

## Contents

Implementation flow .....	2
Document Overview .....	3
Purpose and Objective .....	3
Scope.....	3
Assumptions & Constraints.....	3
AWS Roles and Policies .....	4
Understanding the requirement.....	5
Understanding the source data .....	6
AWS Components Architecture .....	9
Technical Implementation of AWS services.....	9
Additional Features Integration.....	20
Implementation Resources .....	20

## Implementation flow

Sr. No.	Module	Coverage
1	<b>Amazon S3 Bucket</b> Creation for Data Storage	Provisioned and configured Amazon S3 buckets to securely store input datasets and processed output files. Ensured optimal organization and access control to support efficient data workflows and integration with downstream analytics and machine learning pipelines.
2	<b>AWS Glue</b> ETL Workflow for ML-Ready Data Preparation	Designed and implemented an AWS Glue ETL pipeline to apply business transformation rules, creating augmented datasets optimized for machine learning consumption. The ETL process output was stored in both Amazon S3 and Amazon RDS Aurora, enabling scalable storage and transactional support. Leveraged Amazon Athena for interactive data analysis and validation of transformed datasets.
3	<b>Amazon SageMaker</b> - Feature Engineering and Model Development	Utilized Amazon SageMaker for advanced feature engineering and machine learning model development. Leveraged SageMaker Notebooks to preprocess data, train, and validate machine learning models. Successfully deployed the trained model as a managed endpoint for seamless integration and real-time inference.
4	<b>AWS Lambda Function</b> for ML Endpoint Integration	Developed an AWS Lambda function to test, access, and deploy the machine learning endpoint. The function was designed to preprocess input data, invoke the ML endpoint for inference and handle responses efficiently. This streamlined integration with the ML endpoint ensured seamless deployment and scalability for real-time and batch processing use cases.
5	<b>Amazon API Gateway</b> for Application Integration	Configured Amazon API Gateway to serve as a scalable endpoint for the end application. The gateway routes incoming requests to the AWS Lambda function, enabling seamless preprocessing and invocation of the ML endpoint. This setup provides secure, low-latency

		access interaction with the deployed machine learning service.
6	<b>Postman</b> API Testing for ML Endpoint Validation	Utilized Postman to test the Amazon API Gateway endpoint by sending sample data inputs. Verified the functionality, response accuracy, and latency of the integrated ML processing pipeline, ensuring the endpoint's readiness and reliable interaction with the end application.
7	<b>IAM Roles and Policies</b> for Secure Managed Service Access	Defined and configured AWS Identity and Access Management (IAM) roles with policy-level permissions to enable secure access to managed services across the workflow. These roles ensured seamless integration and controlled access to resources such as S3, Glue, Lambda, API Gateway, and SageMaker while adhering to the principle of least privilege for enhanced security and compliance.

## Document Overview

### Purpose and Objective

This document highlights the capabilities of the AWS cloud environment in addressing healthcare challenges through managed services. The focus is on leveraging AWS's robust, scalable, and secure services to streamline data processing, enhance accessibility and improve healthcare solutions without delving into complex machine learning problems.

By prioritizing simplicity and efficiency, the outlined approach demonstrates how AWS managed services can optimize healthcare operations, enable seamless integration and support actionable insights with minimal technical overhead.

### Scope

This document demonstrates AWS Machine Learning capabilities through a sample use case of diabetes prediction based on patient test reports. The use case employs fabricated data to showcase how AWS platform services can address real-world healthcare challenges, enabling predictive analytics and decision support in a scalable and efficient manner.

### Assumptions & Constraints

**Purpose and Focus:** The primary goal of the assignment is to demonstrate the machine learning capabilities of AWS services using a practical use case. The exercise showcases the candidate's proficiency with AWS infrastructure, managed services, and related skill sets.

**Cost Management:** The AWS setup incurs an approximate cost of \$2 per day, emphasizing the need for efficient use of resources. To save costs, the environment will be deleted after execution and the implementation details will be documented through screenshots and a Git repository.

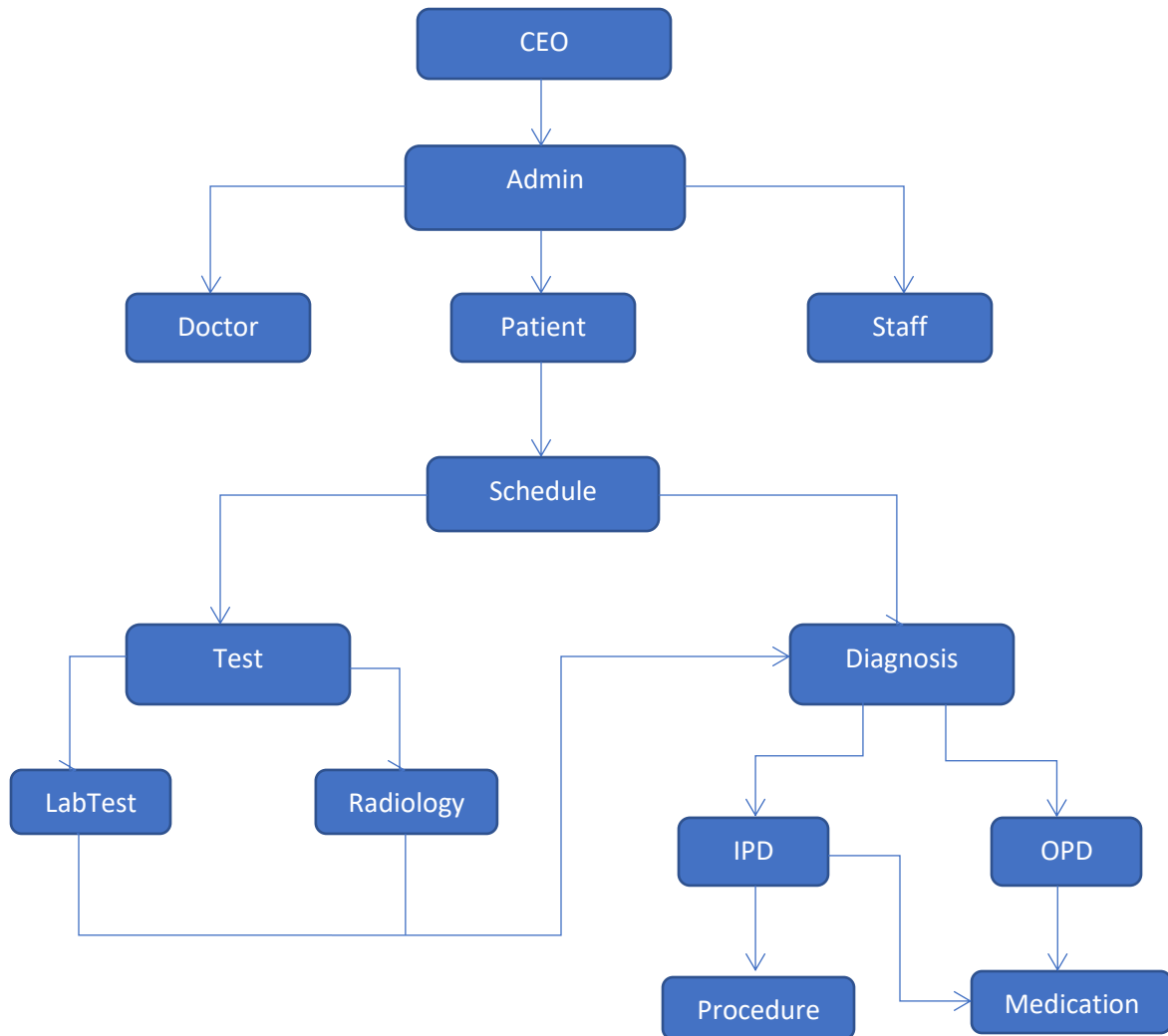
**Resource and Execution Scope:** The demonstration uses fabricated data for simulation and avoids deploying large-scale or resource-intensive ML solutions.

## AWS Roles and Policies

Name	Permission
Amazon SageMaker role	Amazon RDS Access, Amazon S3 Access, Sage Maker Access, SageMaker Execution Access, Amazon CloudWatch Logs Access
AWS Glue ETL Role	Amazon Athena Access, Amazon RDS Access, Amazon S3 Access, AWS Glue Service Access, Amazon CloudWatch Logs Access, AWS Glue Catalog Permissions
Lambda Function Role	Amazon S3 Access, Amazon Sage Maker Access, AWS Lambda Execution Access, Amazon CloudWatch Logs Access
Anthena Role	AWS Athena Spark Execution Access, Amazon S3 and GetObject Access, AWS Glue Catalog Permissions

# Understanding the requirement

## General Process Model



## Understanding the source data

**Table: Patient**

Column	Data Type	Reference Key
patient_id	Int	Primary Key
fname	String	
mname	String	
lname	String	
gender	String	
marital_status	String	
date_of_birth	Date	
aadhar_number	String	

**Table: patient\_labtest**

Column	Data Type	Reference Key
patient_labtest_id	Int	Primary Key
patient_id	Int	Foreign Key to Patient.patient_id
heart_test_id	Int	Foreign Key to heart_test.heart_test_id
blood_test_id	Int	Foreign Key to blood_test.blood_test_id
general_body_test_id	Int	Foreign Key to general_body_test.general_body_test_id

**Table: heart\_test**

Column	Data Type	Reference Key
heart_test_id	Int	Primary Key
patient_id	Int	Foreign Key to Patient.patient_id
heart_rate	Float	
heart_disease	Int	
blood_pressure	Float	

**Table: blood\_test**

Column	Data Type	Reference Key
blood_test_id	Int	Primary Key
patient_id	Int	Foreign Key to Patient.patient_id
glucose	Float	
insulin	Float	
hbA1c_level	Float	

**Table: general\_body\_test**

Column	Data Type	Reference Key
general_body_test_id	Int	Primary Key
patient_id	Int	Foreign Key to Patient.patient_id
skin_thickness	Float	
bmi	Float	
diabetes_pedigree_function	Float	
weight	Float	
height	Float	
smoking_history	String	
pregnancies	Int	
hypertension	Int	

**Table: diagnosis**

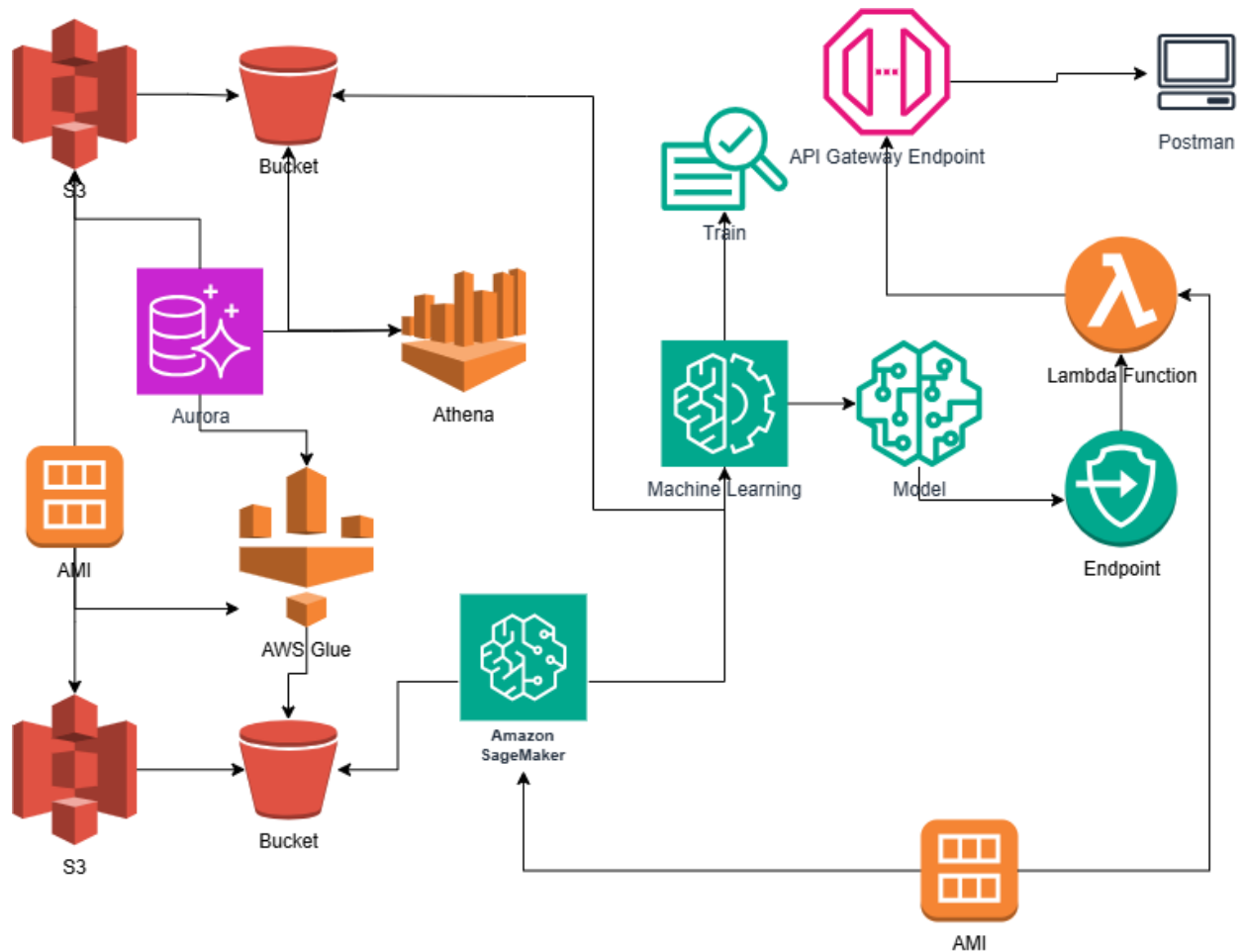
Column	Data Type	Reference Key
patient_id	Int	Foreign Key to Patient.patient_id
patient_labtest_id	Int	Foreign Key to patient_labtest.patient_labtest_id
is_diabetic	Int	

**Target Table: diabetes\_prediction**

Column	Data Type	Reference Key
gender	String	
hypertension	Int	
heart_disease	Int	
smoking_history	String	
hbA1c_level	Float	
pregnancies	Int	
glucose	Float	
blood_pressure	Float	
skin_thickness	Float	
insulin	Float	
bmi	Float	
diabetes_pedigree_function	Float	
age	Int	
<b>is_diabetic</b>	<b>Int</b>	



# AWS Components Architecture



## Technical Implementation of AWS services

### AWS S3 Bucket

laboratory-diagnostics-data [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

#### Objects (3) [Info](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	data-store/	Folder	-	-	-
<input type="checkbox"/>	OutputData/	Folder	-	-	-
<input type="checkbox"/>	SourceData/	Folder	-	-	-

Objects (7) Info

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

< 1

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	bloodtest/	Folder	-	-	-
<input type="checkbox"/>	diabeticprediction/	Folder	-	-	-
<input type="checkbox"/>	diagnosis/	Folder	-	-	-
<input type="checkbox"/>	generalbodytest/	Folder	-	-	-
<input type="checkbox"/>	hearttest/	Folder	-	-	-
<input type="checkbox"/>	patient/	Folder	-	-	-
<input type="checkbox"/>	patientlabtest/	Folder	-	-	-

Objects (1) Info

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

< 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	blood_test.csv	csv	November 27, 2024, 13:24:40 (UTC+05:30)	436.0 B	Standard

AWS RDS

RDS > Databases

Amazon RDS

Dashboard

Databases

Query Editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations

Consider creating a Blue/Green Deployment to minimize downtime during upgrades

Easy path homogeneous data migrations from EC2 database to RDS

Databases (2)

Filter by databases

<input type="radio"/>	DB identifier	Status	Role	Engine	Region	Size	Recommendations	CPU	Current
<input type="radio"/>	database-1	Available	Regional cluster	Aurora MySQL	us-east-2	1 instance	1 informational	-	-
<input type="radio"/>	database-1-instance-1	Available	Writer instance	Aurora MySQL	us-east-2b	db.r7g.large		4.93%	

RDS

Databases

database-1

Amazon RDS

Dashboard

Databases

Query Editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations

Events

Event subscriptions

Filter by databases

DB identifier

Status

Role

Engine

Region ...

Size

Recommendations

CPU

database-1

Available

Regional cl...

Aurora My...

us-east-2

1 instance

1 Informational

-

database-1-instance-1

Available

Writer inst...

Aurora My...

us-east-2b

db.r7g.large

5.56%

Connectivity & security

Monitoring

Logs & events

Configuration

Zero-ETL integrations

Maintenance & backups

Data migrations - new

Tags

Endpoints (2)

Find resources

Endpoint name

Status

Type

database-1.cluster-c1musyekgsng.us-east-2.rds.amazonaws.com

Available

Writer

database-1.cluster-ro-c1musyekgsng.us-east-2.rds.amazonaws.com

Available

Reader

Manage IAM roles

Select IAM roles to add to this cluster

Add an existing IAM role to this cluster.

RDS

Databases

database-1

database-1-instance-1

Amazon RDS

Dashboard

Databases

Query Editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations

Events

Event subscriptions

Recommendations

Certificate update

Filter by databases

DB identifier

Status

Role

Engine

Region ...

Size

Recommendations

database-1

Available

Regional cl...

Aurora My...

us-east-2

1 instance

1 Informational

-

database-1-instance-1

Available

Writer inst...

Aurora My...

us-east-2b

db.r7g.large

Connectivity & security

Monitoring

Logs & events

Configuration

Maintenance & backups

Tags

Recommendations

Connectivity & security

Endpoint & port

Networking

Security

Endpoint

Availability Zone

VPC

Subnet group

Subnets

Network type

VPC security groups

Publicly accessible

Certificate authority

Certificate authority date

DB instance certificate expiration date

AWS Glue – Data Connections (Catalog)

Data connections

Workflows (orchestration)

▼ Data Catalog

Databases

Tables

Stream schema registries

Schemas

Connections

Crawlers

Classifiers

Catalog settings

▼ Data Integration and ETL

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Interactive Sessions

Connectors (0) Info

You can manage your connectors or use them to create connections.

Q Filter connections by property

Name	Status	Type
No connectors		
No connectors to display. You can create a custom connector or <a href="#">get a Marketplace</a>		
<div>Create custom connector</div>		

Connections (1) Info

You can manage your connections or use a connection in a job.

Q Filter connections by property

Name	Status	Type
<input type="radio"/> <a href="#">Aurora_Healthcare</a>	<input checked="" type="checkbox"/> Ready	JDBC

AWS Glue - Crawlers

☰

AWS Glue

> Crawlers

AWS Glue

<

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

▼ Data Catalog

Databases

Tables

Stream schema registries

Crawlers

A crawler connects to a data store, progresses through a prioritized list of classifiers to determine the schema of the data.

Crawlers (2) Info

View and manage all available crawlers.

Q Filter crawlers

<input type="checkbox"/>	Name	State	Schedule	Last run
<input type="checkbox"/>	<a href="#">aroradb_connect</a>	<input checked="" type="checkbox"/> Ready		<input checked="" type="checkbox"/> Succeeded
<input type="checkbox"/>	<a href="#">s3_bloodtest</a>	<input checked="" type="checkbox"/> Ready		<input checked="" type="checkbox"/> Succeeded

AWS Glue

Crawlers

s3\_bloodtest

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

Data Catalog

Databases

Tables

Stream schema registries

Schemas

Connections

Crawlers

Classifiers

Catalog settings

Data Integration and ETL

ETL jobs

s3\_bloodtest

Last updated (UTC)  
November 29, 2024 at 10:20:45

Crawler properties

Name

s3\_bloodtest

IAM role

BlueRole

Database

healthcare

Description

-

Security configuration

-

Lake Formation configuration

-

Maximum table threshold

-

Advanced settings

Crawler runs

Schedule

Data sources

Classifiers

Tags

Data sources (1)

Info

Edit

The list of data sources to be scanned by the crawler.

Type	Data source	Parameters
<input checked="" type="radio"/> S3	s3://laboratory-diagnostics-data/SourceData/	Recrawl all

AWS Glue

Crawlers

aroradb\_connect

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

Data Catalog

Databases

Tables

Stream schema registries

Schemas

Connections

Crawlers

Classifiers

Catalog settings

aroradb\_connect

Last updated (UTC)  
November 29, 2024 at 10:24:13

Crawler properties

Name

aroradb\_connect

IAM role

BlueRole

Database

aroradb

SI

RI

Description

-

Security configuration

-

Table prefix

-

Advanced settings

Crawler runs

Schedule

Data sources

Classifiers

Tags

Data sources (1)

Info

Edit

The list of data sources to be scanned by the crawler.

Type	Data source	Parameters
<input checked="" type="radio"/> JDBC	healthcare/%	-

## AWS Glue – Databases

AWS Glue

Databases

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

Data Catalog

Databases

Tables

Stream schema registries

Databases (3)

Last updated  
November 29, 2024 at 10:

A database is a set of associated table definitions, organized into a logical group.

Filter databases

<input type="checkbox"/>	Name	Description	Location URI
<input checked="" type="checkbox"/>	aroradb	-	-
<input type="checkbox"/>	default	Default Hive database	file:/tmp/spark-warehouse
<input type="checkbox"/>	healthcare	This database is for healthcare analysis	-

## AWS Glue

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

### Data Catalog

#### Databases

Tables

Stream schema registries

Schemas

Connections

Crawlers

Classifiers

Catalog settings

### Data Integration and ETL

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Interactive Sessions

#### Announcing new optimization features for Apache Iceberg tables

Optimize storage for Apache Iceberg tables with automatic snapshot retention and orphan file deletion. [Learn more](#)

## healthcare

November 29, 2024 at 10:29:03

### Database properties

Name  
healthcare

Description  
This database is for healthcare analysis

Location  
-

### Tables (7)

Last updated (UTC)  
November 29, 2024 at 10:29:03

Delete

View and manage all available tables.

Filter tables

<input type="checkbox"/>	Name	Database	Location	Classification	Deprecated	View data
<input type="checkbox"/>	bloodtest	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	diabeticprediction	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	diagnosis	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	generalbodytest	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	hearttest	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	patient	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>
<input type="checkbox"/>	patientlabtest	healthcare	s3://laboratory-dia	CSV	-	<a href="#">Table data</a>

## AWS Glue

Getting started

ETL jobs

Visual ETL

Notebooks

Job run monitoring

Data Catalog tables

Data connections

Workflows (orchestration)

### Data Catalog

#### Databases

Tables

Stream schema registries

Schemas

Connections

Crawlers

Classifiers

Catalog settings

### Data Integration and ETL

ETL jobs

Visual ETL

Notebooks

Job run monitoring

#### Announcing new optimization features for Apache Iceberg tables

Optimize storage for Apache Iceberg tables with automatic snapshot retention and orphan file deletion. [Learn more](#)

## aroradb

Last updated (UTC)  
November 29, 2024 at 10:29:03

### Database properties

Name  
aroradb

Description  
-

Location  
-

### Tables (7)

View and manage all available tables.

Filter tables

<input type="checkbox"/>	Name	Database	Location	Classification	Deprecated
<input type="checkbox"/>	healthcare_blood_test	aroradb	healthcare.blood_test	mysql	-
<input type="checkbox"/>	healthcare_diabetes	aroradb	healthcare.diabetes	mysql	-
<input type="checkbox"/>	healthcare_diagnosis	aroradb	healthcare.diagnosis	mysql	-
<input type="checkbox"/>	healthcare_general_body_test	aroradb	healthcare.general_body_test	mysql	-
<input type="checkbox"/>	healthcare_heart_test	aroradb	healthcare.heart_test	mysql	-
<input type="checkbox"/>	healthcare_patient	aroradb	healthcare.Patient	mysql	-
<input type="checkbox"/>	healthcare_patient_lab_test	aroradb	healthcare.patient_lab_test	mysql	-


## AWS Glue - ETL

### AWS Glue


- Getting started
- ETL jobs
  - Visual ETL**
  - Notebooks
  - Job run monitoring
- Data Catalog tables
- Data connections
- Workflows (orchestration)
- ▼ **Data Catalog**
  - Databases
  - Tables
  - Stream schema registries
  - Schemas
  - Connections
  - Crawlers
  - Classifiers
  - Catalog settings

## AWS Glue Studio Info

### Create job Info



Author in a visual interface focused on data flow.  
**Visual ETL**



Author using an interactive notebook.  
**Notebook**

### Example jobs Info

### Your jobs (2) Info

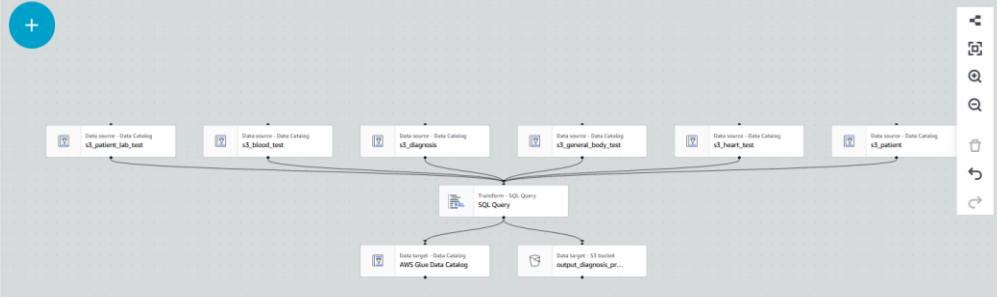
Filter jobs by property

<input type="checkbox"/>	Job name	Type	Created by
<input type="checkbox"/>	<b>diagnosis_populate</b>	Glue ETL	Visual
<input type="checkbox"/>	<a href="#">PopulateBloodTest</a>	Glue ETL	Visual

### diagnosis\_populate

Last modified on 11/27/2024, 5:48:47 PM **Actions** **Save** **Run**

**Visual** | Script | Job details | Runs | Data quality | Schedules | Version Control | Upgrade analysis - preview



## Amazon SageMaker - Notebook

- TensorBoard
- Profiler
- Notebooks**

▼ Admin configurations

- Domains
- Role manager
- Images
- Lifecycle configurations

SageMaker dashboard

Search

### Notebook instances Info

Search notebook instances

	Name	Instance	Creation time	Status	Actions
<input type="radio"/>	<b>diabetesdiagnosis</b>	<b>ml.t2.medium</b>	11/27/2024, 6:36:47 PM	InService	Open Jupyter   <b>Open JupyterLab</b>

## Amazon Sagemaker – DiabeticPrediction notebook

The screenshot shows the Amazon SageMaker JupyterLab interface. The top menu bar includes File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. The left sidebar shows a file explorer with a search bar and a list of files: DiabeticPrediction2.ipynb (7 hours ago), test.csv (7 hours ago), and train.csv (7 hours ago). The main area displays the DiabeticPrediction2.ipynb notebook. The notebook content includes two CSV datasets and a series of Python code cells. The first CSV dataset has columns: blood\_pressure, skin\_thickness, insulin, bmi, diabetes\_pedigree\_function. The second CSV dataset has columns: age, gender\_Male, smoking\_history\_Former, smoker. The code cells include comments and Python code for encoding the CSV string to bytes, making predictions with a specified content type, thresholding predictions to get binary outcomes, and evaluating the model. The final output shows a Model Accuracy of 100.00%.

```
[26]: # Encode the CSV string to bytes
csv_bytes = csv_buffer.getvalue().encode('utf-8')

# Make predictions with specified content type
predictions = predictor.predict(csv_bytes, initial_args={'ContentType': 'text/csv'})

# Threshold predictions to get binary outcomes
predictions = [float(pred) for pred in predictions.decode('utf-8').strip().split(',')
               (pd.Series(predictions) > 0.4).astype(int)]

# Evaluate the model
accuracy = accuracy_score(y_test, predictions)
print(f"Model Accuracy: {accuracy * 100:.2f}%")

Model Accuracy: 100.00%
```

## Amazon Sagemaker – Models Deployment

The screenshot shows the Amazon SageMaker console. The left sidebar contains navigation links for JumpStart, Governance, HyperPod Clusters, Ground Truth, Processing, Training, and Inference. The main area shows the Models page. The top bar includes the Amazon SageMaker logo and a search bar. The Models section includes a search bar and a table of models. The table has columns Name and ARN. A model named sagemaker-xgboost-2024-11-29-03-15-17-038 is highlighted.

Name	ARN
sagemaker-xgboost-2024-11-29-03-15-17-038	arn:aws:sagemaker:us-east-2:241533149150:model/sagemaker-xgboost-2024-11-29-03-15-17-038



## Amazon Sagemaker – Models Endpoints

- HyperPod Clusters
- Ground Truth
- Processing
- Training
- Inference**
- Compilation jobs
- Marketplace model packages
- Models
- Endpoint configurations
- Endpoints**
- Batch transform jobs

## Endpoints

	Name	ARN	Creation time	Status
<input type="radio"/>	sagemaker-xgboost-2024-11-29-03-15-17-038	arn:aws:sagemaker:us-east-2:241533149150:endpoint/sagemaker-xgboost-2024-11-29-03-15-17-038	11/29/2024, 8:45:18 AM	<span>✔ InService</span>

## AWS Lambda

The screenshot shows the AWS Lambda console interface. At the top, the breadcrumb navigation indicates the path: Lambda > Functions > lambda\_handler. Below this, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The 'Code source' tab is active, displaying a message: 'You are using the new console editor.' with a 'Switch to the old editor' button. On the right, there is an 'Upload from' button. The main area shows the code editor for 'lambda\_function.py'. The code defines a 'lambda\_handler' function that takes 'event' and 'context' as arguments. The function body is highlighted with a yellow circle. The code includes comments and a list of example input data, followed by a line that formats the input data into a string using 'ast.literal\_eval'.

AWS - API Gateway

API Gateway > APIs

API Gateway

APIs

Custom domain names Updated

Domain name access associations New

VPC links

Usage plans

API keys

Client certificates

APIs (1/1)

Find APIs

Name	Description	ID	Protocol	API endpoint type
diabetic-prediction-api		wqtvfbwi19	REST	Regional

AWS - API Gateway (Stages)

API Gateway > APIs > diabetic-prediction-api (wqtvfbwi19) > Stages

API Gateway

APIs

Custom domain names Updated

Domain name access associations New

VPC links

API: diabetic-prediction-api

Resources

Stages

Authorizers

Gateway responses

Models

Resource policy

Documentation

Dashboard

API settings

Stages

Dev

Stage details

Stage name

Dev

Cache cluster

Inactive

Default method-level caching

Inactive

Invoke URL

https://wqtvfbwi19.execute-api.us-east-2.amazonaws.com/Dev

Active deployment

siccou on November 28, 2024, 17:49 (UTC+05:30)

Rate

-

Burst

-

Web ACL

-

Client certificate

-

Logs and tracing

## AWS - API Gateway (Resources)

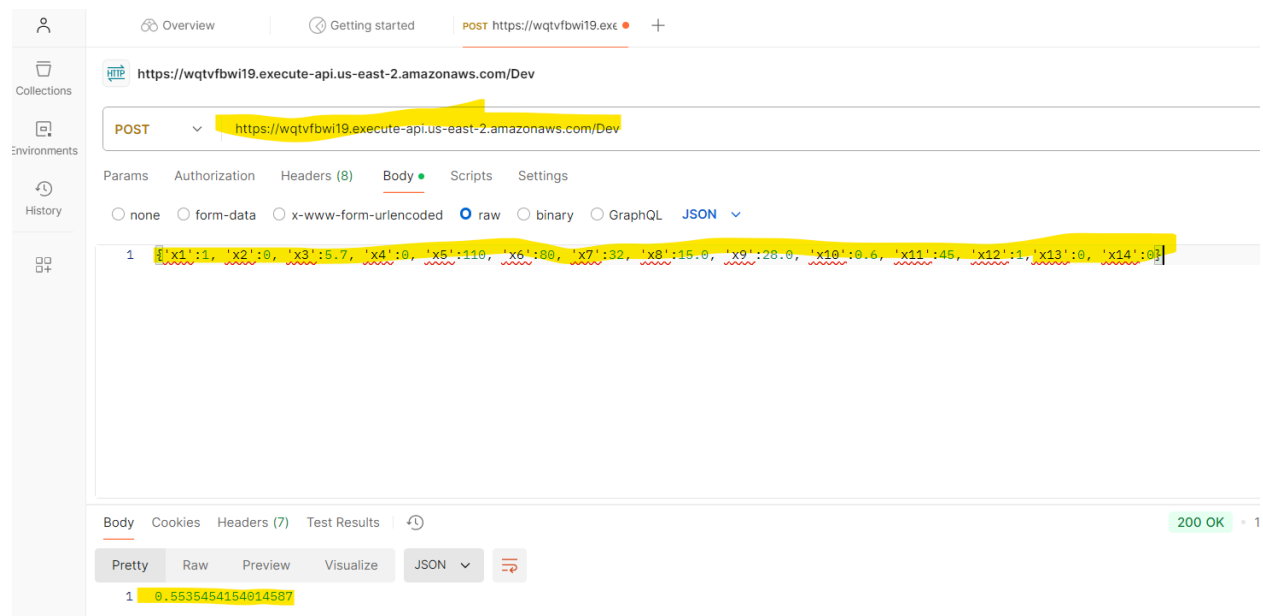
The screenshot shows the AWS API Gateway console interface. The breadcrumb navigation at the top reads: **API Gateway** > **APIs** > **Resources - diabetic-prediction-api (wqtvfbw19)**. On the left, the **API Gateway** sidebar is visible, with the **Resources** section highlighted under the **API: diabetic-prediction-api** heading. The main content area is titled **Resources** and features a **Create resource** button. Below this, a resource path **/** is shown with a **POST** method selected. To the right, the **Method execution** diagram is displayed, showing the flow from **Client** to **Method request**, then to **Integration request**, which connects to a **Lambda integration**. The response flow goes from **Integration response** (labeled **Proxy integration**) to **Method response** and finally back to the **Client**. Below the diagram, tabs for **Method request**, **Integration request**, **Integration response**, **Method response**, and **Test** are present. The **Method request settings** section shows **Authorization** as **NONE** and **API key required** as **False**. Buttons for **Update documentation** and **Delete** are also visible.

## AWS - API Gateway ( API Testing)

This screenshot shows the **Test** tab within the AWS API Gateway console for the same **diabetic-prediction-api**. The breadcrumb navigation remains the same. In the left sidebar, the **Resources** link is highlighted. The main area shows the **Test method** configuration. It includes fields for **Query strings** (containing `param1=value1&param2=value2`), **Headers** (with `Header1:value1` and `Header2:value2`), and a **Client certificate** dropdown set to **No client certificates have been generated.** The **Request body** is shown in a JSON format: `{ "x1": 1, "x2": 0, "x3": 5.7, "x4": 0, "x5": 100, "x6": 0.0, "x7": 3.0, "x8": 15.0, "x9": 20.0, "x10": 0, "x11": 0, "x12": 1, "x13": 0, "x14": 0 }`. At the bottom, the **Test** button is visible. Below the configuration, the **Test results** are displayed in a table:

Request	Latency ms	Status
<b>Response body</b> <code>0.5535454545454545</code>	2390	200

## Postman – API Testing



## Additional Features Integration

- **AWS Encryption:** Implement server-side encryption for Amazon S3 buckets to ensure data security at rest. Extend encryption practices to other AWS services, including RDS Aurora and Glue to maintain end-to-end data protection.
- **API Gateway Endpoint Security:** Enforce security measures such as API keys, IAM security against unauthorized access and ensure robust API performance.
- **Minimal Policy Access Rights:** Design IAM roles and policies adhering to the principle of least privilege, granting only the required permissions for each service to ensure enhanced security and compliance.
- **Git Integration:** Integrate with Git for version control, enabling efficient collaboration and tracking of code changes.

## Implementation Resources

Git Repository	<a href="https://github.com/kapsenitin1/ML-Capabilites_AWS.git">https://github.com/kapsenitin1/ML-Capabilites_AWS.git</a>
Github	<a href="https://github.com/kapsenitin1/ML-Capabilites_AWS">https://github.com/kapsenitin1/ML-Capabilites_AWS</a>