

Test Project

Cloud Computing

Independent Test Project Designers: Shawn Xu, Xiaowei Wang, Qiang Wang, Xu Chen
Independent Test Project Validator: Chit Boon Lee, SCM

Contents

Description of project and tasks	3
Background	3
Initial state	4
Three Stages	4
<i>Stage 1: Data Reception</i>	<i>4</i>
<i>Stage 2: Data Query and Service Development.....</i>	<i>5</i>
<i>Stage 3: Architecture Optimization and Continuous Improvement.....</i>	<i>5</i>
Tasks	5
Technical Details	6
Service Details.....	6
Gentle reminder	7

Contents

This Test Project consists of the following documentation/files:

1. WSC2024_TP53_main_document_actual_en
2. WSC2024_TP53_day1_actual_en
3. **WSC2024_TP53_day2_actual_en**
4. WSC2024_TP53_day3_actual_en

Description of project and tasks

In today's rapidly evolving digital age, data has become one of the most valuable assets for businesses. This is especially true in the healthcare sector, where advancements in telemedicine and personal health monitoring technology have made real-time collection and analysis of health data particularly important. Your company, as a leading unicorn in the fields of artificial intelligence and big data analytics, is dedicated to developing innovative solutions to help businesses and individuals better understand and utilize this data.

The core objective of the project is to develop a cloud-based intelligent analytics platform capable of processing and analyzing two key types of data sent via POST requests: blood pressure monitoring data and enterprise operation data. Real-time analysis of these two types of data is of immense value to healthcare service providers and business decision-makers.

Blood Pressure Monitoring Data: With the proliferation of wearable devices, blood pressure monitoring has become an essential part of personal health management. These devices can collect blood pressure data in real-time, including key indicators such as systolic and diastolic pressure. Real-time monitoring of blood pressure is crucial for preventing cardiovascular diseases, adjusting treatment plans, and improving patients' quality of life.

Enterprise Operation Data: The second type of data involves daily business operations, such as sales transactions, customer interaction records, and inventory changes. This data is critical for businesses to understand market dynamics, optimize inventory management, enhance customer service quality, and formulate sales strategies. Through in-depth analysis of this data, businesses can identify potential growth opportunities, streamline operational processes, and enhance competitiveness.

As a solutions architect, you are responsible for:

- **Resource Management:** Effectively control resource access, track AWS costs, and optimize resource allocation for cost-efficiency.
- **Network Architecture:** Establish secure and reliable connections to AWS services while minimizing public internet exposure. Implement robust network monitoring for visibility and security.
- **Data Security and Compliance:** Protect sensitive data through encryption, access controls, and regular backups. Ensure compliance with relevant regulations and maintain records of suspicious activities.

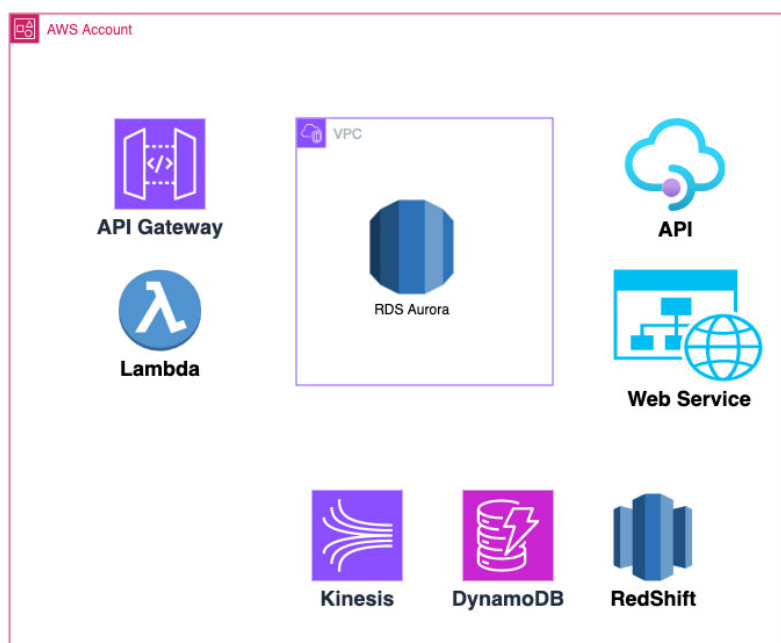
Background

The company intends to build a centralized, secure analytics platform capable of processing and analyzing high-volume, real-time data. This platform will provide intuitive visualizations to support data-driven decision-making.

By delivering robust data analytics solutions for healthcare and business operations, the company aims to accelerate digital transformation and create societal value. This project will solidify the company's leadership in AI and big data services, positioning it as a pioneer in data-driven insights and intelligent decision-making.

Initial state

Your primary responsibility is to design a scalable system capable of handling high volumes of real-time blood pressure and enterprise operation data delivered via POST requests. Successful data ingestion will initiate point accumulation.



The above image is a simplified architecture diagram for reference only.

Three Stages

Stage 1: Data Reception

Design and implement a data reception system capable of simultaneously handling two types of POST request data: real-time blood pressure monitoring data and enterprise operation data.

Blood Pressure Monitoring Data

- Utilize Amazon Kinesis for real-time ingestion and processing of blood pressure data.
- Employ Kinesis Data Analytics to analyze data streams and identify anomalies.
- Persist real-time anomaly data in Amazon DynamoDB for immediate access.
- Conduct in-depth analysis on historical anomaly data by storing it in Amazon Redshift.

Enterprise Operation Data

- Store enterprise operation data in an Amazon RDS (Aurora) database for reliable and efficient access. Bi-weekly backups are sufficient for the anticipated data volume. Serverless databases are not permitted.

Stage 2: Data Query and Service Development

Blood Pressure Monitoring Data

- Utilize the provided binary file (stub1.zip) located in the S3 bucket **s3://tp53-day2-game-assets-2024** to establish a web service.
- This web service will retrieve and present real-time blood pressure analysis results from DynamoDB in a user-friendly JSON format.
- Provide the web service URL for low-latency access to blood pressure insights.
- **Data Access Monitoring:** We strongly recommend capturing detailed request logs including timestamps, client IP addresses, latencies, request paths, and server responses for your web service. This data will be valuable for future analysis and optimization.
- **Backup and Versioning:** Upload the provided binary file to your competition account's S3 storage bucket. This bucket should ideally support efficient storage of multiple file versions for cost-effective lifecycle management.

Enterprise Operation Data

- Develop an API to facilitate user queries against the enterprise operation data stored in Amazon RDS (Aurora). This API will allow users to retrieve specific data points.
- Provide a secure and reliable API URL, protected against malicious attacks, for user access

Stage 3: Architecture Optimization and Continuous Improvement

Architecture Optimization: Align the system architecture with AWS Well-Architected Framework best practices to achieve optimal performance, cost-efficiency, security, reliability, scalability, and operational excellence.

Performance Monitoring: Implement robust monitoring, alerting, and evaluation processes to identify improvement areas and make necessary adjustments.

Tasks

1. Carefully review the task description document.
2. Use the us-east-1 region for your architecture setup.
3. Log in to the CloudRaiser platform and set up your name. Ensure to set the correct name.
4. Collect data through Kinesis and use Kinesis Data Analytics to store real-time analysis results in a DynamoDB database, ensuring it is cost-efficient and suitable for low-throughput applications.
5. Use Amazon Redshift to store the analyzed anomaly data, supporting in-depth analysis and historical data queries. Please DO NOT use serverless Redshift.
6. Store enterprise operation data in Amazon RDS (Aurora) database to ensure persistent storage and fast access.
7. Use the provided bin file to set up a web service that delivers the analysis results from DynamoDB in JSON format to end users.
8. Develop an API that offers query functionality for the enterprise operation data stored in Amazon RDS (Aurora), allowing users to retrieve the necessary data.
9. Monitor the performance of the web server to ensure service stability and response speed.
10. For detailed information on points, please refer to ScoreEvents.
11. Monitor costs to avoid unnecessary infrastructure expansion and minimize cost deductions.
12. Optimize and enhance the architecture to meet the six pillars of the AWS Well-Architected Framework and engineering practices, improving system performance efficiency, cost optimization, security, reliability, scalability, and operational excellence.
13. It is highly recommended to use ECS to build the web service. Consider enhanced flexibility, scalability, high availability, and fault tolerance for containerized applications, along with simplified infrastructure management and provides deep visibility into the configurations, performance and resource utilization of Amazon ECS containers.

14. It is highly recommended to create a CodePipeline to automate the deployment process of ECS applications, including a review step to maintain control during frequent upgrades. For efficient and reliable deployments, using CodeDeploy is strongly advised.

Technical Details

1. Server Application: The provided server application is a pre-compiled binary. Any modifications or use of unofficial binaries will result in disqualification.
2. EKS Restriction: EKS is prohibited for this competition.
3. Instance Management: Utilize SSM for instance management instead of SSH. The server application is optimized for x86 architecture and Amazon Linux 2.

Service Details

POST Data - Blood Pressure Monitoring Data:

The request will use **POST** method, request path is **post_data**:

Blood pressure monitoring data includes the following key information: no, systolic, diastolic, etc.

DataBase - Blood Pressure Monitoring Data

You need to deploy a highly available DynamoDB service to store analysis results. The table should be named "cloudraiser-iot" and have the following structure:

KEY NAME	DATA TYPE	COMMENT
No	String	Partition Key
Systolic	String	
Diastolic	String	
Score	String	

Web Service - Blood Pressure Monitoring Data

The development team has built a program for you to host the web service, which will provide the data stored in DynamoDB to customers. You need to consider how to build a highly available and scalable web service for customers at a lower cost.

Any data in DynamoDB that is older than 48 hours is not required and should be deleted.

POST Data - Enterprise Operation Data

The request will use **POST** method, request path is **post_data**:

Enterprise Operation Data includes the following key information: id, column, value, etc.

e.g.

```
{
  "id": "01"
  "column": "val1"
  "value": "456"
```

```
}

```

DataBase - Enterprise Operation Data

The database table structure is as follows: (The data type is "string")

id	val1	val2	val3	val4	val5	val6	val7	val8	val9	val10
----	------	------	------	------	------	------	------	------	------	-------

Web Service - Enterprise Operation Data

Please construct an API that returns a specific row of data based on the URL parameters of the REQUEST.

The request will use the GET method, and the REQUEST format is as follows:

[http\(s\)://xxx/get_value?id=xxx](http(s)://xxx/get_value?id=xxx)

The returned data structure is as follows: Format: JSON

```
{
  "id": "xxx",
  "val1": "xxx",
  "val2": "xxx",
  "val3": "xxx",
  "val4": "xxx",
  "val5": "xxx",
  "val6": "xxx",
  "val7": "xxx",
  "val8": "xxx",
  "val9": "xxx",
  "val10": "xxx"
}
```

xxx are just examples.

Gentle reminder

1. Post data will occur from the beginning of the match.
2. After approximately 2 hours, your system will be evaluated on its ability to process both blood pressure monitoring data and enterprise operation data, as well as its ability to deliver accurate results through the web service and API.
3. Points are awarded for successful achievements in the following areas:
 - (a) Receiving blood pressure monitoring data.
 - (b) Receiving enterprise operation data.
 - (c) Web service availability with correct result delivery.
 - (d) Web service responsiveness to user queries.
 - (e) Creation of resources conforming to architectural requirements.
 - (f) Architecture aligned with the six pillars of Well-Architected Frameworks.
4. Higher point totals are earned by correctly configuring more elements.

5. It is recommended to use Random Cut Forest for real-time analysis in Amazon Kinesis Data Analytics. Reference link: <https://aws.amazon.com/cn/blogs/big-data/real-time-anomaly-detection-via-random-cut-forest-in-amazon-kinesis-data-analytics/>

Reference Document:

- <https://docs.aws.amazon.com/>
- <https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html>
- <https://docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html>