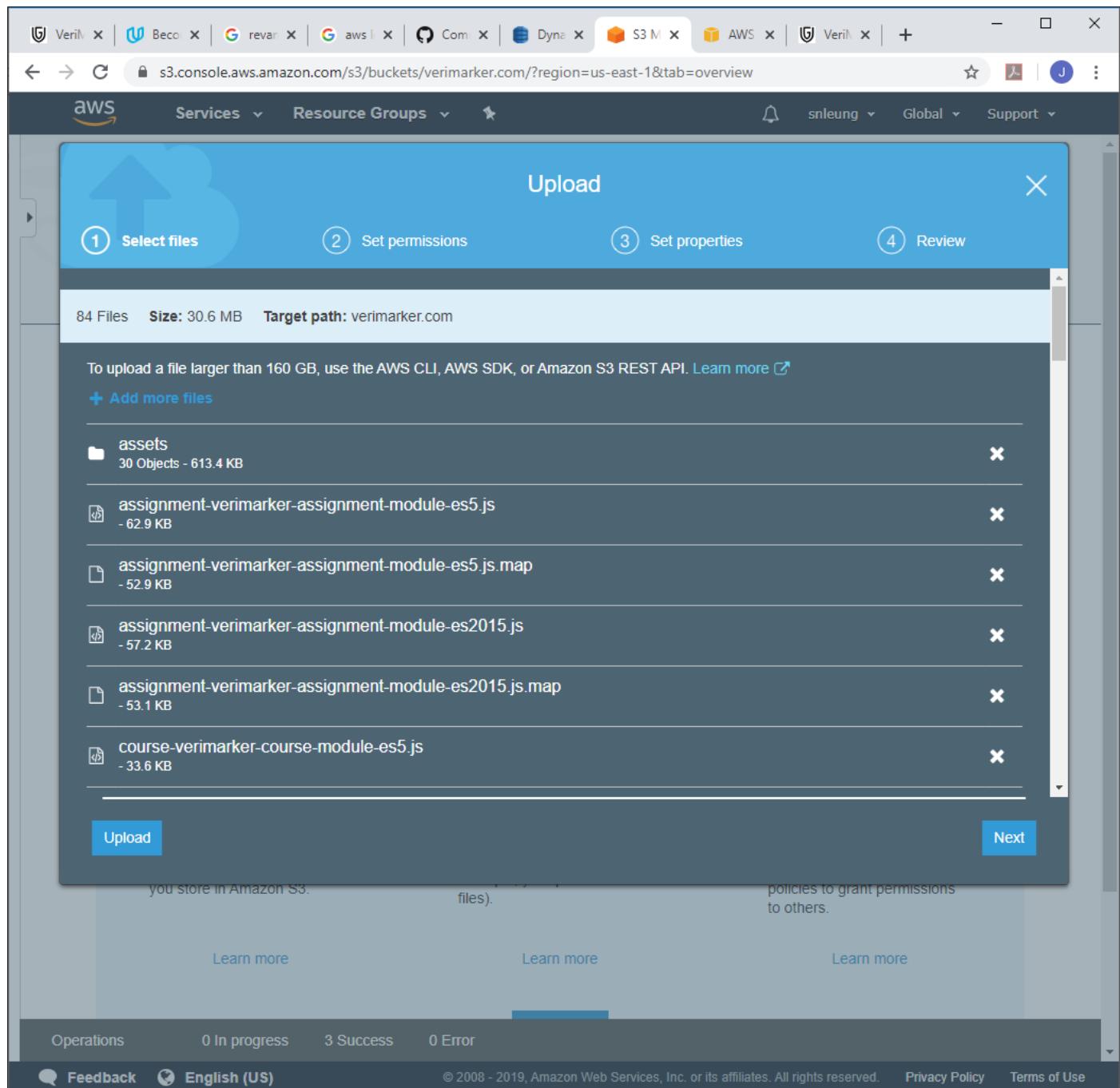
CORS configuration editor ARN: arn:aws:s3:::verimarker.com

Add a new cors configuration or edit an existing one in the text area below.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
3   <CORSRule>
4     <AllowedOrigin>*</AllowedOrigin>
5     <AllowedMethod>HEAD</AllowedMethod>
6     <AllowedMethod>GET</AllowedMethod>
7     <AllowedMethod>PUT</AllowedMethod>
8     <AllowedMethod>POST</AllowedMethod>
9     <AllowedMethod>DELETE</AllowedMethod>
10    <MaxAgeSeconds>3000</MaxAgeSeconds>
11    <ExposeHeader>ETag</ExposeHeader>
12    <AllowedHeader>*</AllowedHeader>
13  </CORSRule>
14 </CORSConfiguration>
```

Delete Cancel Save

5. Go to Visual Studio Code. In client folder, execute ng build.
6. This will build the Angular application located in client/dist/verimarker.
7. In S3, select the bucket www.verimarker.com. In Overview, click on Upload.
8. Drag the files inside the dist folder to the S3 Upload dialog:



9. Click Next.
(continued to next page ...)

10. Select “Grant public access to this object(s)” and Click Next to complete the upload.

The screenshot shows the AWS S3 console with the URL s3.console.aws.amazon.com/s3/buckets/verimarker.com/?region=us-east-1&tab=overview. The page is titled "Upload" and is at Step 2: Set permissions. It shows 84 files totaling 30.6 MB being uploaded to the target path "verimarker.com".

Manage users:

User ID	Objects	Object permissions
jsleung506(Owner)	<input checked="" type="checkbox"/> Read	<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write

Access for other AWS account: + Add account

Account	Objects	Object permissions

Manage public permissions:

Grant public read access to this object(s)

Warning: This object(s) has public read access.
Everyone in the world will have read access to this object(s).

Buttons: Upload, Previous, Next

Footer: Operations: 0 In progress, 3 Success, 0 Error. © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use. Feedback English (US)

(continued to next page ...)

11. Go to Properties. Select Static Web Site Hosting.
12. Select Use this bucket to host a website. Enter index.html for BOTH the Index document and the Error document:

The screenshot shows the 'Static website hosting' configuration dialog box on the AWS S3 console. The 'Index document' field contains 'index.html' and the 'Error document' field also contains 'index.html', both highlighted with red boxes. The 'Bucket hosting' checkbox is checked. To the right, there is a sidebar titled 'Object-level logging' with a note about CloudTrail data events and a 'Disabled' status indicator. The browser tab shows the URL s3.console.aws.amazon.com/s3/buckets/verimarker.com/?region=us-east-1&tab=properties.

13. Click Save. Go to Route 53. In registered domain, purchase a domain (verimarker.com).

(continued to next page ...)

14. In Certificate Manager, click on Create record in Route 53 for both verimarker.com and *.verimarker.com

The screenshot shows the AWS Certificate Manager interface. At the top, there are tabs for Capstone - Udacity, Udacity Reviews, Student Hub - Udaci, AWS Certificate Manager, DynamoDB - AWS Co., CloudWatch Manager, VeriMarker, New Tab, and Support. The AWS logo and Services menu are also visible.

The main page displays a table of certificates. One row is selected for "verimarker.com", which is listed under "Domain name". The "Status" column shows "Issued", "Type" is "Amazon Issued", "In use?" is "Yes", and "Renewal eligibility" is "Eligible".

Below the table, a "Status" section provides detailed information: "Status Issued" and "Detailed status The certificate was issued at 2019-10-11T14:33:24UTC".

A "Domain" table shows validation status for "verimarker.com" as "Success".

Instructions for DNS configuration are provided: "Add the following CNAME record to the DNS configuration for your domain. The procedure for adding CNAME records depends on your DNS service Provider." A table lists the record: Name _9d4988517e586f003cb1d35587dbc7d5.verimarker.com, Type CNAME, Value _83200160f46ca8c0c24b7b01fafb7ce0.lprtlswt.acm-validations.aws.

A note states: "Note: Changing the DNS configuration allows ACM to issue certificates for this domain name for as long as the DNS record exists. You can revoke permission at any time by removing the record." A link "Create record in Route 53" is highlighted with a red box.

15. Go to CloudFront, Create Distribution (for Web):

The screenshot shows the "Create Distribution" wizard in the CloudFront console. The "Origin Settings" section includes fields for "Origin Domain Name" (set to "verimarker.com.s3-website-us-east-1.am"), "Origin Path" (empty), "Origin ID" (empty), and "Origin Custom Headers" (empty). The "Default Cache Behavior Settings" section includes "Path Pattern Default (*)" (with a dropdown arrow), "Viewer Protocol Policy" (radio buttons for "HTTP and HTTPS" (disabled), "Redirect HTTP to HTTPS" (selected), and "HTTPS Only" (disabled)), "Allowed HTTP Methods" (radio buttons for "GET, HEAD" (disabled), "GET, HEAD, OPTIONS" (disabled), and "GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE" (selected)), "Field-level Encryption Config" (dropdown menu), "Cached HTTP Methods" (checkboxes for "GET, HEAD (Cached by default)" and "OPTIONS" (disabled)), "Cache Based on Selected Request Headers" (dropdown menu set to "None (Improves Caching)"), and "Object Caching" (radio buttons for "Use Origin Cache Headers" (selected) and "Customize"). Finally, "Minimum TTL" is set to 0 and "Maximum TTL" is set to 31536000.

16. For Origin Domain Name, MANUALLY typed in verimarker.com.s3-website-us-east-1.amazonaws.com .

17. Select Redirect HTTP to HTTPS.

18. Allowed HTTP Methods: GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE.

The screenshot shows the AWS CloudFront distribution configuration interface. Key settings highlighted with red boxes include:

- Alternate Domain Names (CNAMEs):** verimarker.com, www.verimarker.com
- SSL Certificate:** Custom SSL Certificate (example.com) selected. A dropdown shows "verimarker.com (bb55655a-264f-486b-ac)" and a button to "Request or Import a Certificate with ACM".
- Custom SSL Client Support:** Clients that Support Server Name Indication (SNI) - (Recommended) selected. A note states: "CloudFront serves your content over HTTPS to clients that support SNI. SNI is supported by browsers and clients released after 2010. There is no additional charge for this option." with a "Learn More" link.
- Security Policy:** TLSv1.1_2016 (recommended) selected. A note states: "See the list of protocols and ciphers that CloudFront uses for each security policy."
- Supported HTTP Versions:** HTTP/2, HTTP/1.1, HTTP/1.0 selected.
- Default Root Object:** index.html
- Logging:** On

19. For Alternate Domain Names, enter verimarker.com and www.verimarker.com

20. SSL Certificate: select Custom SSL, and select verimarker.com in the dropdown.

21. Default Root Object: index.html.

22. Remove checkbox for Enable IPv6 (at the bottom of the page).

23. Click Create Distribution.

(continued to next page ...)

24. Go to Route 53 (The following already show the A and CNAME records for CloudFront distribution were already created):

The screenshot shows the AWS Route 53 console with the URL `console.aws.amazon.com/route53/home?region=us-east-1#resource-record-sets:Z1NUHN6Q5EJOW8`. The interface displays a list of record sets for the domain `verimarker.com.`. The table has columns for Name, Type, and Value. The record sets listed are:

Name	Type	Value
<code>verimarker.com.</code>	A	ALIAS <code>d1hxrejopy4ta.cloudfront.net. (z2fdtnadataqyw</code>
<code>verimarker.com.</code>	NS	<code>ns-1494.awsdns-58.org.</code> <code>ns-203.awsdns-25.com.</code> <code>ns-1844.awsdns-38.co.uk.</code> <code>ns-700.awsdns-23.net.</code>
<code>verimarker.com.</code>	SOA	<code>ns-1494.awsdns-58.org. awsdns-hostmaster.amazon</code>
<code>_9d4988517e586f003cb1d35587dbc7d5.verimarker.com.</code>	CNAME	<code>_83200160f46ca8c0c24b7b01fafb7ce7.olprtlswtu.ac</code>
<code>www.verimarker.com.</code>	CNAME	<code>d1hxrejopy4ta.cloudfront.net</code>

25. Create 'A' Record Set: Select Alias = 'Yes', Alias Target: `d1hxrejopy4ta.cloudfront.net`. (the CloudFront distribution of VeriMarker):

The screenshot shows the AWS Route 53 console with the URL `console.aws.amazon.com/route53/home?region=us-east-1#resource-record-sets:Z1NUHN6Q5EJOW8`. A new record set is being created for the domain `verimarker.com.`. The 'Edit Record Set' dialog is open, showing the following configuration:

- Name:** `verimarker.com.`
- Type:** `A – IPv4 address`
- Alias:** `Yes` (radio button selected)
- Alias Target:** `d1hxrejopy4ta.cloudfront.net.`
- Alias Hosted Zone ID:** `Z2FDTNADATAQYW2`
- Routing Policy:** `Simple`
- Evaluate Target Health:** `No`

26. Create CNAME record set. Enter value: d1hxrejopy4ta.cloudfront.net.

The screenshot shows the AWS Route 53 console with the URL `console.aws.amazon.com/route53/home?region=us-east-1#resource-record-sets:Z1NUHN6Q5EJOW8`. The 'Create Record Set' button is highlighted. On the right, the 'Edit Record Set' panel is open for a record set named 'www.verimarker.com.'. The 'Type' dropdown is set to 'CNAME – Canonical name'. The 'Value' field contains the value 'd1hxrejopy4ta.cloudfront.net', which is highlighted with a red box. The 'TTL (Seconds)' dropdown shows '300'. The 'Routing Policy' is set to 'Simple'. At the bottom right of the panel is a 'Save Record Set' button.