AWS Task Management System - Documentation & Architecture Diagram

# 1. Project Overview

A cloud-native Task Management System hosted entirely on AWS. It enables users to sign up, create tasks, attach files, and receive notifications. The system uses a mix of serverless and traditional compute services, and is designed with high availability, scalability, and observability in mind.

# 2. Key AWS Services Used

|  |  |  |
| --- | --- | --- |
| Component | AWS Service | Purpose |
| User Authentication | Amazon Cognito | Manages user sign-up, sign-in, and access tokens |
| Application Hosting | Amazon EC2 | Hosts the frontend web application |
| Serverless Backend | AWS Lambda | Processes business logic for task operations |
| API Management | Amazon API Gateway | Manages RESTful endpoints |
| Relational Data Store | Amazon RDS | Stores user profiles and task relationships |
| NoSQL Data Store | Amazon DynamoDB | Stores task metadata |
| File Storage | Amazon S3 | Stores task attachments and uploaded files |
| Async Processing | Amazon SQS | Handles background task notifications |
| Monitoring & Logging | Amazon CloudWatch | Logs and monitors Lambda, EC2, API Gateway |
| Access Control | IAM | Manages access to AWS services |

# 3. Architecture Diagram Description

Horizontal Layered Design

\* User Layer  
 \* Users interact through the web interface.  
 \* Authentication handled via Amazon Cognito.  
  
\* Application Layer (Hosted in VPC)  
 \* Web app hosted on EC2 inside a public subnet.  
 \* IAM roles grant EC2 secure access to AWS resources.  
  
\* API & Serverless Logic  
 \* API Gateway exposes RESTful endpoints.  
 \* Lambda functions execute on demand to handle task logic.  
  
\* Storage Layer  
 \* Amazon RDS stores structured relational data.  
 \* DynamoDB stores flexible, fast-access metadata.  
 \* S3 stores uploaded file attachments.  
  
\* Asynchronous Processing Layer  
 \* SQS queues task update events.  
 \* Lambda functions read from SQS and send notifications.  
  
\* Monitoring & Logging  
 \* All components report metrics and logs to CloudWatch.

# 4. User Flow

1. User signs up or logs in via Amazon Cognito.  
2. User submits a task via the web frontend.  
3. Request hits API Gateway → triggers a Lambda function.  
4. Lambda:  
 - Stores task in DynamoDB  
 - Stores user-task link in RDS  
 - Uploads file to S3 (if present)  
5. If task is updated:  
 - Lambda sends a message to SQS  
 - Another Lambda reads from SQS and sends notification  
6. CloudWatch logs all activity and metrics

# 5. Security Considerations

\* IAM policies tightly scoped to limit access.  
\* VPC with public and private subnets to isolate resources.  
\* Cognito enforces secure sign-in and token usage.

# 6. Setup Guide

Prerequisites

\* AWS account  
\* IAM user with administrative access  
\* AWS CLI and AWS Management Console access

Step-by-Step Deployment

1. Create a VPC with public/private subnets.

2. Launch EC2 Instance:  
 - Install web server (e.g., Nginx)  
 - Deploy frontend code  
 - Assign IAM role to access S3, RDS, etc.

3. Set Up Amazon Cognito:  
 - Create a user pool  
 - Set up app client for frontend integration

4. Configure API Gateway:  
 - Create REST API  
 - Define resources and methods (GET, POST, PUT, DELETE)  
 - Integrate with Lambda functions

5. Create Lambda Functions:  
 - Write functions for CRUD operations  
 - Link each function to an API Gateway endpoint  
 - Grant permissions to access DynamoDB, RDS, S3, and SQS

6. Set Up DynamoDB & RDS:  
 - Create DynamoDB table for task metadata  
 - Launch RDS instance (e.g., PostgreSQL)  
 - Set up schema for users and task relationships

7. Enable S3 for Attachments:  
 - Create an S3 bucket  
 - Configure CORS and bucket policy

8. Add SQS for Async Tasks:  
 - Create a queue  
 - Create Lambda function triggered by SQS for notifications

9. Configure Monitoring:  
 - Enable CloudWatch logging for Lambda, API Gateway, and EC2  
 - Set up CloudWatch dashboards and alarms

10. Test End-to-End Flow:  
 - Sign up a user, create a task, attach a file, and trigger an update

# 7. User Manual

Logging In:  
- Navigate to the web app and sign in using your email and password.  
- If you don’t have an account, click on Sign Up to create one.

Creating a Task:  
1. Click on the Create Task button.  
2. Enter task title, description, due date, and priority.  
3. (Optional) Attach a file using the file upload field.  
4. Click Submit to create the task.

Viewing Tasks:  
- The dashboard displays a list of all your tasks.  
- You can filter by status (Pending, Completed, In Progress).

Editing a Task:  
1. Click the Edit icon next to the task.  
2. Update the fields and submit the changes.

Deleting a Task:  
- Click the Delete icon to remove a task permanently.

Receiving Notifications:  
- You will receive an email notification when tasks are updated or completed (if enabled).

Account Settings:  
- Go to your profile to update password or sign out.