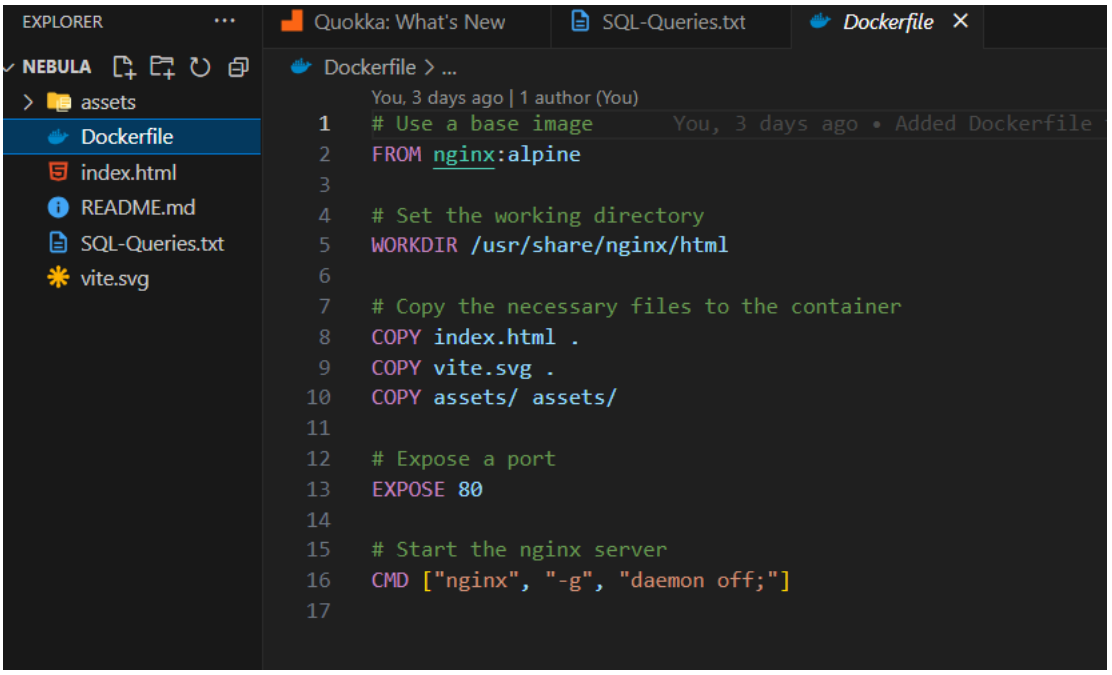
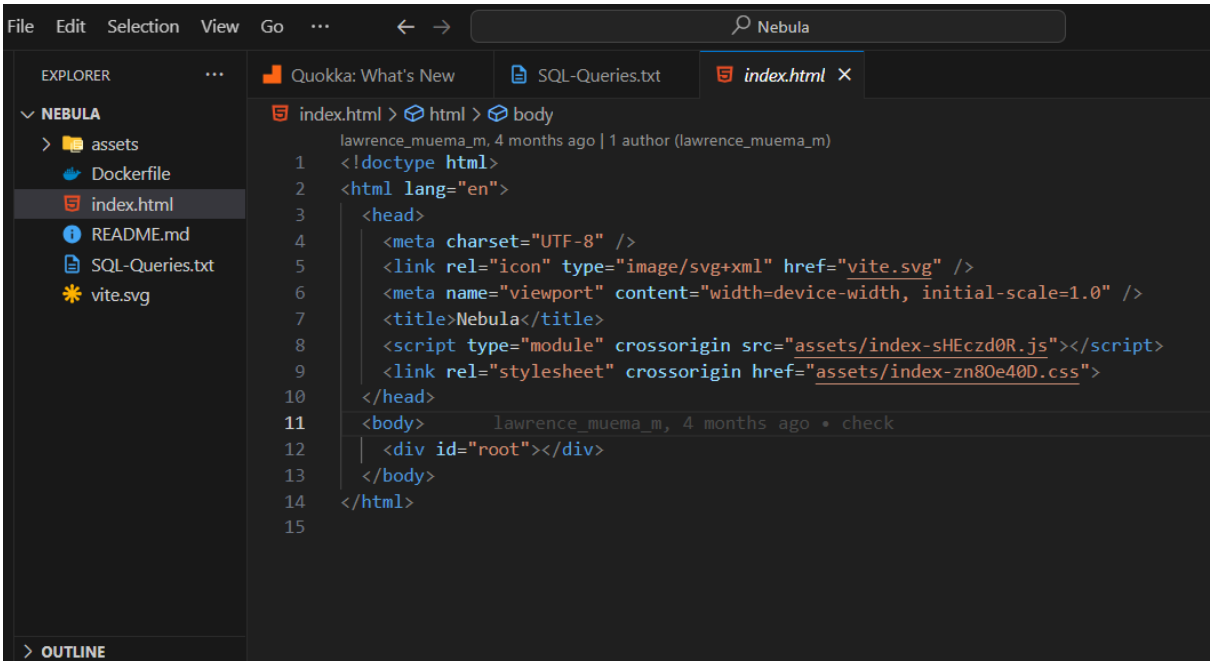
Azubi Project 3: Nebula FrontEnd+Backend Project

Author	Olatunji Olayinka Oluwaseun
Contributors	Brian Mathenge, Adade Sedom Percy, and Pauline Andege Omondi
GitHub Repository	https://github.com/olatunji-weber/dockerproj

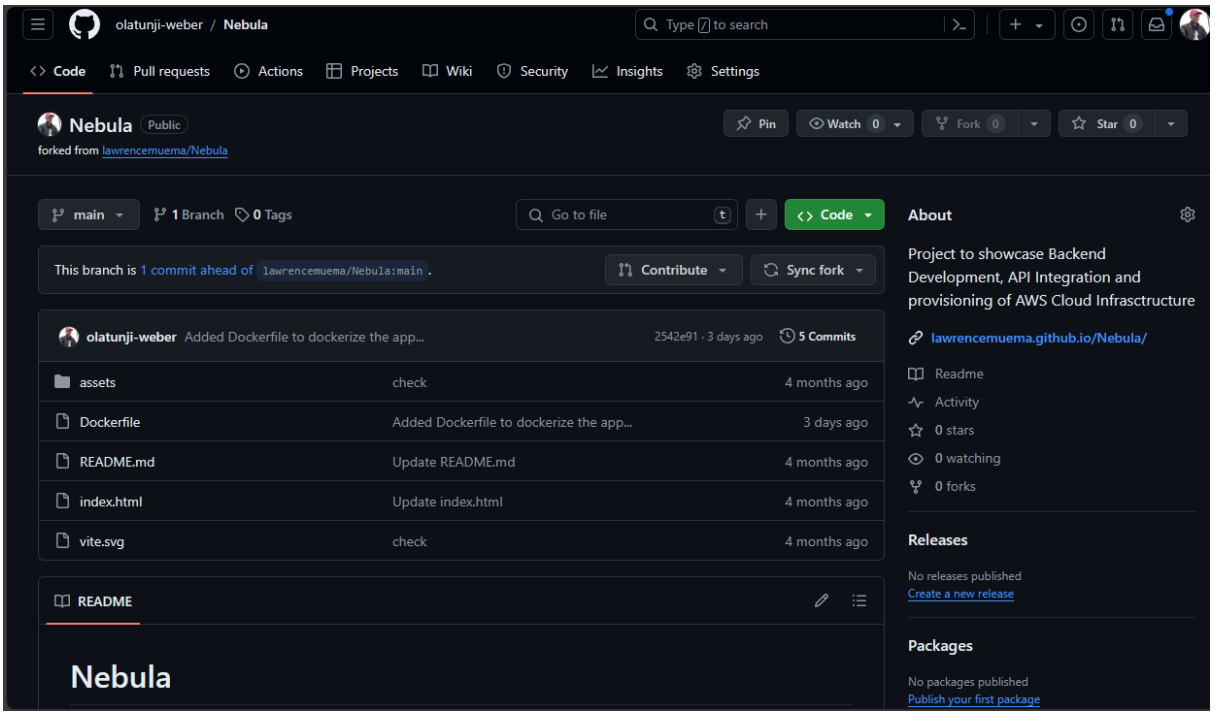
We cloned the given repository and using the Visual Studio code editor, we then dockerized the application using the “nginx” image (alpine light weight version).



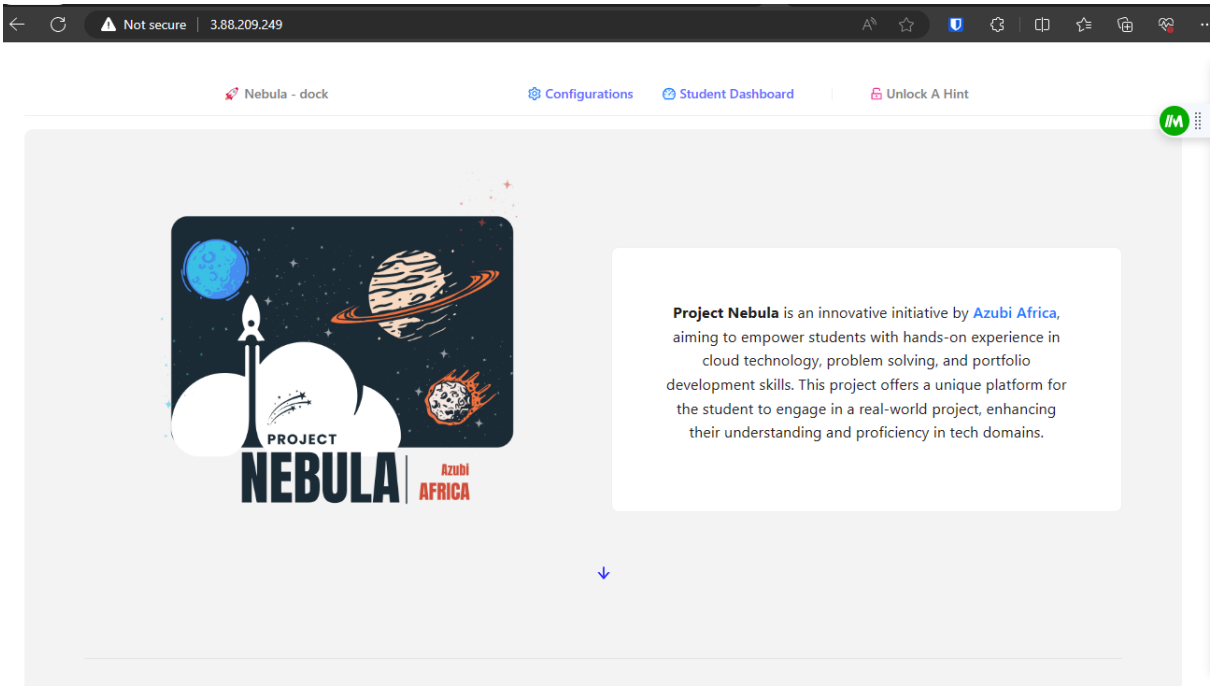
And this is the “index.html” file to be served when the application loads.



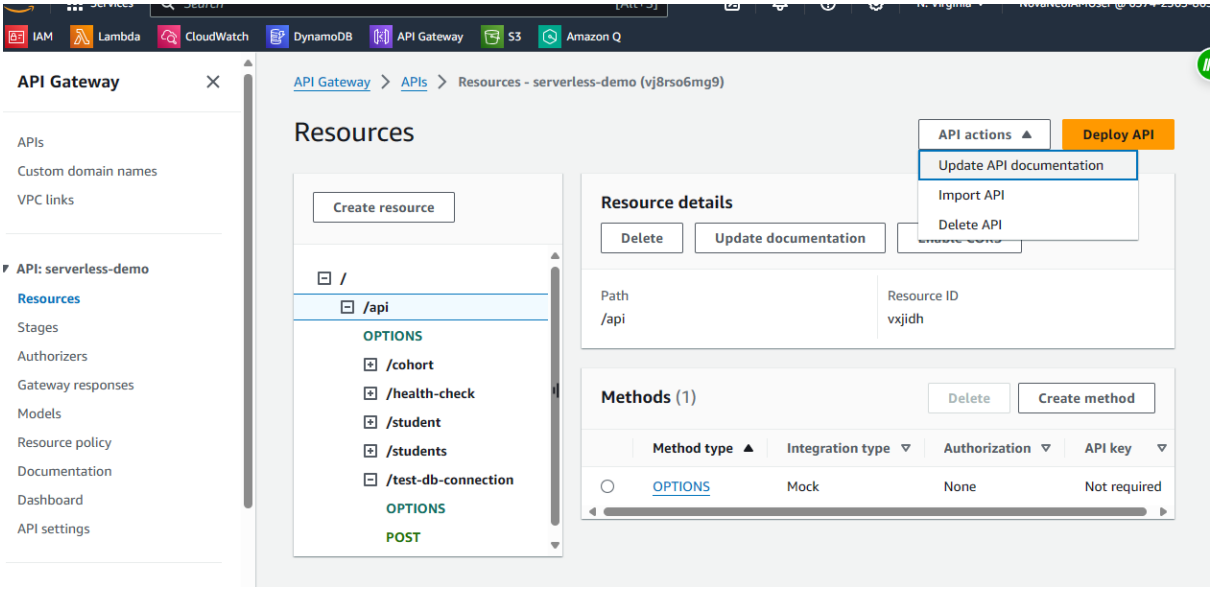
Our GitHub Repository:

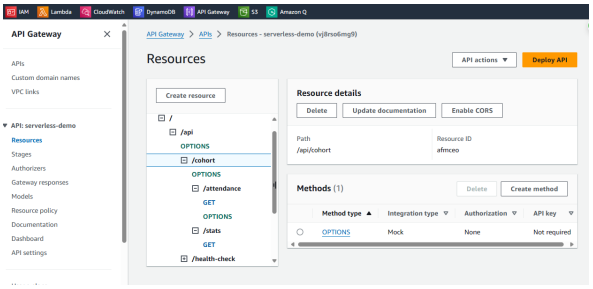


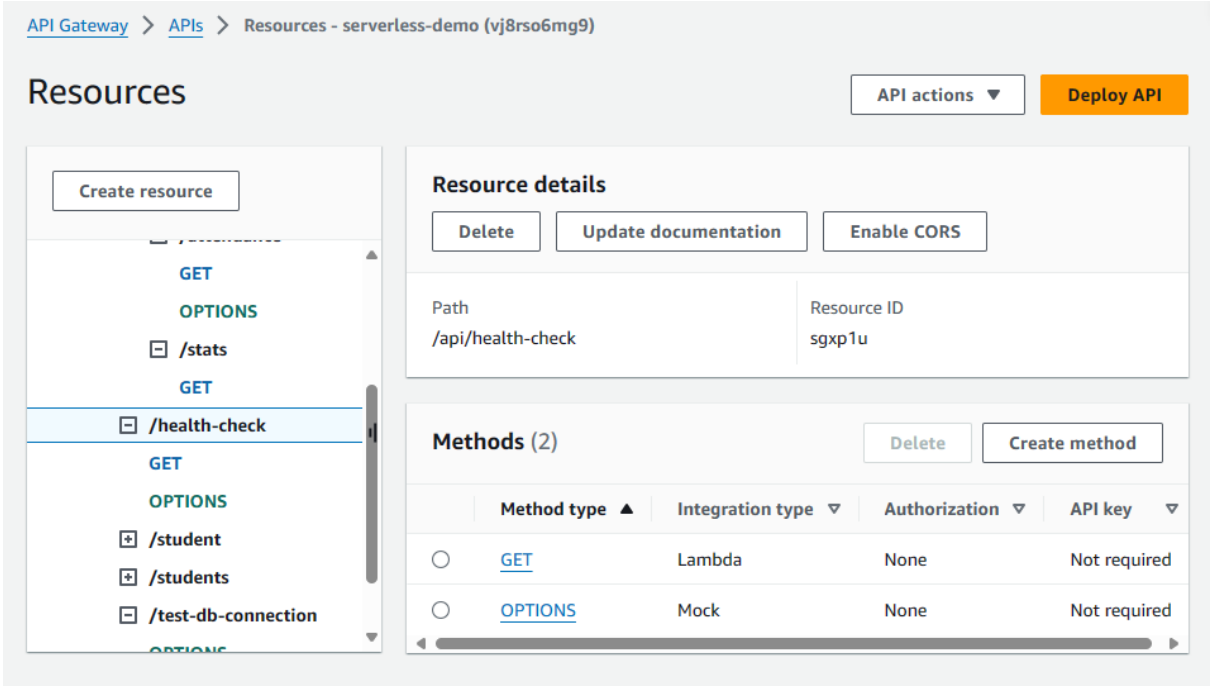
We loaded the docker image to ECR and here is the index.html being serves from a live container on AWS ECS.

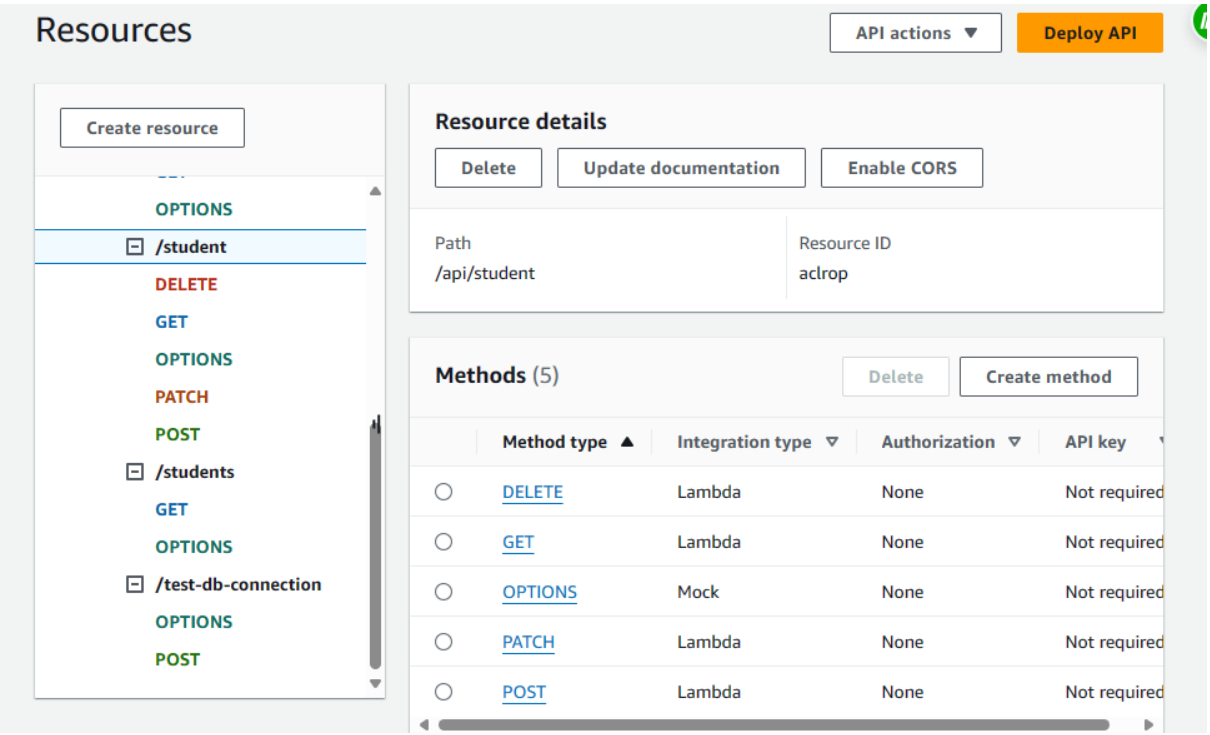


We then provisioned API endpoints in AWS API Gateway which will be used to query the MySQL backend database which we are creating on AWS RDS.


IAM
Lambda
CloudWatch
DynamoDB
API Gateway
S3
Amazon Q
API Gateway
APIs
Custom domain names
VPC links
API: serverless-demo
Resources
Stages
Authorizers
Gateway responses
Models
Resource policy
Documentation
Dashboard
API settings
API Gateway > APIs > Resources - serverless-demo (vj8rso6mg9)
Resources
API actions
Deploy API
Update API documentation
Import API
Delete API
Create resource
/
/api
OPTIONS
/cohort
/health-check
/student
/students
/test-db-connection
OPTIONS
POST
Resource details
Delete
Update documentation
Path
/api
Resource ID
vxjidh
Methods (1)
Delete
Create method
Method type
Integration type
Authorization
API key
OPTIONS
Mock
None
Not required




API Gateway > APIs > Resources - serverless-demo (vj8rso6mg9)
Resources
API actions
Deploy API
Create resource
GET
OPTIONS
/stats
GET
/health-check
GET
OPTIONS
/student
/students
/test-db-connection
Resource details
Delete
Update documentation
Enable CORS
Path
/api/health-check
Resource ID
sgxp1u
Methods (2)
Delete
Create method
Method type
Integration type
Authorization
API key
GET
Lambda
None
Not required
OPTIONS
Mock
None
Not required



This part of the project is meant to create a working database and host a backend server (locally or online). The front-end files will be found on GitHub as well as some data attached in the project doc below.

1. GitHub app to clone:

https://github.com/lawrencemuema/Nebula.git

- Api and technical stuff will be found on the **configuration page**

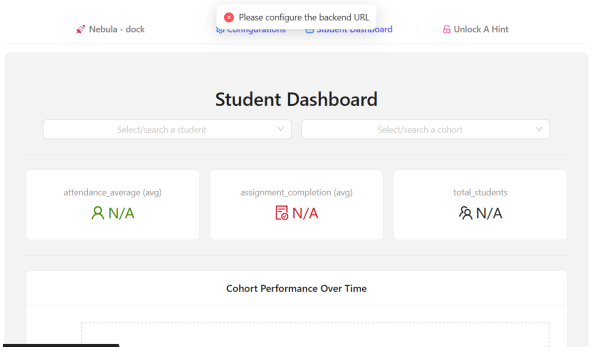
2. Sample front-end:

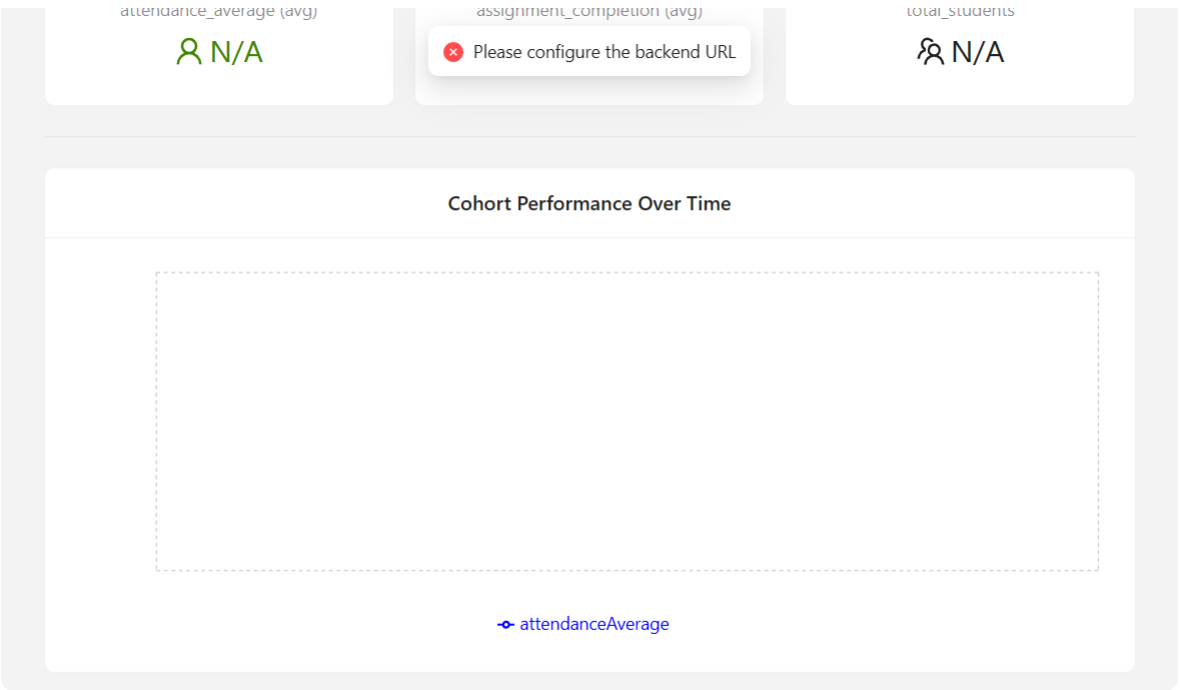
https://nebula1.labmero.com

3. Sample back-end:

https://labmero.com/nebula_server

4. Project doc and sample data is attached





Data Structure Explanation

The data represents information about a student in a dashboard system. Each student record includes:

- **Basic Details:**

  Includes the `name`, `email`, `cohort` (the group or class the student belongs to), and `ranking` (a numerical ranking of the student).

- **Academic Performance Metrics:**

  Consists of `assignment_completion` (the number of assignments completed) and `attendance_average` (the average attendance percentage).

- **Weekly Attendance List:**

  A detailed breakdown of weekly attendance. Each entry includes the `week` identifier, the number of days the student was `present` and `absent`.

Data Modeling for the Nebula Student Information Backend Database

To design a data model for a student information database based on the given specifications, we need to consider the different components and their relationships. Here's a approach we adopted:

1. Identify Entities:

The entities in this case are the student and the weekly attendance. Each student record will contain basic details and academic performance metrics, while the weekly attendance list will provide a detailed breakdown of the student's attendance.

2. Define Attributes:

For the student entity, the attributes include:

- Name
- Email
- Cohort
- Ranking

- Assignment completion
- Attendance average

For the weekly attendance entity, the attributes include:

- Week identifier
- Number of days present
- Number of days absent

3. Establish Relationships:

The relationship between the student entity and the weekly attendance entity is one-to-many, as each student can have multiple weekly attendance records.

4. Design the Data Structure:

Based on the given information, a relational database model would be suitable for this scenario. In a relational database, the data is organized into tables, with each table representing an entity. The relationships between entities are established through common fields.
Here's an example of how the data structure could be represented:

Table: students

ranking	name	email	cohort	Assignment Completion	Attendance Average
1	John Travolta	john@example.com	Nebula	10	90
2	Jane Heir	jane@example.com	Nebula	8	95

Table: weekly_attendance

attendance_id	ranking	week_id	days_present	days_absent
1	1	Week 1	5	2
2	1	Week 2	6	1
3	2	Week 1	7	0

Table: assignments

attendance_id	ranking	week_id	assignmer
1	1	Week 1	5
2	1	Week 2	6
3	2	Week 1	7

In this structure, the "ranking" field in the "weekly_attendance" table serves as a foreign key (originally in the “students” table), linking each attendance record and assignment tables to the corresponding student.

We downloaded the practice data “nebula_cohort_practice_data” and created tables that will be utilized in our data modeling.

Then, we modeled the data using Power Pivot in MS Excel and exported the data in “Open Document Spreadsheet” format in order to be able to import it into the AWS RDS MySQL database that we have created.

Power Pivot for Excel - nebula_cohort_practice_data.xlsx

File | Home | Design | Advanced

Paste | Paste Append | Paste Replace | Copy — Clipboard

From Database | From Data Service | From Other Sources | Existing Connections — Get External Data

Refresh | PivotTable

Data Type : | Format : | $ % , — Formatting

Sort A to Z | Sort Z to A | Clear Sort | Clear All Filters | Sort by Column — Sort and Filter

Find — Find

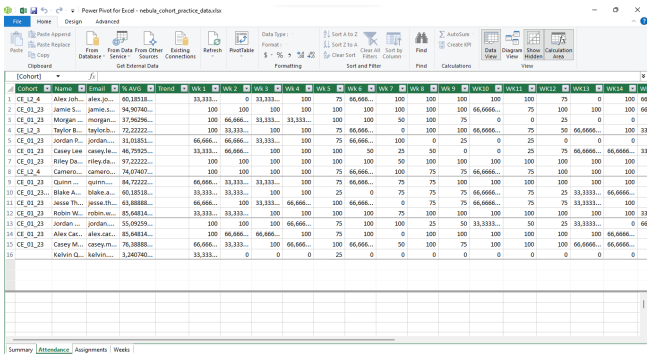
AutoSum | Create KPI — Calculations

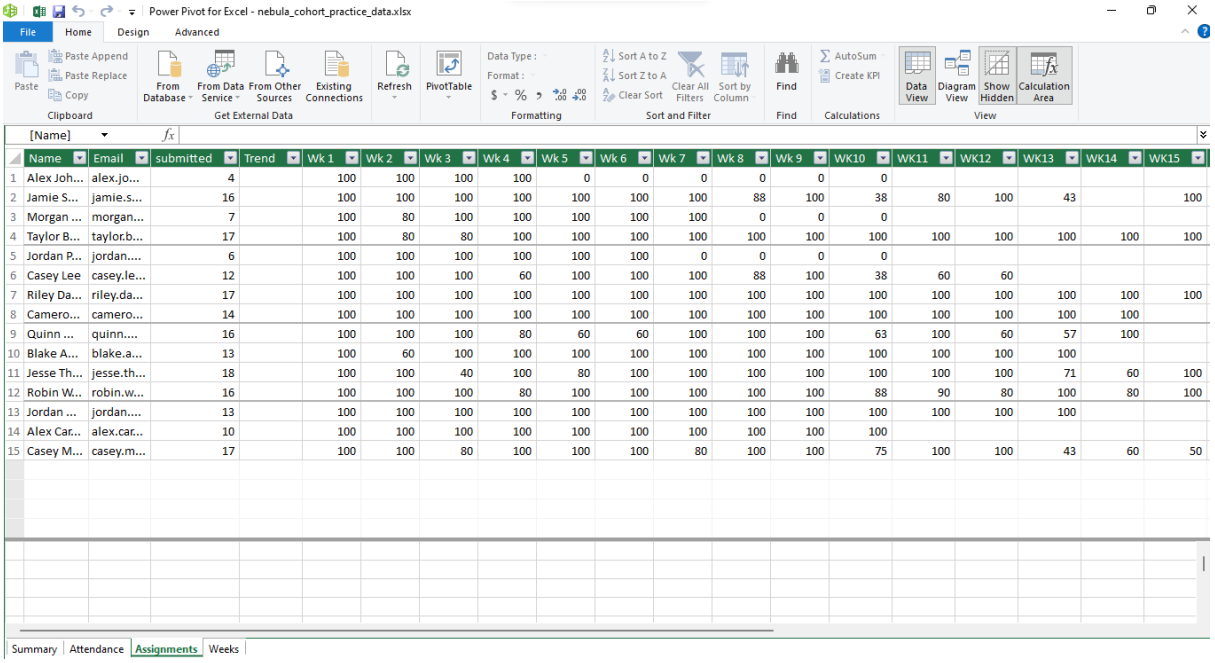
Data View | Diagram View | Show Hidden | Calculation Area — View

[Cohort]

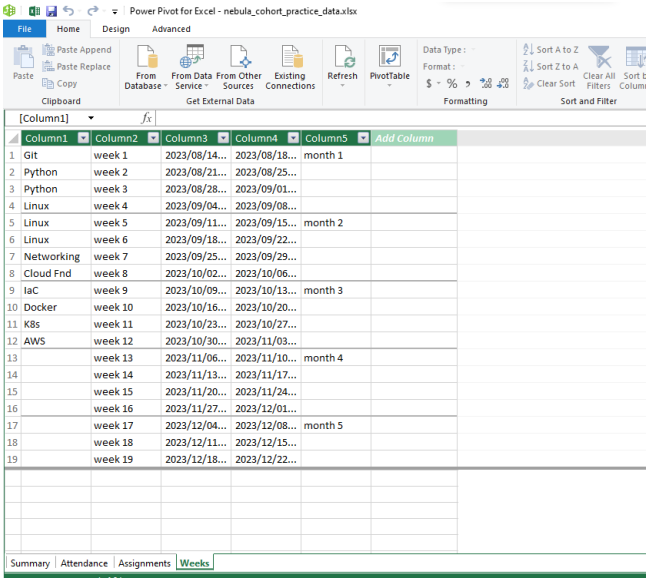
	Cohort	RANK	Student	Email	personal	Attendance	A-Comp.	Quiz Submitted	Q-Comp.	Score	Compliance	Watcher	Add Column
1	Nebula	1	Riley Davis	riley.da...	wamwangi...	97,222222222...	High	17	High	99,9305...	High Complia...		
2	Nebula	2	Jamie Smith	jamie.s...	adadesedo...	94,907407407...	High	16	High	96,4351...	High Complia...		
3	Nebula	3	Robin Whi...	robin.w...	fkayitey@s...	85,648148147...	High	16	High	89,9537...	High Complia...		
4	Nebula	4	Quinn Wil...	quinn....	catherinek...	84,722222222...	High	16	High	89,3055...	High Complia...		
5	Nebula	5	Casey Mar...	casey.m...	munguiisa...	76,388888889...	Good	17	High	85,3472...	High Complia...		
6	Nebula	6	Taylor Bro...	taylor.b...	audreymur...	72,222222222...	Good	17	High	82,4305...	High Complia...		
7	Nebula	7	Alex Carter	alex.car...	micahcons...	85,648148148...	High	10	Good	78,7037...	Good Complia...		
8	Nebula	8	Jesse Tho...	jesse.th...	faithlihabi...	63,888888889...	Good	18	High	78,4722...	Good Complia...		
9	Nebula	9	Cameron ...	camero...	calebyebo...	74,074074074...	Good	14	High	78,1018...	Good Complia...		
10	Nebula	10	Blake And...	blake.a...	chijiokedg...	60,185185185	Good	13	High	66,5046...	Good Complia...		
11	Nebula	11	Jordan Gr...	jordan....	higustave...	55,092592592...	Medium	13	High	62,9398...	Good Complia...		
12	Nebula	12	Casey Lee	casey.le...	daltonbigir...	46,759259259...	Medium	12	Good	55,2314...	Medium Com...		
13	Nebula	13	Alex John...	alex.jo...	adamsyezi...	60,185185185	Good	4	Low	49,6296...	Medium Com...		
14	Nebula	14	Morgan B...	morgan...	gillesadjrol...	37,962962962...	Low	7	Medium	39,6990...	Medium Com...		
15	Nebula	15	Jordan Par...	jordan....	ayubmaina...	31,018518518...	Low	6	Low	32,9629...	Medium Com...		

Summary | Attendance | Assignments | Weeks



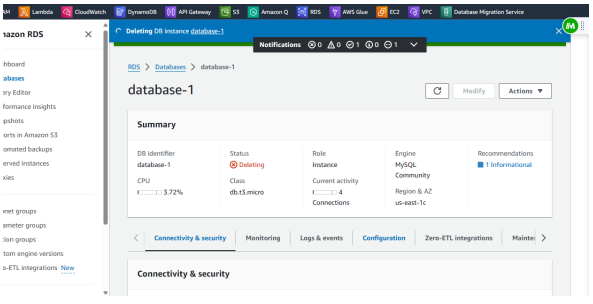


	Name	Email	submitted	Trend	Wk 1	Wk 2	Wk 3	Wk 4	Wk 5	Wk 6	Wk 7	Wk 8	Wk 9	WK10	WK11	WK12	WK13	WK14	WK15
1	Alex Joh...	alex.jo...	4		100	100	100	100	0	0	0	0	0	0					
2	Jamie S...	jamie.s...	16		100	100	100	100	100	100	100	88	100	38	80	100	43		100
3	Morgan ...	morgan...	7		100	80	100	100	100	100	100	0	0	0					
4	Taylor B...	taylor.b...	17		100	80	80	100	100	100	100	100	100	100	100	100	100	100	100
5	Jordan P...	jordan....	6		100	100	100	100	100	100	0	0	0	0					
6	Casey Lee	casey.le...	12		100	100	100	60	100	100	100	88	100	38	60	60			
7	Riley Da...	riley.da...	17		100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
8	Camero...	camero...	14		100	100	100	100	100	100	100	100	100	100	100	100	100	100	
9	Quinn ...	quinn....	16		100	100	100	80	60	60	100	100	100	63	100	60	57	100	
10	Blake A...	blake.a...	13		100	60	100	100	100	100	100	100	100	100	100	100	100		
11	Jesse Th...	jesse.th...	18		100	100	40	100	80	100	100	100	100	100	100	100	71	60	100
12	Robin W...	robin.w...	16		100	100	100	80	100	100	100	100	100	88	90	80	100	80	100
13	Jordan ...	jordan....	13		100	100	100	100	100	100	100	100	100	100	100	100	100		
14	Alex Car...	alex.car...	10		100	100	100	100	100	100	100	100	100	100					
15	Casey M...	casey.m...	17		100	100	80	100	100	100	80	100	100	75	100	100	43	60	50



	Column1	Column2	Column3	Column4	Column5	Add Column
1	Git	week 1	2023/08/14...	2023/08/18...	month 1	
2	Python	week 2	2023/08/21...	2023/08/25...		
3	Python	week 3	2023/08/28...	2023/09/01...		
4	Linux	week 4	2023/09/04...	2023/09/08...		
5	Linux	week 5	2023/09/11...	2023/09/15...	month 2	
6	Linux	week 6	2023/09/18...	2023/09/22...		
7	Networking	week 7	2023/09/25...	2023/09/29...		
8	Cloud Fnd	week 8	2023/10/02...	2023/10/06...		
9	IaC	week 9	2023/10/09...	2023/10/13...	month 3	
10	Docker	week 10	2023/10/16...	2023/10/20...		
11	K8s	week 11	2023/10/23...	2023/10/27...		
12	AWS	week 12	2023/10/30...	2023/11/03...		
13		week 13	2023/11/06...	2023/11/10...	month 4	
14		week 14	2023/11/13...	2023/11/17...		
15		week 15	2023/11/20...	2023/11/24...		
16		week 16	2023/11/27...	2023/12/01...		
17		week 17	2023/12/04...	2023/12/08...	month 5	
18		week 18	2023/12/11...	2023/12/15...		
19		week 19	2023/12/18...	2023/12/22...		

We created a MySQL database in AWS RDS



Here are some SQL queries that we used to specify the database to work on, display the tables in the database (which was empty initially), and create the schema for the database by creating tables named: Summary, Attendance and Assignment tables.

```
use mydb;

CREATE TABLE Summary (
ranking PRIMARY KEY,
name VARCHAR(255),
work_email VARCHAR(255),
personal_email VARCHAR(255),
attendance FLOAT,
assignment_completion VARCHAR(10),
quiz_submitted INT,
quiz_completed VARCHAR(255),
score INT,
compliance VARCHAR(255),
watcher VARCHAR(255)
);

CREATE TABLE Summary (
ranking INT AUTO_INCREMENT PRIMARY KEY,
cohort VARCHAR(64),
name VARCHAR(255),
email VARCHAR(255),
personal_email VARCHAR(255),
attendance FLOAT,
assignment_completion VARCHAR(20),
quiz_submitted INT,
quiz_completed VARCHAR (20),
score FLOAT,
compliance VARCHAR(255),
watcher VARCHAR(20)
);
attendance_average week present absent

CREATE TABLE Attendance (

);

show tables;
```