Domain:

TaskManagement-cc-A3-env.eba-zfyrgevb.ap-southeast-2.elasticbeanstalk.com

Architecture Document

TASK MANAGEMENT APP(AWS SERVICES)

Cloud Computing - 2024

Created by: Paulus David

s3[redacted]33

RMIT UNIVERSITY

Summary

The Task Management App is a simple online tool that helps users manage their tasks. Users can add, view, update, and mark tasks as done. The app also connects with Google Calendar, so users can easily sync their task deadlines to stay organised.

The app uses AWS services like Elastic Beanstalk to host the website, DynamoDB to store tasks and user data, S3 to save profile pictures, API Gateway for secure communication, and Lambda to handle the backend functions. CloudWatch is used to track failed login attempts. By using Google Calendar, the app makes it easy for users to keep track of deadlines.

This project uses cloud services to create a reliable and easy-to-use task manager for everyone.

Introduction

Motivation

The main motivation behind creating this project is to challenge myself and evaluate my programming skills by designing a simple, accessible, and free application that helps individuals track their tasks effectively. Since I am relatively new to Python programming, this project serves as a way for me to explore and learn the language. My primary focus is to build a functional app that fulfils its core purpose—allowing users to create, edit, and delete tasks—rather than expanding it with numerous features. This approach enables me to concentrate on delivering a working solution while gradually improving my programming capabilities.

What it does. High-Level View

The app allows users to register and create an account to log in and manage their tasks.

- **No Admin Features:** Since the app doesn't hold sensitive data beyond email and password (which are stored in plain text), there's no need for admin access or privileges.
- **Once Logged In:**
    - Users can see how many failed login attempts they've had in the last 5 minutes, shown on the Profile page. This is done by querying data from CloudMonitor.
    - The app greets users by displaying their username at the top of the page.
    - On the Profile page, users can update their username and default profile photo. However, email addresses cannot be changed.
    - Users can perform basic task management: adding, viewing, deleting, and marking tasks as either Complete or Incomplete.
    - Users may sync their tasks with google calendar which can serve as backup

The app focuses on being ***simple, easy to use****, and* ***functional***, with just the key features needed to manage tasks.

Beneficiaries

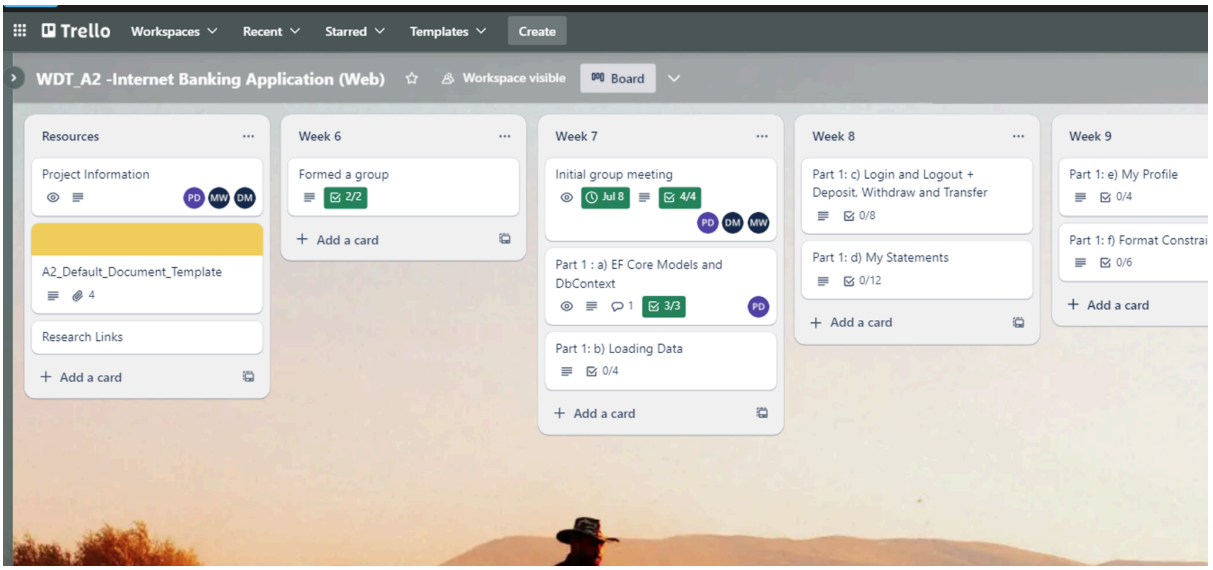
The primary beneficiaries of this application are:

1. **Individual Users:**
    - Anyone looking for a straightforward way to organise and manage their daily tasks.
    - Users who value a simple, distraction-free tool to create, edit, and track tasks without unnecessary features or complexity.
2. **Students and Professionals:**
    - Individuals managing multiple deadlines, assignments, or projects who need a lightweight solution to stay organised.
    - Especially useful for those who prefer a web-based task management system they can access from anywhere.
3. **Beginner Programmers and Developers:**
    - Those learning about cloud-based applications can use this as a reference for understanding how AWS services like Lambda, DynamoDB, and CloudWatch work together in a real-world application.

By keeping the application lightweight and focused on core functionalities, the app serves as an ideal solution for users who need a reliable and accessible tool to manage tasks efficiently.

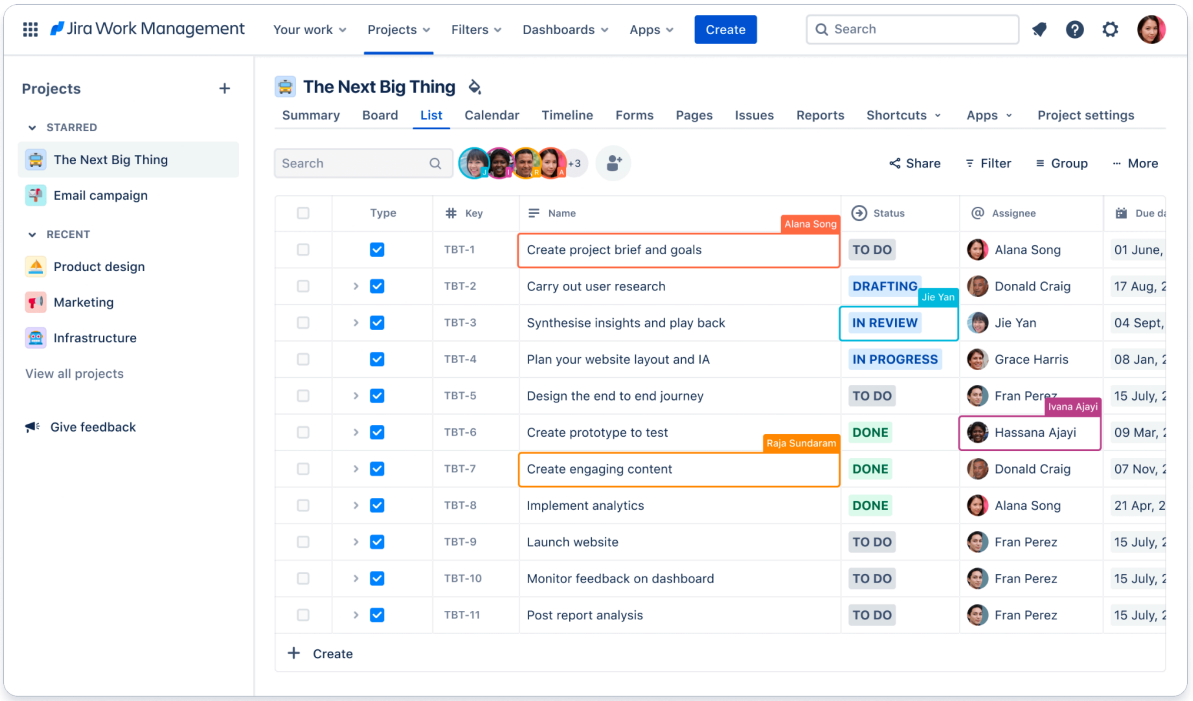
Related work: Trello

Trello shares key similarities with my application, as it also allows users to create, delete, and edit tasks. While Trello has been a popular tool for many years, its extensive features and complexity can sometimes overwhelm users who only need basic task management. This is why I chose to develop a simpler task management app that focuses solely on core functionalities. My goal was to create a straightforward and user-friendly solution for tracking tasks without unnecessary distractions.



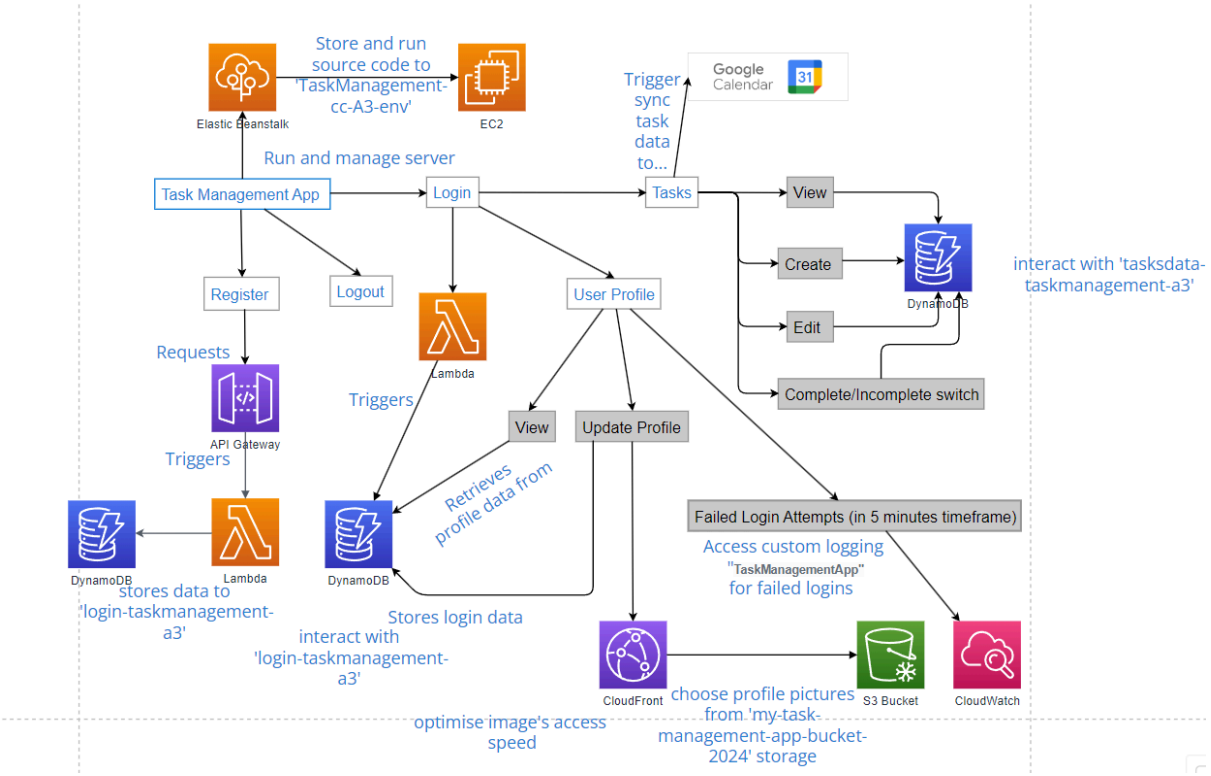
Jira

A project management application that consists of similar core functionality of Trello however is far more advanced as it supports graph representations, cleaner GUI, despite being similar its core functionality is the appeal of the creation of this project.



System Architecture

System Overview



System Components

The following sections provide an overview of the various services utilised to host and operate the Task Management web application effectively.

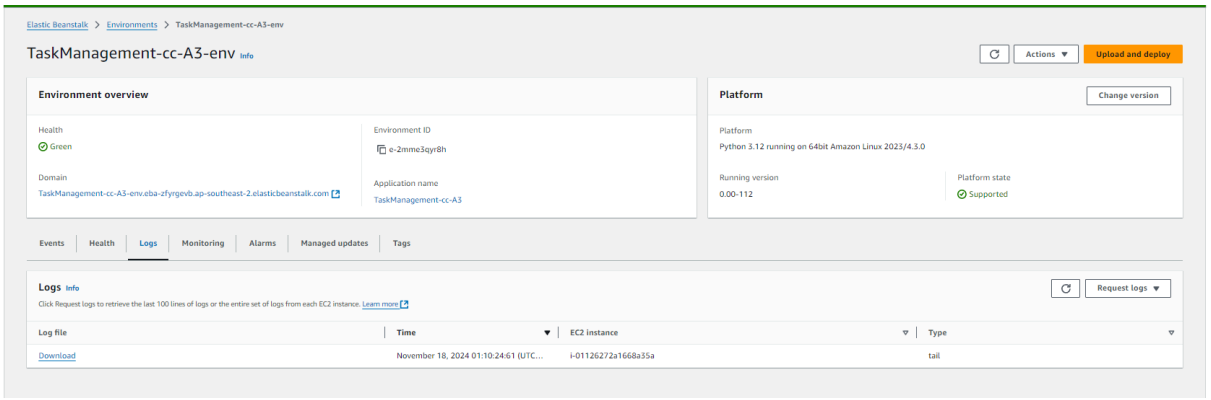
Elastic Beanstalk (EB)

Elastic Beanstalk is one of the key services in my Task Management app that simplifies the hosting and deployment process. By handling server setup, scaling, and deployment management automatically, Elastic Beanstalk allows me to focus on developing the app itself instead of managing its infrastructure. This service streamlines the operational side of the application, making it more efficient and easier to maintain.

How it's used in the app:

- **Quick Setup:** Elastic Beanstalk simplifies the initial setup process by automatically provisioning the servers and necessary resources to host the application. This saves time and effort, particularly during development and testing stages.
- **Traffic Management:** The service scales resources automatically depending on the app's traffic. For instance, during peak usage, Elastic Beanstalk increases server capacity to handle the load and reduces it during off-peak times to save costs.
- **Built-In Monitoring:** Elastic Beanstalk integrates seamlessly with CloudWatch, providing real-time health monitoring and troubleshooting options. This ensures that any performance or operational issues are identified and resolved quickly.
- **Integration with Other Services:** Elastic Beanstalk connects easily with other AWS services used in this project, such as S3 for storing profile pictures, DynamoDB for database management, and CloudFront for faster content delivery.

Overall, Elastic Beanstalk acts as the backbone for running the Task Management app by handling the technical details behind hosting and scaling.



Beanstalk configuration reference

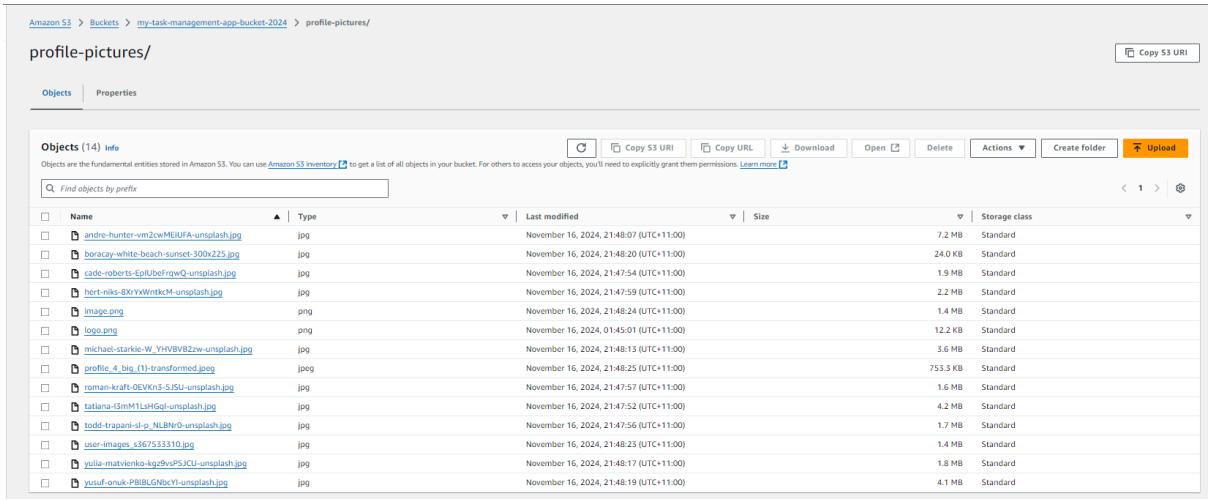
Simple Storage Service (S3)

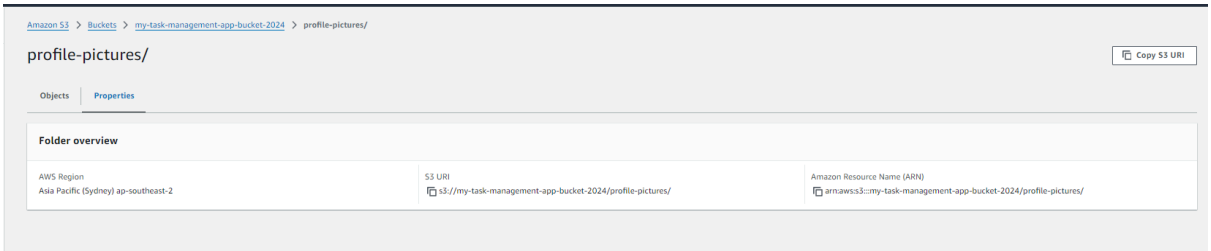
Amazon S3 is an essential storage service in my Task Management app, primarily used for saving user profile pictures. It ensures secure, scalable, and highly available storage, which is critical for managing media content efficiently. S3's simplicity and integration with other AWS services make it a reliable choice for this application.

How it's used in the app:

- **Profile Picture Storage:** When users upload or update their profile pictures, the images are stored in an S3 bucket. This ensures that the images are kept secure and can be accessed reliably whenever needed.
- **URL-Based Access:** Each uploaded image is stored as a unique URL, which is referenced in the app's database. These URLs are dynamically displayed on the user's profile page.
- **Seamless Integration with CloudFront:** To enhance performance, S3 works in tandem with CloudFront to deliver stored images quickly to users. This reduces load times and improves the overall user experience.
- **Scalability:** As the app grows and more users upload their profile pictures, S3 automatically scales to handle the increased data storage without requiring manual intervention.

Amazon S3 plays a crucial role in ensuring that user-generated content is handled effectively while maintaining high performance.





S3 Image Storage for default photos and bucket location

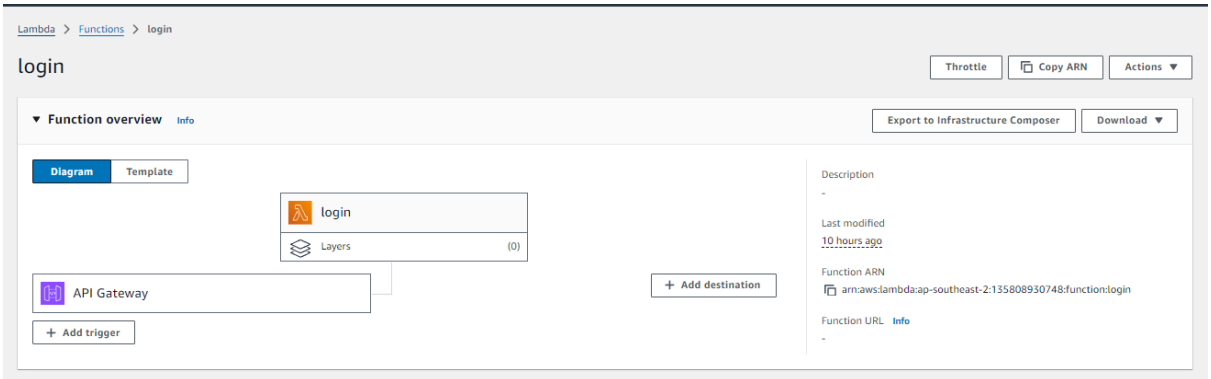
AWS Lambda

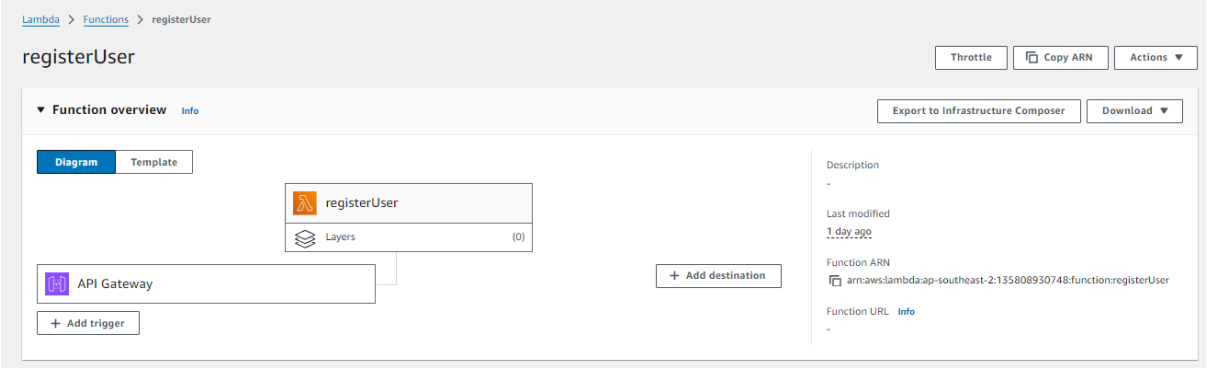
AWS Lambda provides serverless computing capabilities for my app, handling various backend operations efficiently without the need for dedicated servers. Lambda is event-driven, meaning it runs specific functions whenever triggered by user actions, which makes it a cost-effective and flexible solution.

How it's used in the app:

- **Login and Registration:** When users log in or register, Lambda functions validate their credentials or create new user accounts in DynamoDB. This ensures secure and reliable authentication processes.
- **Task Management:** Lambda handles all CRUD (Create, Read, Update, Delete) operations for tasks, communicating with DynamoDB to manage task data effectively.
- **Failed Login Tracking:** Lambda works with CloudWatch to process and track failed login attempts, displaying the count in real-time on the user's profile page.
- **Profile Updates:** When users update their profiles, such as changing their usernames or uploading new profile pictures, Lambda processes these updates and ensures the data is stored correctly.

AWS Lambda simplifies backend operations by automating essential tasks and scaling effortlessly with user demand, making it a cornerstone of the app's architecture.





Lambda function for login and register

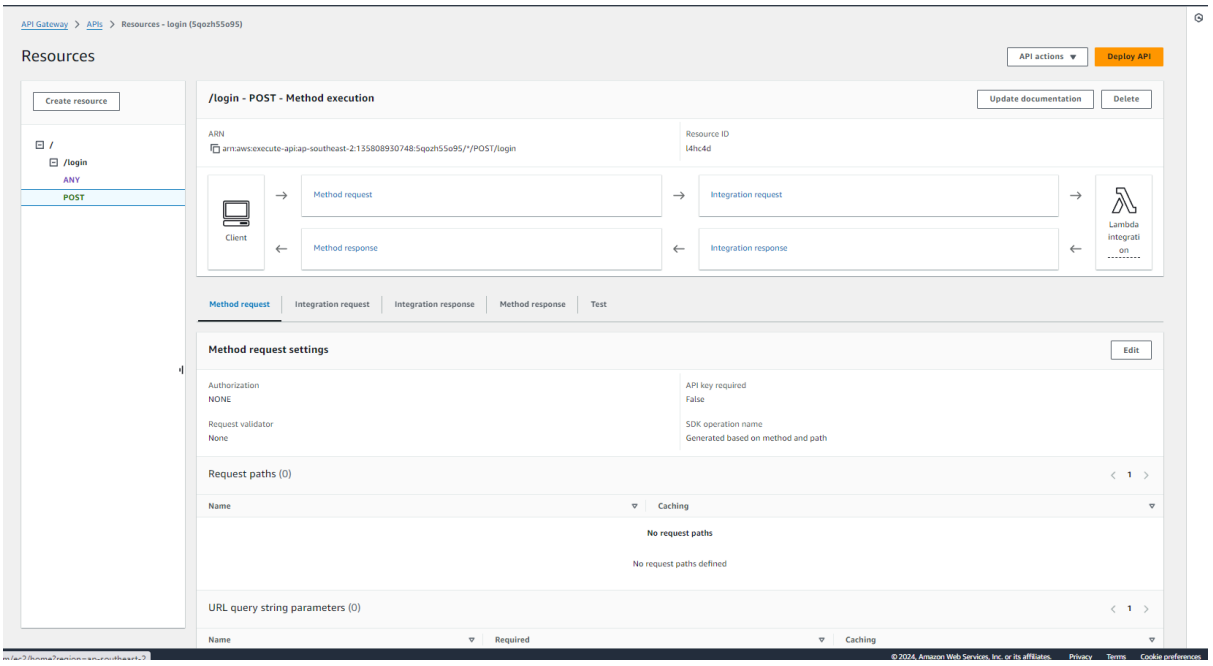
Amazon API Gateway

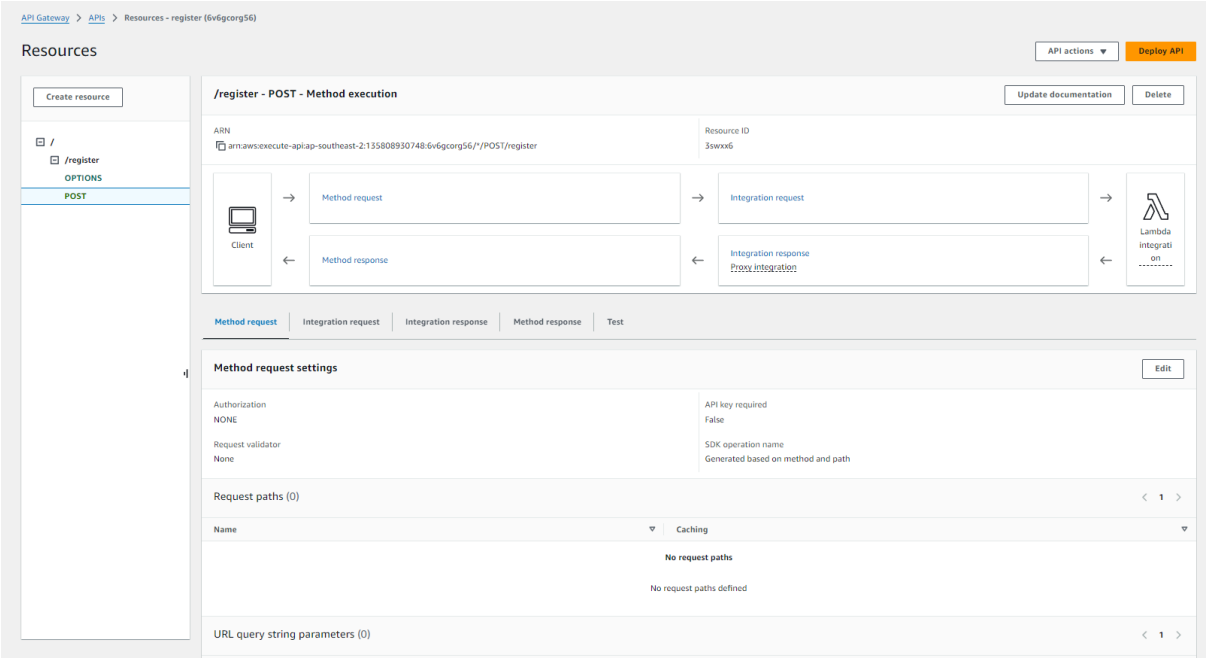
Amazon API Gateway acts as a central interface that connects the app's frontend with its backend services. By securely routing user requests to the appropriate components, API Gateway ensures smooth and reliable communication across the system.

How it's used in the app:

- **Routing Login and Registration Requests:** API Gateway forwards login and registration data from the frontend to Lambda functions for processing. It ensures secure and accurate validation of user credentials.
- **Task Operations:** All task-related actions, such as adding, updating, deleting, and viewing tasks, are routed through API Gateway. These requests are passed to Lambda for execution and then stored in DynamoDB.
- **Profile Updates:** API Gateway manages user profile-related updates, ensuring that username changes or profile picture uploads are processed seamlessly between the frontend and backend.
- **Failed Login Queries:** The service also facilitates queries to CloudWatch through Lambda, enabling users to view failed login attempts in their profile.

Amazon API Gateway ensures that all user interactions are processed securely and efficiently while providing a structured pathway for data to flow between components.





API Gateway (‘POST’) for register and login resources

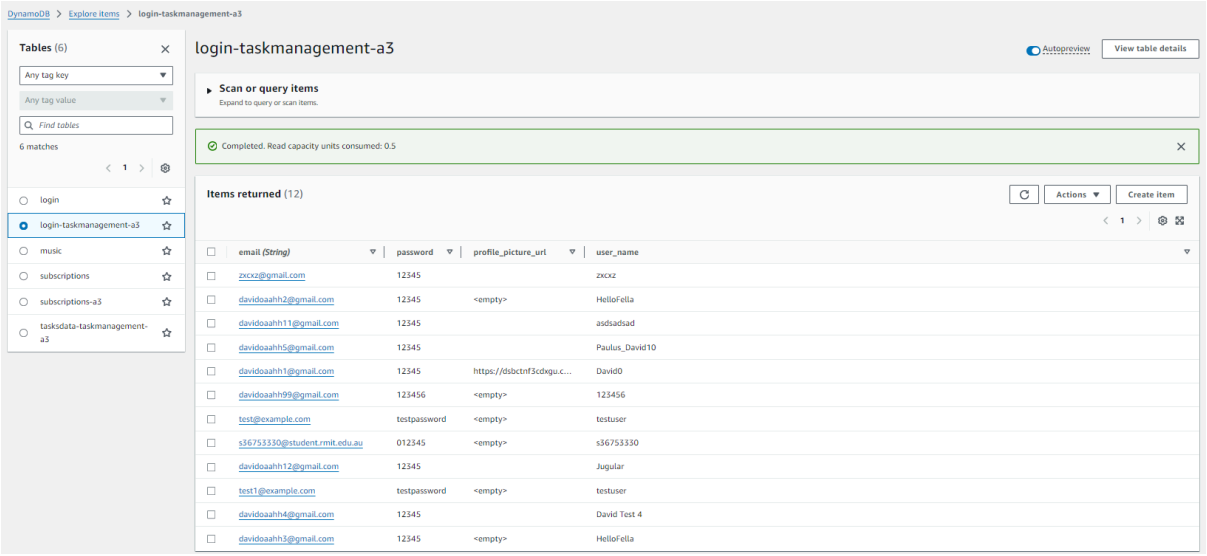
Amazon DynamoDB

DynamoDB serves as the primary database for my app, storing all critical user and task data. It’s a NoSQL database that is highly scalable and offers fast performance, which is perfect for handling the app’s data-intensive operations.

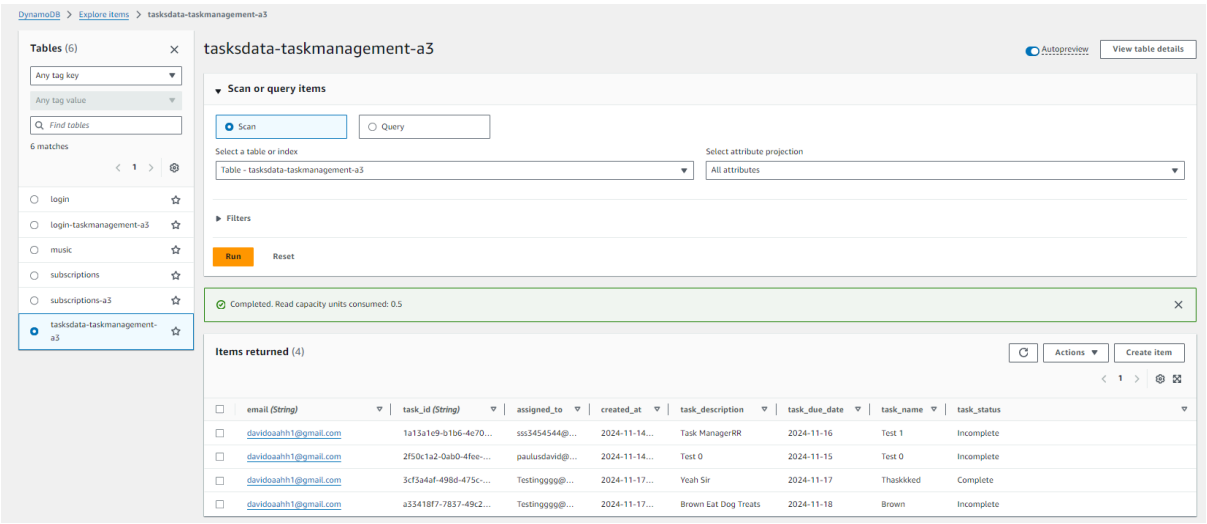
How it’s used in the app:

- **User Authentication:** DynamoDB securely stores user credentials, such as email addresses and passwords, which are validated during login and registration processes.
- **Task Storage:** Each task created by a user is stored in a DynamoDB table. Task details, including name, description, due date, status, and timestamps, are managed efficiently.
- **Failed Login Tracking:** Failed login counters are maintained in DynamoDB, ensuring real-time updates whenever a user experiences an unsuccessful login attempt.
- **Profile Data Management:** User profile information, such as usernames and profile picture URLs, is stored and updated dynamically within DynamoDB.

DynamoDB’s speed and flexibility make it a reliable database solution for handling diverse data needs in the Task Management app.



login-taskmanagement-a3 is used to store login data used for login and registrations



taskdata-taskmanagement-a3 is used to store tasks created by users

Amazon CloudWatch

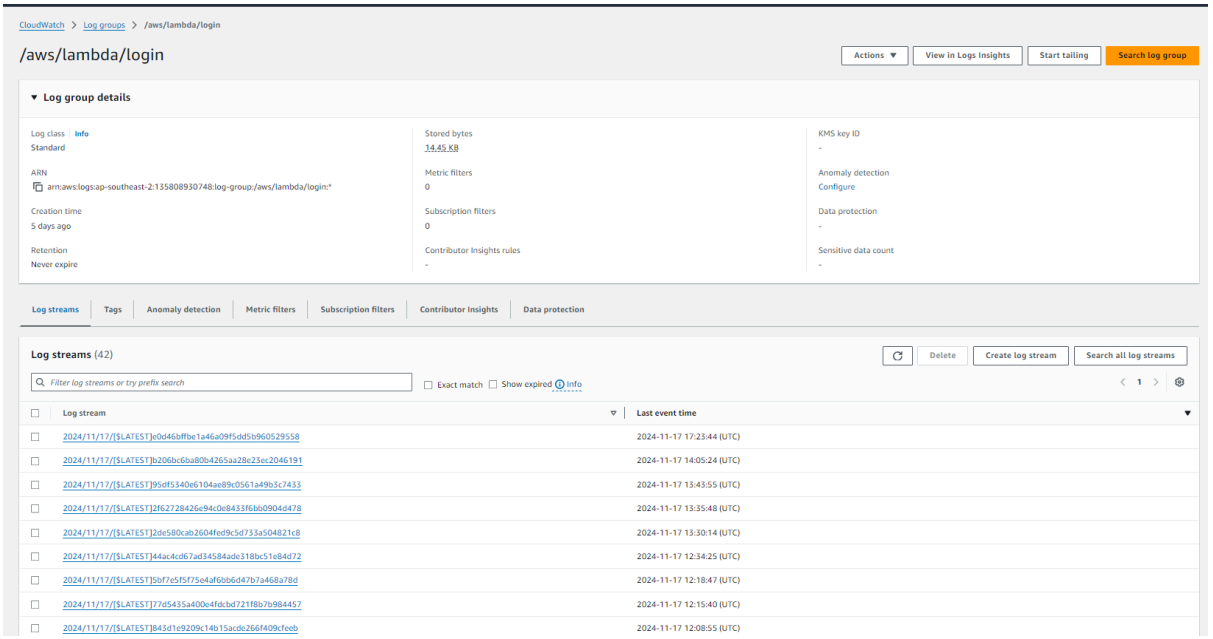
Amazon CloudWatch plays a crucial role in maintaining the app's stability by monitoring its performance and collecting important logs and metrics. This helps ensure that the system remains functional and any issues are addressed promptly.

How it's used in the app:

- **Failed Login Monitoring:** CloudWatch logs track failed login attempts, providing users with a counter that is updated in real-time on their profile page.
- **Error Tracking:** System errors, such as issues in Lambda functions, are logged in CloudWatch. This helps in diagnosing and resolving problems efficiently.

- **Performance Metrics:** Metrics like API Gateway latency, Lambda execution times, and DynamoDB usage are monitored to ensure optimal performance.
- **Alerts:** Although not implemented currently, CloudWatch can be configured to send alerts for unusual activity or errors, providing additional security and reliability.

CloudWatch ensures the app runs smoothly by providing detailed insights into its operations and performance.



User’s metrics

Google Calendar API

The Google Calendar API adds an extra layer of functionality to the Task Management app by synchronising task deadlines with users’ Google Calendars. This feature makes it easier for users to manage their schedules effectively across platforms.

How it’s used in the app:

- **Task Deadline Sync:** When users create or update tasks, they have the option to sync the task with their Google Calendar. This creates an event with the task details.
- **Cross-Platform Accessibility:** Users can view their tasks alongside other scheduled events in their Google Calendar, ensuring they never miss a deadline.

By integrating with Google Calendar, the app offers an additional convenience to users, making task management seamless and efficient.

References

[1] XALT GmbH, “Jira Work Management Dashboard and Overview,” [Online]. Available: https://www.xalt.de/wp-content/uploads/2023/08/xalt_jira-work-management-dashboard-und-overview.png. [Accessed: Nov. 18, 2024].

[2] Amazon Web Services, “Amazon Elastic Beanstalk,” [Online]. Available: https://aws.amazon.com/elasticbeanstalk/. [Accessed: Nov. 18, 2024].

[3] Amazon Web Services, “Amazon API Gateway,” [Online]. Available: https://aws.amazon.com/api-gateway/. [Accessed: Nov. 18, 2024].

[4] Amazon Web Services, “AWS Lambda,” [Online]. Available: https://aws.amazon.com/lambda/. [Accessed: Nov. 18, 2024].

[5] Amazon Web Services, “Amazon DynamoDB,” [Online]. Available: https://aws.amazon.com/dynamodb/. [Accessed: Nov. 18, 2024].

[6] Amazon Web Services, “Amazon S3,” [Online]. Available: https://aws.amazon.com/s3/. [Accessed: Nov. 18, 2024].

[7] Amazon Web Services, “Amazon CloudWatch,” [Online]. Available: https://aws.amazon.com/cloudwatch/. [Accessed: Nov. 18, 2024].

[8] Amazon Web Services, “AWS IAM (Identity and Access Management),” [Online]. Available: https://aws.amazon.com/iam/. [Accessed: Nov. 18, 2024].

[9] Amazon Web Services, “Amazon Certificate Manager,” [Online]. Available: https://aws.amazon.com/certificate-manager/. [Accessed: Nov. 18, 2024].

[10] Amazon Web Services, “Amazon QuickSight,” [Online]. Available: https://aws.amazon.com/quicksight/. [Accessed: Nov. 18, 2024].

[11] Amazon Web Services, “Amazon API Gateway - Method Settings,” [Online]. Available: https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-method-settings-method-request.html#:~:text=To%20set%20up%20an%20API,on%2C%20to%20its%20parent%20identifier. [Accessed: Nov. 18, 2024].

[12] SitePoint, “AWS CloudFront Tutorial: Setup and Configuration,” [Online]. Available: https://www.sitepoint.com/aws-cloudfront-tutorial-setup-and-configuration/. [Accessed: Nov. 18, 2024].

[13] Visual Paradigm, “Amazon's Diagram Creation Tool,” [Online]. Available: https://online.visual-paradigm.com/. [Accessed: Nov. 18, 2024].

[14] CloudZero, “What is the difference between AWS monitoring and observability?” [Online]. Available: https://www.cloudzero.com/blog/aws-monitoring/. [Accessed: Nov. 18, 2024].