DALHOUSIE UNIVERSITY

# CSCI 5409

# CLOUD COMPUTING

# TERM ASSIGNMENT

Full Name: Vishaka Vinod

Banner ID: B00955686

Dal Email: vs235403@dal.ca

GitLab: https://git.cs.dal.ca/courses/2024-winter/csci4145-5409/vvinod

# Table of Contents

# Introduction

## Objective

Growing up, I observed my mother, a dedicated teacher, encountering challenges in communicating with parents regarding their children's homework and assignments. The students would frequently forget to submit their work, and the parents seemed to be unaware of impending deadlines. This made me realize that there was a communication gap between the parents, teachers and students. There was also an absence of applications tailored to help solve this problem.

Young children, especially between the ages of 8 and 15, often find it difficult to keep track of and submit their work. I also noticed that the parents and teachers are not completely in sync with the tasks shared in the classroom. This could be due to a variety of reasons such as poor parent-teacher communication, busy schedules or simply that the child forgot to inform their parents. Recognizing this communication gap, I saw an opportunity to leverage the cloud services taught in this course to develop my project called ***TaskTrack***. The idea was to create a task tracking application that would empower children to take responsibility in managing their homework while also promoting efficient communication between the teachers and parents.

For the purposes of this term project, I opted to start with a more basic version of the application due to the complexity of implementing separate interfaces for teachers, parents, and students. The plan is to progressively enhance the application's functionality and expand it over time.

## Features

At present, TaskTrack is a single-user interface, catering to individuals seeking a simple and user-friendly application for task tracking and management. The following table outlines the application's current functionalities, and the planned additions for the future. It aims to help understand the decision behind choosing each service and feature.

*Table 1: Features table*

| Features | Current Scope | Future Scope |
|---|---|---|
| Task Management | Allows users to add and delete tasks. | The teacher would have the ability to add and delete homework and assignments. |
| Create New Task With File Upload | Caters to individuals who may receive task details in PDF format. | The teachers would be able to just upload a pdf document of the assignment details and it would get added into the task database. |
| Task Dashboard | Provides a centralized view of all tasks and their details. | Enhance dashboard with customizable widgets and advanced filtering and sorting options. |
| SNS Notifications | The user gets an email notification everytime a task is added. | The parents get an email notification everytime the teacher adds a task. Additionally, the parents get notified when a due date is approaching. |

## Technology Used

The frontend is created with React.js, while the backend API is built using Spring Boot. The application is hosted on an EC2 instance, and the database used is Aurora RDS MySQL.

### Performance Targets

The performance targets for ***TaskTrack*** include achieving the AWS pillars for a well architected framework:

- ***Operational Excellence:*** TaskTrack is designed to effectively support development and run application workloads. It also providing operational insights through monitoring capabilities via CloudWatch.
- ***Security:*** Recognizing the importance of data security, TaskTrack implements measures like password hashing using MD5 for encryption and authentication. These measures safeguard user privacy and prevent unauthorized access to sensitive data.
- ***Reliability:*** TaskTrack ensures responsiveness and quick user interactions for adding or deleting tasks, uploading files, and navigating the dashboard. Users can expect minimal delays during these actions, enhancing their overall experience.
- ***Performance Efficiency:*** TaskTrack optimizes computing resources to efficiently meet evolving demands and performance requirements, ensuring smooth operation under varying workloads.
- ***Cost Optimization:*** TaskTrack focuses to deliver business value by minimizing costs and efficient resource utilization.
- ***Sustainability:*** TaskTrack aims to utilizing sustainable resources throughout development, promoting environmental responsibility and long-term sustainability.

## Comparative Analysis

The AWS services EC2, Lambda, Aurora RDS, API Gateway, Textract, and Simple Notification System (SNS) were used to develop TaskTrack.

*Table 2: List of AWS Services used*

| AWS Service Categories | Service Selected |
|---|---|
| Compute | EC2 |
| | Lambda |
| Storage | Aurora RDS |
| | S3 |
| Network | API Gateway |
| General | SNS |
| | Textract |

### Compute Services: EC2 & Lambda

For the compute services in my project, I opted for Amazon EC2 to host my application because it offers better control over the infrastructure. This means I could choose the operating system, set up security measures, and install necessary software easily. Using EC2 Storage ensures quick access to data, fast disk operations, and minimal delays compared to other services like Elastic Beanstalk. This feature of EC2 suited my needs well, especially since I plan to scale the application and anticipate a large user base. Additionally, the costs associated with EC2 are predictable, and its flexibility, along with auto-scaling capabilities, would help reduce unnecessary expenses [1].

For the second compute service, I used AWS Lambda into my architecture to take advantage of its serverless computing features. It's ideal for executing tasks based on external triggers. I set up Lambda functions to respond to specific events, like file uploads or task creations. This approach takes off the responsibility of deploying and managing servers where the trigger code is located [2].

### Storage Services: Aurora & S3

I used the Aurora MySQL RDS database to store application data like user information and task details. It is a managed relational database service that helps in managing multiple database instances. It also provides advanced monitoring capabilities, for tracking the performance metrics [3]. While DynamoDB, AppSync, and Neptune appeared to be a reasonable alternative,

I preferred the functionality of a SQL database over a NoSQL or Graph database. Although I used S3 for file uploads, AWS Athena's querying capabilities were not required. As TaskTrack needed structured data storage and querying, Aurora MySQL RDS, with its relational model, was the better fit.

In addition to AWS Aurora, I integrated the Amazon S3 service to store user-uploaded files. I then configured a Lambda function to use Textract for data extraction from these files. The extracted data is subsequently stored in the database, and an SNS email notification is sent to the user.

### Network Services : API Gateway

For the network services, I chose API Gateway to create an endpoint for triggering a Lambda function, which sends an SNS notification when a user adds a task [4]. While this could have been achieved just through Lambda and RDS, I wanted to have more control on the scenarios where the lambda would be triggered. I achieved this by creating an endpoint via the API Gateway. I was able to differentiate Lambda function calls based on how tasks were inserted. For instance, if a user adds a task through the UI, the SNS Lambda is invoked via the API Gateway endpoint. If a task is inserted by uploading a file, the Textract Lambda is triggered.

### General: Textract & SNS

TaskTrack aims to offer users the flexibility to create and manage their tasks efficiently. Individuals with PDF files containing task details, can simply upload the file and have the data extracted and stored in the database. AWS Textract provides this functionality. It is simple to use, can be seamlessly integrates into the application, and requires no prior machine learning knowledge [5].

SNS, or Simple Notification Service, enables TaskTrack to send emails to the users whenever they create a task. This service can be automated using Lambda functions, making it easy to trigger notifications based on specific conditions or events. Additionally, SNS provides options for sending different types of notifications, such as SMS and push [6].

## Deployment Model

The deployment model for TaskTrack makes use of the public cloud, specifically AWS services provided by Amazon. This choice allows me to leverage the benefits of cloud computing without having to setup and manage the physical IT resources. By using this model, costs are incurred only for the resources being consumed, allowing for cost optimization. Additionally, deploying on a public cloud aligns with our goal of making TaskTrack accessible to all users.

## Delivery Model

TaskTrack is a software application that uses many AWS services. Each of these services follow different delivery models. For instance, EC2 operates under the Infrastructure as a Service (IaaS) model, while AWS Lambda functions adhere to the Function as a Service (FaaS) model. However, as a whole, TaskTrack chooses to adopt the Software as a Service (SaaS) delivery model. This is because TaskTrack is a fully-developed software application accessible to users across devices via the internet. It also does not require the users to make modifications or updates to the product. This makes the SaaS model the most suitable choice.

# Architecture

TaskTrack is a cloud-based application that seamlessly integrates AWS services to deliver a user-friendly, intuitive and efficient task management platform. Its architecture consists of essential components such as a frontend user interface, backend APIs, data storage capabilities, and notifications functionalities.

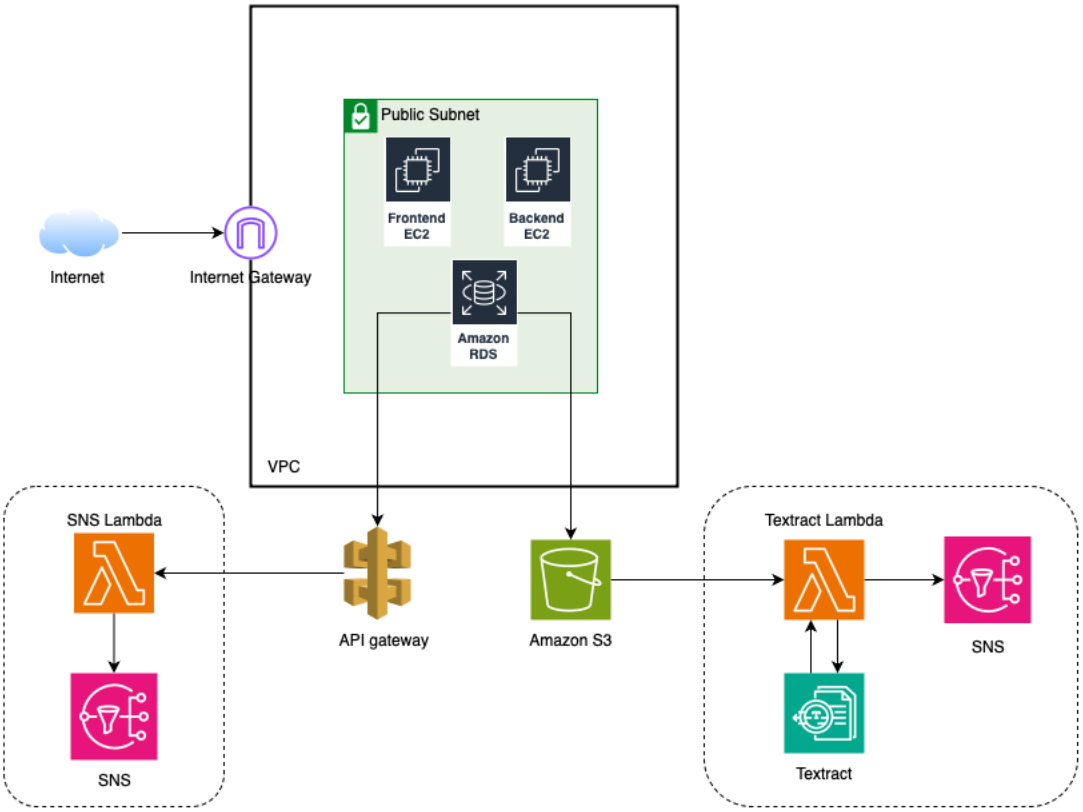


*Figure 1: Architecture of TaskTrack [7]*

## Application Flow

The user starts their TaskTrack journey by creating an account, followed by logging into the system using their newly created credentials. Upon successful login, users gain access to the task dashboard, along with a file drop box feature. For first-time users, the dashboard remains empty, while returning users find their tasks neatly displayed.

Within the task dashboard, users enjoy a range of functionalities, including the ability to add, delete, and sort tasks according to their priority level. The user can add a task in one of two ways: they can either select the 'add' icon within the dashboard or simply upload PDF documents via the drop box feature. With each task addition, users receive instant SNS email notifications, ensuring timely updates and enhanced task management efficiency.

## Architecture Flow

In Figure 1, the frontend of the application is hosted on the 'Frontend EC2,' while the backend is deployed on the 'Backend EC2' instance. Additionally, the Aurora RDS database is hosted within the same Virtual Private Cloud (VPC) and public subnet as the EC2 instances. This architectural decision is driven by the current non-critical nature of the stored user and task information. Moreover, security measures like password hashing and user authentication have been implemented.

The database is linked to both an API Gateway and an S3 bucket. When a user clicks on the 'add' button in the dashboard to create a new task, the API Gateway endpoint is called which triggers the SNS Lambda function. At the same time, an email notification is sent to the user, confirming the successful addition of the task via the user interface. Similarly, when a user creates a task by uploading a PDF document, the file is uploaded to the S3 bucket. Upon successful upload, the Textract Lambda function is triggered. This function uses Textract to get the contents of the file. It sets the first line as the task name and the rest of the text as the task description. The extracted data is then inserted into the Aurora RDS database. It also sends an SNS email notification to inform the user of the successful extraction and addition of the file to the database.

### Data Storage

The user data and task details are stored in Aurora RDS. When a user uploads a task file, the file is stored in the S3 bucket within a folder named after the user ID. The extracted data is inserted into the Aurora RDS database.

### Programming Languages Used

The frontend application is developed using React.js due to its widespread popularity and flexibility. React promotes code reusability by allowing developers to create individual components for different parts of the webpage. It combines HTML, CSS, and JavaScript to be integrated into a single JSX file. Additionally, React provides a lot of resources, documentation, and community support, making it an ideal choice for beginners [8].

The backend application and APIs are created using Spring Boot, a popular open-source Java framework. Spring Boot simplifies web application and API development, making it easy and efficient. It offers extensive resources and documentation, enabling developers to enhance their skills and build powerful backend systems. Additionally, Spring Boot provides built-in features for security, database connectivity, and dependency management, further streamlining the development process and ensuring robust functionality [9].

The Lambda functions are implemented using the Python programming language. Python is chosen because it's one of the primary languages used by AWS Lambda developers, along with Node.js. Additionally, Python is easy to learn and provides extensive library support, which helps reduce coding efforts. Moreover, the AWS Lambda allows Python developers to directly write and deploy code within the AWS console. According to DataDog [10], AWS Lambda cold starts are shortest when using Python, with Java experiencing 2.7 times longer cold starts. Furthermore, Python consumes less memory, contributing to improved performance and efficiency within AWS Lambda functions [10].

The CloudFormation script used to provision all AWS services was composed in a YAML file, following to the templates outlined in the AWS documentation [11].

## Data Security

TaskTrack has implemented multiple layers of security to ensure that the data is kept safe. The user data is securely stored in Aurora RDS, which is an AWS storage service that offers three levels of security. The first level involves the use of Identity and Access Management (IAM) to regulate access to the database. This means that only authorized users with appropriate permissions can interact and modify with the data in RDS. Secondly, Aurora RDS instances are always created within a Virtual Private Network (VPC), restricting access

only to EC2 instances within that VPC and any other configured resources. This setup prevents unauthorized access from external sources and enhances the overall security of the database. Finally, user authentication adds another layer of security. Aurora RDS requires an IAM role to provide permissions, ensuring that only authenticated users with the necessary credentials can access and manipulate the data.

Moreover, the spring boot application further protects user data by encrypting the user passwords with the MD5 hashing algorithm before storing it into the database.

Additionally, the Spring Boot application code further protects user data by encrypting passwords using the MD5 hashing algorithm before storing them in the database. This cryptographic technique ensures data security by safeguarding user credentials from potential breaches and unauthorized access. All APIs are designed following the REST architectural style, offering several advantages and ensuring security.

### Potential Vulnerabilities

No system can guarantee 100% safety. As developers serving real-world clients, prioritizing security should be the utmost priority. While TaskTrack implements various security measures, there are definitely areas for improvement.. Starting with securing the communication between the backend, frontend and cloud by using the secure HTTPS protocol that encrypts any data in transit. In the architectural design, placing the RDS instance into a private subnet within the VPC enhances security by reducing the risk of unauthorized access as it would not be directly accessible from the internet. TaskTrack could also improve how it handles session expiration to further improve security and user experience. This could be done by invalidating sessions effectively once users log out.

### Security Mechanisms

- **Virtual Private Cloud (VPC) Configuration:**
  Aurora RDS instances are deployed within a VPC, limiting access to only EC2 instances in the VPC and any other specified resources.
- **Aurora RDS:**
  IAM controls access to the Aurora RDS database, ensuring that only authorized users with the necessary privileges can interact with the data. This approach uses AWS IAM, an Amazon Web Services (AWS) solution for securely managing access to AWS resources.
- **Password Encryption with Spring Boot:**
  User passwords are encrypted with the MD5 hashing technique before they are stored in the database. MD5 stands for message-digest algorithm and is used to hash passwords and other sensitive information.

## Cost Analysis

### Reproducing Current Architecture to On-Premises

To transition our current architecture to an on-premises private cloud architecture, several key factors need to be taken into consideration. This includes the investment in hardware and software infrastructure, provisioning databases and networking components, configuring logging and monitoring tools, and establishing a robust backup and recovery mechanism. An estimate can be seen in Table 3. All the costs are in Canadian dollars.

*Table 3: Cost for moving from cloud to on-premises*

| Hardware Infrastructures | | Description | Potential Costs |
|---|---|---|---|
| **Servers** | Physical Servers | Dedicated servers are can be rented or owned to host the full application. | $100 to $200/month |
| **Storage Devices** | Network Attached Storage (NAS) or Storage Area Network (SAN) | Invest in high-performance storage systems, such as NAS or SAN solutions, to match Aurora RDS's capabilities. | $2500 to $5480/system |
| **Network Equipments** | Switches, routers | These eqipments will provide a network infrastructure for communication between servers and clients. | $1750 to $2500/month |
| **Software Infrastucture** | | | |
| **Operating System** | Windows, Mac, Linux | Some OS are free while others charge as they are proprietery like Windows OS. | $0 to $800/system |
| **Database Software** | MySql Workbench, RDBMS | This allows to create and store user data. | $0 to $200/month |
| **Networking and Security Software** | Firewall and Intrusion Detection Systems, VPNs | Install firewall and VPNs to monitor and protect network traffic and prevent unauthorized access. | Firewalls: $700 to $800/month<br>VPNs: $9 to $15/month |
| **Monitoring and Management Tools** | | | |
| **Monitoring Software** | Splunk | To monitor the performance and health of the on-premise infrastructure. | $180/month |
| **Backup and Disaster Recovery Solutions** | | | |
| **Backup & Recovery Software** | SAS Viya, CData Sync, etc | To achieve equal availability to the AWS cloud failover procedures. | $80 to $1900/month |

Apart from the factors mentioned in the table, there is also additional costs such as hiring skilled employees, server maintenance, cost for physical location to name a few.

## Cost Metrics for Current System

The current cost analysis reveals that operating TaskTrack incurs an annual expense of approximately $265,544.04. While this amount may seem to be very large, it is important to recognize that these costs are incurred under the assumption of TaskTrack operating at full scale application with thousands of users. There is also the possibility to incur additional charges based on usage, support required, and the demands of the application.

*Table 4: Cost Metrics Data for Current System*

| Service | Cost per Month ($) | Total Cost per Year ($) |
|---|---|---|
| EC2 | 12.1 | 145.2 |
| RDS | 3,281.70 | 39380.4 |
| S3 | 0 | 0 |
| Lambda | 0.03 | 0.36 |
| API Gateway | 18732.86 | 224794.32 |
| Textract | 0 | 0 |
| SNS | 101.98 | 1223.76 |
| **Total Cost per Year ($)** | | 265544.04 |

As shown in Table 4, the most expensive service would be the API gateway. This high cost is primarily attributed to the substantial number of requests made each month. In future iterations, there are opportunities to mitigate these expenses by exploring alternative solutions. For example, replacing the API Gateway with a custom-built API solution could potentially increase cost savings.

## Cost Monitoring

Based on the cost analysis provided, the API Gateway and the Aurora RDS are the cloud services with the greatest potential to exceed the budget. API Gateway handles API requests, and its charges are based on request volume and data transfer rates. Similarly, RDS is responsible for data storage and incurs charges based on instance type, storage capacity, and data transfer rates. As a result, without any monitoring, there is a risk of exceeding the budget due to sudden increases in demand.

## Future Scope

In the future, TaskTrack aims to evolve its functionalities and capabilities to better meet the needs of its users, with a specific focus on enhancing interactions for teachers, parents, and students. The plan involves developing separate interfaces tailored for each stakeholder while retaining all current functionalities. For teachers, this means gaining the ability to upload and delete tasks and assignments, with automatic notifications sent to parents via SNS upon task creation. Students will have a dedicated dashboard to view tasks, along with an interface to upload assignments, triggering notifications to teachers upon submission.

TaskTrack intends to introduce advanced communication features, including real-time notifications, and alerts, particularly for instances where a student has not submitted an assignment and the due date is approaching. Using Amazon EventBridge, these notifications will ensure timely reminders to support students in meeting their deadlines.

To improve communication, TaskTrack intends to enhance the current Textract functionality, to extract text from handwritten submissions. This aims to provide teachers with accurate and legible text, thereby minimizing misunderstandings associated with handwritten assignments. This feature will benefit students with dysgraphia, a condition characterized by difficulties in handwriting. Finally, to enhance accessibility, TaskTrack plans to incorporate Amazon Polly to convert task descriptions into speech, benefiting students who may face challenges in reading from screens. These accessibility features will promote inclusivity and accommodate diverse learning needs within the TaskTrack platform.

# References


[1] A. Y. Ogun, "I chose EC2 Storage over EBS despite it's ephemeral — Here's Why…," *Medium.com*, May 20, 2023. [Online]. Available: https://medium.com/@ayogun/i-chose-ec2-storage-over-ebs-despite-its-ephemeral-here-s-why-377f3280f5c2. [Accessed: March 17, 2024].

[2] E. Mikac, "AWS Lambda vs EC2: Which to Use and When," *CBT Nuggets*, September 14, 2020. [Online]. Available: https://www.cbtnuggets.com/blog/certifications/cloud/aws-lambda-vs-ec2-which-to-use-and-when. [Accessed: March 17, 2024].

[3] "Amazon RDS features," *Amazon.com*. [Online]. Available: https://aws.amazon.com/rds/features/. [Accessed: March 25, 2024].

[4] "API Gateway use cases," *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-overview-developer-experience.html. [Accessed: March 27, 2024].

[5] "What is Amazon Textract?," *Amazon.com*. [Online]. Available: https://docs.aws.amazon.com/textract/latest/dg/what-is.html. [Accessed: March 27, 2024].

[6] "Amazon Simple Notification Service," *Amazon.com*. [Online]. Available: https://aws.amazon.com/sns/. [Accessed: March 27, 2024].

[7] "Flowchart maker & online diagram software," *Draw.io*. [Online]. Available: https://www.drawio.com. [Accessed: April 03, 2024].

[8] "Why You Should Use React.js For Web Development," *FreeCodeCamp*, February 18, 2021. [Online]. Available: https://www.freecodecamp.org/news/why-use-react-for-web-development/. [Accessed: April 09, 2024].

[9] "What is Java Spring Boot?," *IBM*. [Online]. Available: https://www.ibm.com/topics/java-spring-boot. [Accessed: April 09, 2024].

[10] "The State of Serverless," *Datadog*, June 2022. [Online]. Available: https://www.datadoghq.com/state-of-serverless/. [Accessed: April 09, 2024].

[11] "AWS CloudFormation," *Amazon.com*. [Online]. Available: https://aws.amazon.com/cloudformation/. [Accessed: April 02, 2024].