




School of Computing Technologies
COSC2626/2640 Cloud Computing
Assessment 1

	<p>Assessment Type: Group project (maximum of four students)</p> <p>Submit your solutions online via Canvas>Assignments>Assessment 1.</p> <p>Clarifications/updates may be made via announcements/relevant discussion forums.</p>
	<p>Due date: 11:59PM on Friday, 11th April 2025.</p> <p>A university standard late penalty of 10% (= 4 marks) per day applies for the days that you are late to submit your work for, unless a special consideration or an extension has been granted.</p>
	<p>Weighting: 40 marks</p>

Overview

Develop an AWS EC2, S3, API Gateway, Lambda, and DynamoDB based web application.

Assessment Criteria

This assessment will assess your ability to develop cloud-based applications using AWS EC2, S3, API Gateway, Lambda, and DynamoDB services.

Learning Outcomes

This assessment is relevant to the following Learning Outcomes:

- Develop and deploy cloud applications using popular cloud platforms.
- Design and develop highly scalable cloud-based applications by creating and configuring virtual machines on the cloud and building private cloud.

Group Work

Group work should be completed in teams of no more than four students. You may form groups with any student in the course. Each group must designate a group leader.

You have the option to complete this assignment on your own.

Each student is required to actively participate and demonstrate collaboration on every task. By default, all group members will receive the same grade. To ensure effective management of your group work and to demonstrate ongoing contributions to your project, **the group leader will be responsible for maintaining detailed contribution logs that document each member's involvement in all tasks.** These logs must document the tasks completed, hours spent, and any other relevant details of everyone's contributions. They may be reviewed to resolve any disputes over contributions or to adjust grades if necessary.

Should the group disband, each member must be capable of completing the project on their own.

If there are any issues that affect your work, such as being sick, you must keep your group informed in a timely manner. Your final grade will be based on the work you and your partner complete throughout the duration of the assignment. We will treat everybody fairly, and account for times when issues arise when marking your group work and contributions. However, you must be upfront and communicate with us. If you fail to inform us of issues in a timely fashion and we deem that your actions significantly harm your group's performance, we may award you a reduced grade. It is academic misconduct if you are deliberately dishonest, lie to, or mislead your group or teaching staff in a way that harms your group.

Notify your tutor immediately if you encounter any issues working with your group at any time. We will do our best to help manage group issues, so that everybody receives a fair grade for their contributions.

Assessment Details

Create a simple online music subscription application that leverages AWS S3 and DynamoDB. Deploy your web application on a free-tier EC2 instance running Ubuntu Server 20.04/18.04 24.04 LTS AMI).

Your assessment will be marked during your demo time (further details about the demo will be provided in Week 5). Examiners will ask questions to assess how well students understand the AWS services implemented. Marks will be assigned based on two key criteria: (1) the functional correctness of your application, and (2) your demonstrated understanding of the services employed in the project.

Important Note:

1. The application should be **FULLY hosted** in a web server environment – **Apache2 (or a suitable alternative)** and be able to be rendered in a web browser by directly entering the root Public IPv4 DNS (in either https or http) of the EC2 instance hosting the application.

Please make sure that markers can access your web application via the standard HTTP(S) port of your EC2 public DNS, for example:

<http://ec2-54-85-208-90.compute-1.amazonaws.com> OR

<https://ec2-54-85-208-90.compute-1.amazonaws.com>

2. Using Elastic Beanstalk is **NOT allowed** as a valid means of deploying your application.
3. Using non-standard HTTP ports (i.e. ones other than 80/443) are also **NOT allowed**.
4. AWS supports multiple programming languages, allowing developers to choose the language that best suits their needs and preferences.

Please refer to <https://aws.amazon.com/developer/>

You can use any language(s) that you are familiar with. Your application still needs to fulfil the same requirements (. e.g., running on an AWS Ubuntu EC2 instance, connecting to a DynamoDB, S3 actions, ...etc.).

5. You can use any framework providing you are writing the code for the core functions.
6. AWS Academy account does not allow to create any IAM roles. However, a role named **LabRole** has been pre-created for you. Use **LabRole** instead.
7. Students would also need to securely access the images stored in S3. This is considered security best practice and should always be done.
8. Students must meticulously design the DynamoDB key schema and secondary index to maximize query performance. Both the **Query** and **Scan** operations should be implemented appropriately for data retrieval from DynamoDB tables.
9. Please ensure that your implementation correctly handles various HTTP request methods — such as GET and POST — when using the API Gateway REST API.

Task Description

Task 1: AWS DynamoDB (2 + 2 + 2 = 6 marks)

- 1.1** Create a **login** table in DynamoDB containing 10 entities with the following attributes and values. **Note: This task can be done via code or console.**

email (Type: String)	user_name (Type: String)	password (Type: String)
s3#####0@student.rmit.edu.au i.e., your RMIT student id+'0@student.rmit.edu.au'	FirstnameLastname0 i.e., your name+'0'	012345
s3#####1@student.rmit.edu.au i.e., your RMIT student id+'1@student.rmit.edu.au'	FirstnameLastname1 i.e., your name+'1'	123456
...
s3#####9@student.rmit.edu.au i.e., your RMIT student id+'9@student.rmit.edu.au'	FirstnameLastname9 i.e., your name+'9'	901234

- 1.2** Write a program to create a table titled **music** in DynamoDB with the following attributes: **title**, **artist**, **year**, **album**, and **image_url**.
- 1.3** Write a program to load the data from **2025a1.json** to your **music** table.

A single song can have multiple versions (for example, deluxe editions, live recordings, and remix versions), performed by either the same or different artist. Different versions of the same song are released on different albums.

Note: It is acceptable that the program creates the table with only the key attribute(s) in Task 1.2. The non-key attributes can be created when the program loads the data from the JSON file **2025a1.json** in this task.

Task 2: AWS S3 (2 marks)

Write a program that automatically downloads all artist images based on the **image_url** values found in **2025a1.json** and then uploads these images to an S3 bucket.

Task 3: Login Page (2 + 2 = 4 marks)

The **login** page contains an “**Email**” text field, a “**Password**” field, and a “**Login**” button as well as a **register** link.

3.1 When user clicks the **Login** button, it will validate if the user-entered credentials match with the information stored in the **login table**.

3.2 If the user credential is invalid, the login page will display “**email or password is invalid**”; it will be redirected to the **main** page otherwise.

Task 4: Register Page (1 + 2 + 2 = 5 marks)

The **register** page contains an “**Email**” text field, a “**Username**” field, a “**Password**” field, and a “**Register**” button. When a user clicks the “**Register**” button, it will validate if the user-entered email matches with the email stored in the **login** table created in task 1.1.

Note: **Each registration must have a unique email address.** The username is not intended to serve as a unique credential for authentication, therefore, it is possible for two (or more) users to have accounts with the same username.

4.1 If the entered email matches with the email stored in the login table, the register page will show “**The email already exists**”.

4.2 If the entered email is unique, the new user information will be stored in the **login** table, and the user will be redirected to the login page, where user can login with the new email and password.

Task 5: Main Page (15 marks)

The **main** page contains **three areas** (a **user area**, a **subscription area**, and a **query area**) and a “**Logout**” link.

Note: For a newly registered user who hasn't subscribed to any songs yet, the subscription area should be empty. Displaying all songs stored in the database is not advisable, since in a real-world scenario it could contain millions of songs.

5.1 (1 mark) After a user log in, the **user area** will show the corresponding **user_name**.

5.2 (1 + 2 + 3 = 6 marks) The **subscription area**

5.2.1 The subscription area will show all the user subscribed music information (title, artist, year, and album) stored in DynamoDB.

5.2.2 Each music information is followed by the corresponding artist image retrieved from S3 and a **“Remove”** button.

5.2.3 If the user clicks the **“Remove”** button, the corresponding subscribed music information and artist information will be removed from the subscription area and the corresponding table in DynamoDB.

5.3 (1 + 2 + 2 + 2 = 7 marks) The **query area** should contain four text areas, **“Title”**, **“Year”**, **“Artist”**, **“Album”**, and a **“Query”** button. The user may fill in one or more of these text areas and then click the **Query** button, but at least one field must be completed.

5.3.1 If the queried information is **not contained** in the entities' corresponding attribute value(s) in the **music** table, it will show **“No result is retrieved. Please query again”**.

5.3.2 (2 + 2 + 2 = 6 marks) If the queried information is **contained** in (one or more) entities' corresponding attribute value(s) in the **music** table, the area will show

- All the retrieved music information (title, artist, album, and year).

Note: If a user enters more than one query conditions, the query conditions are connected by **“AND”** operator(s) by default.

- Each music information is followed by the corresponding artist images retrieved from S3 and a **“Subscribe”** button.
- If the user clicks the **“Subscribe”** button, the subscribed music information and the corresponding artist image will be added into the subscription area and the subscribed music information will be stored in DynamoDB.

Note: During the demonstration, markers will provide specific criteria for search songs, for example **“Please find all songs of Jimmy Buffett in 1974”**, **“Please find all versions of the song Rivers of Babylon”**, **“Please find all songs by Taylor Swift in the album Fearless”**, **“List all tracks from the album White Blood Cells”**, etc. You will need to formulate a query based on that information, and your web application should return the corresponding results.

5.4 (1 mark) If the user clicks the **“Logout”** link, the user will be redirected to the **login** page.

Task 6 (Advanced Task): The use of API Gateway and Lambda functions (8 marks)

The application needs to read and update the records (user, music, and subscription) in DynamoDB through API Gateway and Lambda function. For example, when

1. users register on the website, and
2. users subscribe to new songs, and

3. users remove subscribed song(s),

the requests will pass through the API Gateway **REST API**.

The Lambda function gets the requests (read or write) and communicates with DynamoDB.

Warning: If you have too many Lambda functions running concurrently (**up to 10**), AWS will deactivate your account. This issue occurs not because of the total number of Lambda functions you have, but rather because of the number that are running simultaneously. For instance, making too many consecutive Lambda calls in a short period of time can trigger this deactivation since only running functions consume resources. Please avoid making consecutive Lambda calls within a short time span.

Important Notes:

1. The whole application must be **COMPLETELY** programmed by yourself.

2. Writing code is similar to academic writing in that **when you use or adapt code developed by someone else as part of your project, you must cite your source**. However, instead of quoting or paraphrasing a source, you include an inline comment in the code. For example:



Code adapted from MSDN example:

[http://msdn.microsoft.com/en-us/library/ms680578\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms680578(VS.85).aspx)



3. The whole application must be **FULLY** deployed in the designated EC2 instance, otherwise **NO MARK** will be given for Task 3, 4 and 5. **To be more specific**, you will use EC2 to host the application, S3 to store the images, and DynamoDB to store the records.

4. Please read the spec carefully to avoid any unnecessary mark deduction.

5. Please read the online rubric carefully before your submission.

Submission Format

Create a .zip file. This .zip file will contain the source code you developed – individual programs (Task 1 and 2), the code running on the Apache, and your Lambda functions (pasted into a file).

You do not need to submit the API gateway configuration.

Each group must submit ONE consolidated submission, which includes all required components, including the working logs. If the group leader's student ID is S1234567, please name your zip file as S1234567_A1.zip.

Submit the .zip file on Canvas before the deadline. Your submission will be marked during the demonstration time. This submission is only for keeping the records. **Your assessment will NOT be allowed to be demonstrated until you submit in Canvas.**

Demonstration

You will need to demonstrate your project **online via Teams** by scheduling an appointment with a tutor in Week 5. The demo booking will be made available to students in early Week 5.

All demonstrations must be completed during Week 7 and mid-break. There will be penalties if you fail to complete and demonstrate your work by 28/04 unless you have a valid extension or special consideration granted. The demo will take around 15 minutes for each group.

Further details on the demonstrations and their structure will be announced in Week 5.

Note: Each group is allowed only one demonstration session for this assessment. Overbooking additional slots will result in a 10% penalty. The group should collaboratively decide which member(s) will lead the demonstration and field questions during the demo. The demonstrator(s) should have a thorough understanding of every aspect of the project.

Academic Integrity and Plagiarism (standard warning)

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge, and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarized, paraphrased, discussed, or mentioned in your assessment through the appropriate referencing methods,
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviors, including:

- Failure to properly document a source.
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to the [University website](#).

Assessment Declaration

When you submit work electronically, you agree to the assessment declaration.