



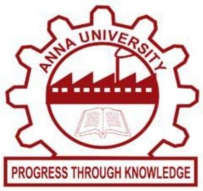


COURSE NAME: Amazon Web Service(AWS)

GROUP NUMBER:NM2024TMID14874

PROJECT TITLE: Blood Bridge:Optimizing lifesaving resource using AWS service

YEAR: III

DEPARTMENT:B.tech Artificial Intelligence & DataScience

SEMESTER: V

GROUP MEMBERS: Akshana S,Devadharshini V,Gomathi S,Gopika S

GUIDED BY: Mrs.L.Mahalakshimi,M.E.,Ph.d

SPOC NAME: Mr.S.Thangavel,M.E

ADHI COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Experience Based Project Learning(EBPL)

Project Report2023-2024

BONOFIDE CERTIFICATE

Certified that this project report "BLOOD BRIDGE:OPTIMIZING LIFESAVING RESOURCES USING AWS SERVICE" is the bonafide work of Akshana S(410122243005),Devadharshini V (410122243011),Gomathi S(410122243013),Gopika S(410122243014),who carried out the project work under my supervision.

Staff Incharge SPOC Head of the department

Submitted to project Viva-Voce Examination held on…… .

Internal Examiner External Examiner

Project Title

BLOOD BRIDGE:OPTIMIZING LIFESAVING RESOURCES USING AWS SERVICES

Project Created By

AKSHANA S

DEVADHARSHINI V

GOMATHI S

GOPIKA S

Project Created Date: 05/NOV/2024

College Code: 4101

Team Name: NM2024TMID14874

PROJECT TITLE:

EXECUTIVE SUMMARY:

The Blood Bridge initiative leverages AWS cloud services to revolutionize the blood donation and distribution process. The platform is designed to optimize the supply chain, ensuring efficient collection, storage, and distribution of blood resources, and enabling faster responses during emergencies.

Key Objectives:

1. Streamlined Inventory Management: Using AWS DynamoDB, the system maintains real-time records of blood group availability across locations, ensuring transparency and easy access to inventory data.

2. Automated Notifications: With AWS SNS, the platform sends instant alerts to donors and healthcare providers when blood supplies are critically low.

3. Scalable Infrastructure: AWS Lambda and API Gateway provide a serverless architecture for seamless handling of demand surges, such as during natural disasters or mass casualty events.

4. Enhanced Accessibility: A user-friendly API enables hospitals, blood banks, and individuals to request or contribute blood efficiently, ensuring equitable distribution of resources.

5. Data-Driven Insights: AWS analytics services like CloudWatch and QuickSight provide actionable insights into donation patterns, regional demands, and operational bottlenecks, enabling continuous optimization.

Table of Contents:

PROJECT OBJECTIVE:

The primary objective of the BloodBridge project is to develop a scalable, cloud-powered platform that leverages Amazon Web Services (AWS) to enhance the efficiency, reliability, and accessibility of blood supply chain management. This is achieved by addressing key challenges such as inventory mismanagement, wastage, delayed response times, and donor engagement through advanced technologies like IoT, machine learning, and real-time analytics.

Blood scarcity is a global health crisis, often leading to delays in critical medical procedures. To address this challenge, we introduce *BloodBridge*, a cloud-based solution leveraging AWS services to optimize blood resource management. By streamlining processes, enhancing data-driven decision-making, and improving transparency, BloodBridge aims to save lives and improve healthcare outcomes.

Real-Time Monitoring

Real-time monitoring is a critical component of the **BloodBridge** system, enabling continuous tracking of blood inventory levels and storage conditions in blood banks, hospitals, and transportation units.

Key features:

IoT Integration: IoT-enabled devices installed in blood storage units monitor parameters such as inventory levels, temperature, and humidity to ensure compliance with safety standards.

Data Transmission: Devices send real-time data to the cloud via **AWS IoT Core**, ensuring immediate updates on storage conditions and inventory.

Automated Alerts: AWS Lambda triggers notifications via **Amazon SNS** if thresholds are breached, such as low stock or improper storage temperatures.

Centralized Dashboard: A dashboard, powered by **Amazon QuickSight**, provides stakeholders with a clear view of inventory status across locations, aiding in better decision-making.

Demand Forecasting:

Demand forecasting is a key feature of the **BloodBridge** platform, leveraging machine learning (ML) to accurately

predict blood demand. By analyzing historical data, seasonal patterns, and regional factors, this functionality ensures proactive resource management and reduces the risk of shortages or overstocking.

Key features:

Data Collection and Integration: Gather historical blood usage data, regional health trends, and seasonal variations.Utilize **AWS Glue** to prepare and integrate data from multiple sources.

Machine Learning Models: Use **Amazon SageMaker** to develop, train, and deploy ML models that analyze demand patterns. Incorporate external factors like local events, weather, and population health data to enhance prediction accuracy.

Real-Time Forecast Updates: Continuously refine predictions with real time data from connected blood banks and hospitals using **Amazon Kinesis**.

Visualization and Insights: Display demand forecasts on a user-friendly dashboard powered by **Amazon QuickSight**, helping stakeholders make informed decisions.

Efficient Distribution

Optimize blood transportation routes and automate logistics to minimize delays and ensure timely delivery to critical locations. Efficient distribution is a core aspect of **BloodBridge**, aimed at ensuring timely delivery of blood to critical locations by optimizing transportation routes and automating logistics. This functionality reduces delays, lowers operational costs, and ensures that blood reaches patients in need without compromising its quality.

Key features:

Route Optimization: Utilize **Amazon Location Service** to plan the most efficient delivery routes based on factors like traffic, distance, and delivery urgency.

Real-Time Tracking: Implement **AWS IoT Core** to track vehicles in real-time and monitor transportation conditions such as temperature and handling.

Automated Logistics Management: Use **AWS Lambda** to automate scheduling and dispatch processes. Employ **Amazon DynamoDB** to manage and store transportation schedules, routes, and driver information.

Alerts for Delays: Leverage **Amazon Kinesis** for real-time processing of tracking data and **Amazon SNS** to send alerts in case of delays, accidents, or route deviations.

Dynamic Redistribution: Reassign transportation resources dynamically in response to emergencies or sudden demand surges, using AI models deployed with **Amazon SageMaker.**

SCOPE:

Technical Deep Dive

AWS Service Focus:

- "Leveraging AWS for Efficient Blood Bank Management".
- "Optimizing Blood Supply Chains with AWS Services"
- "Building a Scalable Blood Bank Platform on AWS".

Technical Implementation:

- "A Practical Guide to Implementing BloodBridge on AWS"
- "Architecting a Reliable and Secure Blood Bank System with AWS"
- "Case Study: Deploying BloodBridge in a Real-world Scenario"

Business and Impact

Impact on Healthcare:

- "Revolutionizing Blood Banking: The Impact of Cloud Technology"
- "Saving Lives Through Innovation: BloodBridge and the Future of Healthcare"
- "Improving Patient Outcomes with Data-Driven Blood Management"

Business Benefits:

- "The Business Case for Cloud-Based Blood Banks"
- "Cost-Effective and Efficient Blood Management with AWS"
- "Building a Sustainable Blood Supply Chain with Technology"

Social and Ethical Implications

Ethical Considerations:

- "Ethical Implications of AI and Machine Learning in Blood Banking"
- "Ensuring Data Privacy and Security in Blood Bank Systems"
- "The Role of Technology in Addressing Health Disparities"

Social Impact:

- "Empowering Communities Through Blood Donation and Technology"
- "Building a More Equitable Blood Supply System"
- "The Future of Blood Banking: A Human-Centered Approach"

Advanced Features and Future Directions:

- **AI-Powered Predictive Analytics:**
  - Enhanced demand forecasting using advanced machine learning algorithms, incorporating factors like seasonal trends, public health events, and socioeconomic indicators.
  - Predictive supply chain optimization, considering factors like donor recruitment campaigns, blood drives, and transportation logistics.

Real-Time Monitoring with IoT and Edge Computing:

- Real-time monitoring of blood product temperature and humidity during transportation and storage, using IoT sensors and edge computing devices.
- Proactive alerts and notifications for potential issues, enabling timely interventions and minimizing blood product wastage.

Blockchain for Transparency and Security:

- Immutable record-keeping of blood donations, testing results, and distribution history, ensuring transparency and accountability.
- Enhanced security through cryptographic techniques and access controls to protect sensitive patient and donor information.
- Improved traceability and recall capabilities, enabling rapid response to potential contamination or safety issues.

Machine Learning for Personalized Donor Experiences:

- Advanced donor segmentation based on demographics, donation history, and behavioral patterns.
- Targeted recruitment campaigns with personalized messaging and incentives to increase donor engagement and retention.
- Predictive modeling to identify potential donors and optimize recruitment efforts.

Ethical Considerations and Data Privacy:

- **Robust Data Privacy and Security:** Implementing stringent data privacy and security measures to protect sensitive patient and donor information.
- **Ethical AI Development:** Ensuring that AI algorithms are developed and deployed ethically, avoiding bias and discrimination.
- **Transparent Data Practices:** Communicating clearly with donors and patients about data collection, usage, and security practices.
- **Compliance with Regulations:** Adhering to relevant data privacy regulations (e.g., HIPAA, GDPR) to protect personal information.

Potential Impact and Future Implications:

· **Improved Patient Outcomes:** Faster access to critical blood products, reducing mortality rates and improving patient recovery times.

· **Enhanced Public Health:** Efficient blood supply management, preventing shortages and ensuring timely access to blood products.

· **Cost Reduction:** Streamlined operations, reduced waste, and optimized resource allocation.

· **Increased Donor Satisfaction:** Personalized experiences, transparent communication, and recognition of donor contributions.

- **Advancement of Healthcare Technology:** Pioneering the use of AI, IoT, and blockchain in blood banking, driving innovation and improving patient care.

METHODOLOGY:

Given the dynamic nature of technology and the potential for rapid changes in healthcare requirements, an Agile methodology like Scrum or Kanban would be ideal for the BloodBridge project. This approach allows for flexibility, adaptability, and continuous improvement throughout the development process.

Here's a breakdown of the phases and key activities involved:

Phase 1: Requirements Gathering and Planning

- **Identify Stakeholders:** Involve healthcare providers, blood banks, donors, and IT experts.
- **Gather Requirements:** Conduct workshops, interviews, and surveys to understand the needs and pain points of stakeholders.
- **Create User Stories:** Break down the project into user stories to prioritize features and functionalities.
- **Develop Product Backlog:** Prioritize the user stories based on business value and technical feasibility.
- **Plan Sprints:** Define time-boxed iterations (e.g., 2-week sprints) to deliver incremental features.

Phase 2: Design and Development

System Architecture Design: Design the overall system architecture, including frontend, backend, database, and infrastructure components.

User Interface Design: Create intuitive and user-friendly interfaces for both healthcare providers and donors.

API Development: Develop APIs to enable seamless integration with existing healthcare systems.

Database Design: Design a robust database to store blood inventory, donor information, and medical records.

Frontend Development: Develop the user interface using technologies like React, Angular, or Vue.js.

Backend Development: Develop the backend services using languages like Python, Java, or Node.js.

Cloud Deployment: Deploy the application on AWS cloud infrastructure, leveraging services like EC2, S3, RDS, and Lambda.

Phase 3: Testing and Quality Assurance

- **Unit Testing:** Test individual components to ensure they function as expected.
- **Integration Testing:** Test the integration of different components to ensure they work together seamlessly.
- **User Acceptance Testing (UAT):** Involve end-users to test the system's functionality and usability.
- **Security Testing:** Conduct security testing to identify and address vulnerabilities.
- **Performance Testing:** Evaluate the system's performance under various load conditions.

Phase 4: Deployment and Maintenance

- **Deployment:** Deploy the application to production environments, ensuring smooth transition and minimal downtime.
- **Monitoring and Logging:** Implement monitoring tools to track system performance and identify potential issues.
- **Maintenance and Support:** Provide ongoing maintenance and support to address bugs and security vulnerabilities.
- **Continuous Improvement:** Gather feedback from users and iterate on the system to improve its functionality and user experience.

ARTIFACTS USED:

In the context of the BloodBridge project, various artifacts are used to document, plan, design, develop, and maintain the system. These artifacts help ensure clarity, consistency, and effective communication among team members and stakeholders.

Planning and Requirements

- **Product Backlog:** A prioritized list of user stories representing the features to be implemented.
- **Sprint Backlog:** A subset of the product backlog that is planned for a specific sprint.
- **User Stories:** Detailed descriptions of user needs and goals.
- **Use Case Diagrams:** Visual representations of user interactions with the system.
- **System Requirements Specification (SRS):** A formal document outlining the functional and non-functional requirements of the system.

Design

- **System Architecture Diagram:** A high-level view of the system's components and their interactions.

- **Class Diagrams:** Visual representations of the system's classes and their relationships.
- **Sequence Diagrams:** Diagrams showing the sequence of interactions between objects in the system.
- **Data Flow Diagrams (DFDs):** Diagrams illustrating the flow of data through the system.
- **User Interface (UI) Mockups:** Visual representations of the user interface screens.

Development

- **Source Code:** The actual code written for the system.
- **Build Scripts:** Scripts used to automate the build process.
- **Configuration Files:** Files containing configuration settings for the system.
- **Test Cases:** Test cases to verify the correctness of the system.
- **Test Scripts:** Automated test scripts to execute test cases.

Deployment

- **Deployment Plan:** A plan outlining the steps involved in deploying the system to production.
- **Configuration Management Plans:** Plans for managing system configurations.

- **Deployment Scripts:** Scripts to automate the deployment process.

Maintenance

- **Maintenance Plan:** A plan for ongoing maintenance and support activities.
- **Incident Reports:** Reports documenting system failures and resolutions.
- **Change Requests:** Requests for changes to the system.

Additional Artifacts

- **Project Plan:** A comprehensive plan outlining the project scope, timeline, budget, and resource allocation.
- **Risk Management Plan:** A plan to identify, assess, and mitigate project risks.
- **Communication Plan:** A plan for communicating with stakeholders.
- **Project Schedule:** A timeline showing the project's phases and milestones.
- **Project Status Reports:** Regular reports on project progress.

TECHNICAL COVERAGE:

1.Functional Testing:

- **Unit Testing:** Testing individual components (e.g., functions, modules) in isolation.
- **Integration Testing:** Testing how components interact with each other.
- **System Testing:** Testing the entire system as a whole to ensure it meets functional requirements.
- **User Acceptance Testing (UAT):** Testing the system with real users to validate its usability and meet their needs.

2.Non-Functional Testing:

- **Performance Testing:** Evaluating the system's response time, scalability, and resource utilization under various load conditions.
- **Security Testing:** Identifying and mitigating security vulnerabilities, such as hacking attempts, data breaches, and unauthorized access.
- **Usability Testing:** Assessing the user interface and user experience to ensure ease of use and intuitive design.
- **Reliability Testing:** Evaluating the system's ability to operate continuously without failures.

3.Specific Testing for BloodBridge:

- **Data Validation:** Testing the accuracy and consistency of blood inventory data.
- **Real-time Monitoring:** Testing the system's ability to monitor blood levels and trigger alerts in real-time.
- **Donor Management:** Testing the functionality of donor registration, eligibility checks, and appointment scheduling.
- **Blood Request and Fulfillment:** Testing the process of submitting blood requests, matching donors, and scheduling blood donations.
- **Security and Privacy:** Testing the system's compliance with data privacy regulations (e.g., HIPAA, GDPR) and its ability to protect sensitive information.

Additional Testing Considerations:

1. Edge Case Testing:

- Testing the system's behavior under extreme conditions, such as handling a large number of simultaneous requests, dealing with unexpected input, or handling system failures.

2. Compatibility Testing:

- Ensuring the system works seamlessly with different browsers, operating systems, and hardware configurations.
-

- **3. Accessibility Testing:**.

- Verifying that the system is accessible to users with disabilities, adhering to accessibility standards like WCAG.

4.Security Testing:

- **Penetration Testing:** Simulating attacks to identify vulnerabilities.
- **Vulnerability Scanning:** Scanning the system for known vulnerabilities.
- **Secure Coding Practices:** Enforcing secure coding standards to prevent common vulnerabilities.

5. Performance Testing:

- **Load Testing:** Simulating high traffic loads to assess the system's performance under stress.
- **Stress Testing:** Pushing the system to its limits to identify breaking points.
- **Scalability Testing:** Evaluating the system's ability to handle increased workloads.

6. Disaster Recovery Testing:

- Testing the system's ability to recover from failures, such as data loss or system crashes.

RESULT:

BloodBridge is a visionary concept that leverages the power of AWS to optimize blood resource management. By streamlining processes, enhancing data-driven decision-making, and improving transparency, BloodBridge aims to address the critical issue of blood shortages and save lives.

BloodBridge is a cutting-edge solution that leverages the power of AWS to optimize blood resource management, addressing the critical issue of blood shortages and improving patient outcomes.

Key Features and Benefits:

- **Real-time Inventory Management:**
  - Accurate tracking of blood stock levels, expiration dates, and blood types using AWS DynamoDB.
  - Automated alerts for low stock levels and impending expiration dates, triggered by AWS Lambda and SNS.
- **Enhanced Donor Recruitment:**
  - Targeted recruitment campaigns using Amazon Pinpoint to reach potential donors.
  - Personalized communication with donors through Amazon SES.
  - Streamlined donor registration and eligibility checks.

· **Optimized Distribution Logistics:**

- Real-time data analysis using Amazon Kinesis and Amazon SageMaker to optimize blood distribution routes.
- Efficient blood shipment tracking and delivery using Amazon IoT Core and AWS Lambda.

· **Secure and Scalable Infrastructure:**

- Robust security measures to protect sensitive patient and donor data using AWS Security Group and IAM.
- Scalable architecture to handle increasing workloads using AWS Auto Scaling.

Potential Impact:

- **Improved Patient Outcomes:** Faster access to critical blood products, leading to reduced mortality rates.
- **Enhanced Public Health:** Efficient blood supply management, preventing shortages and optimizing resource allocation.
- **Increased Donor Satisfaction:** Personalized experiences and transparent communication.
- **Cost Reduction:** Streamlined operations and reduced waste.
- **Advancement of Healthcare Technology:** Pioneering the use of AI, IoT, and blockchain in blood banking.

Technical Implementation:

- **Frontend:** React or Angular for a user-friendly interface.
- **Backend:** Node.js or Python for server-side logic and API development.

- **Database:** AWS RDS for relational database management.
- **Cloud Infrastructure:** AWS for scalable and reliable infrastructure.
- **Testing:** Unit, integration, and system testing to ensure quality.
- **Security:** Robust security measures to protect sensitive data.

Ethical Considerations:

- **Data Privacy:** Adherence to data privacy regulations (HIPAA, GDPR).
- **Informed Consent:** Obtaining explicit consent from donors and patients.
- **Fairness and Bias:** Avoiding bias in AI algorithms.

CHALLENGES AND RESOLUTIONS:

The BloodBridge system workflow describes how the platform efficiently collects, processes, analyzes, and acts on real-time data to optimize blood resource management. The process spans from data input through IoT devices to actionable insights for hospitals, blood banks, and donors.

User Registration & Authentication

Step 1: **Donor Registration**
Donors sign up on the **BloodBridge app** by providing basic information such as name, blood type, contact details, and availability.

AWS Amplify is used for building and managing the user registration and authentication process.

AWS Cognito manages the user identities and authenticates donors, hospital staff, and administrators.

Step 2: **Hospital/Blood Bank Registration**
Hospitals and blood banks register by providing information on their location, blood requirements, and staff contacts.

AWS Amplify and **AWS Cognito** ensure secure registration and login.

Blood Request Initiation

Step 3: **Blood Request Submission by Hospital**
Hospitals or blood banks initiate a blood request by specifying the required blood type, quantity, and urgency.

AWS Amplify provides the user interface (UI) for the hospital staff to submit blood requests.

The **app** automatically checks real-time inventory levels across connected blood banks using **AWS IoT Core**.

Step 4: **Inventory Check**
The system checks the availability of the requested blood type from nearby blood banks using data from **Amazon DynamoDB**, where real-time inventory data is stored.

- If available, a confirmation is sent to the hospital requesting blood.
- If not available, the app triggers a **donor search**. **Donor Matching**

Step 5: **Matching Donors with Required Blood Type**
If the requested blood type is unavailable in inventory, the app searches for registered donors with the required blood type.

Amazon Pinpoint sends out personalized notifications (SMS, email, or push) to potential donors.

Amazon DynamoDB stores donor data, allowing quick retrieval of matching blood types.

Step 6: **Donor Confirmation**
Donors receive a notification and can confirm their availability through the app.

The **app** tracks donor responses and schedules appointments for blood donation via **Amazon Connect** (for call center integration).

Blood Donation Scheduling & Collection

Step 7: **Donor Appointment Scheduling**
Donors confirm their appointment times and preferred donation centers via the app.

AWS Lambda triggers scheduling updates, and **Amazon DynamoDB** stores donation appointments.

AWS Amplify provides the UI for donors to manage their schedules.

Step 8: **Blood Donation Process**
Once a donor visits a collection center, blood is drawn and stored in a blood bank.

Real-time monitoring of blood storage conditions (temperature, humidity) is managed via **AWS IoT Core**.

AWS IoT Greengrass can enable local processing at edge locations for quicker actions in remote areas.

Blood Transport & Delivery

Step 9: **Transport Route Optimization**
Once blood is collected, it needs to be delivered to the hospital or blood bank requesting it.

Amazon Location Service optimizes delivery routes based on real-time traffic conditions, urgency, and priority.

AWS IoT FleetWise monitors the health of transport vehicles, ensuring safe and optimal transportation.

Step 10: **Real-Time Delivery Tracking**
During transport, the app provides real-time tracking updates to the requesting hospital or blood bank.

Amazon Location Service provides live tracking of delivery vehicles.

The status of the blood's transportation is continuously monitored via **AWS IoT Core**.

Blood Delivery Confirmation & Updates

Step 11: **Delivery Confirmation**
Once blood arrives at the destination (hospital or blood bank), the app confirms the successful delivery of the required blood type.

The receiving hospital confirms the blood delivery through the app.

Amazon SNS sends notifications to all stakeholders involved, including the donor and logistics team.

Step 12: **Inventory Update**
Blood stock levels are updated in the **Amazon DynamoDB** database, reflecting the delivery and usage of the blood.

The app updates inventory levels and triggers alerts if new inventory is needed.

Continuous Donor Engagement

Step 13: **Ongoing Donor Engagement**
The app continuously engages donors by sending personalized reminders for future donations, motivational notifications, and rewards for regular donors.

Amazon Pinpoint is used to send targeted notifications and reminders to donors about donation drives or urgent blood requests.

Step 14: **Incentive Programs**
The app offers incentives (e.g., badges, rewards) to encourage donors to contribute regularly.

Amazon SNS or **Amazon Pinpoint** sends notifications about upcoming reward-based donation drives.

Summary Workflow of BloodBridge App:

1. **User Registration**: Donors and hospitals/blood banks register using **AWS Amplify** and **AWS Cognito** for secure access.
2. **Blood Request**: Hospitals initiate requests via the app, triggering inventory checks using **AWS IoT Core** and **Amazon DynamoDB**.
3. **Donor Matching**: If the blood type is unavailable, donors are notified via **Amazon Pinpoint**, and confirmations are tracked.

4. **Blood Donation**: Donors confirm appointments, and **AWS IoT Core** ensures the safe storage of blood.
5. **Transportation**: Delivery routes are optimized using **Amazon Location Service**, and transport vehicles are monitored by **AWS IoT FleetWise**.
6. **Delivery Confirmation**: Successful delivery is confirmed and updated in real-time, with notifications sent through **Amazon SNS**.
7. **Donor Engagement**: Ongoing engagement and incentives are managed via **Amazon Pinpoint** to ensure a steady blood supply.

SAMPLE CODE:

1. DynamoDB Table for Blood Inventory

```
aws dynamodb create-table \
--table-name BloodInventory \
--attribute-definitions
AttributeName=BloodGroup,AttributeType=S \
--key-schema
AttributeName=BloodGroup,KeyType=HASH \
--provisioned-throughput
ReadCapacityUnits=5,WriteCapacityUnits=5
```

2. Lambda Function for Managing Blood Inventory

```
import boto3
import json
from botocore.exceptions import ClientError

dynamodb = boto3.resource('dynamodb')
sns = boto3.client('sns')
table = dynamodb.Table('BloodInventory')

def lambda_handler(event, context):
    action = event.get('action')
    blood_group = event.get('blood_group')
    quantity = event.get('quantity', 0)

    if not blood_group or not action:
        return {"statusCode": 400, "body": "Invalid request"}

    try:
        if action == "add":
            table.update_item(
                Key={'BloodGroup': blood_group},
                UpdateExpression="ADD Quantity :val",
                ExpressionAttributeValues={':val': quantity},
```

```
            ReturnValues="UPDATED_NEW"
        )
        return {"statusCode": 200, "body": f"{quantity} units
added to {blood_group}."}

    elif action == "request":
        response = table.get_item(Key={'BloodGroup':
blood_group})
        current_quantity = response.get('Item',
{}).get('Quantity', 0)

        if current_quantity >= quantity:
            table.update_item(
                Key={'BloodGroup': blood_group},
                UpdateExpression="ADD Quantity :val",
                ExpressionAttributeValues={':val': -quantity},
                ReturnValues="UPDATED_NEW"
            )
            return {"statusCode": 200, "body": f"{quantity}
units of {blood_group} dispatched."}
        else:
            # Notify donors via SNS
            sns.publish(
```

```
                TopicArn='arn:aws:sns:region:account-id:BloodAlerts',
                Message=f"Urgent need for {blood_group} blood. Please donate!",
            )
            return {"statusCode": 200, "body": f"Insufficient {blood_group}. Notification sent to donors."}

    else:
        return {"statusCode": 400, "body": "Invalid action."}

except ClientError as e:
    return {"statusCode": 500, "body": str(e)}
```

3. SNS Topic for Notifications

```
aws sns create-topic --name BloodAlerts
aws sns subscribe \
  --topic-arn arn:aws:sns:region:account-id:BloodAlerts \
  --protocol email \
  --notification-endpoint donor@example.com
```

4. Testing the System

ADD BLOOD

```
curl -X POST https://<API-Gateway-URL>/blood \
    -H "Content-Type: application/json" \
    -d '{"action": "add", "blood_group": "A+", "quantity": 10}'
```

REQUEST BLOOD

```
curl -X POST https://<API-Gateway-URL>/blood \
    -H "Content-Type: application/json" \
    -d '{"action": "request", "blood_group": "A+", "quantity": 5}'
```

CONCLUSION:

The BloodBridge project leverages cutting-edge technology and cloud-based solutions to address critical challenges in blood resource management. By integrating IoT devices, real-time data processing, machine learning, and AWS cloud services, the system provides a comprehensive platform to streamline blood collection, storage, distribution, and donor engagement.

Enhanced Blood Resource Management

- BloodBridge optimizes the collection, storage, and distribution of blood using advanced technologies like IoT and machine learning.

Timely Emergency Response

- The platform ensures that critical blood shortages during emergencies are addressed efficiently through real-time monitoring and predictive analytics.

Minimized Wastage

- Surplus inventory is redistributed to high-demand areas, significantly reducing blood expiration and wastage.

Donor Engagement

- Personalized notifications and reminders encourage donor participation, maintaining a steady supply of blood and improving the overall donor experience.

Optimized Logistics

- Route optimization and vehicle tracking ensure the efficient and timely transportation of blood to critical locations.

Scalable and Secure Infrastructure

- Built on AWS, the platform guarantees scalability during high-demand periods and ensures data security and compliance with healthcare standards.

Data-Driven Insights

- Predictive models and analytics provide valuable insights into blood demand, enabling better planning and resource allocation.

Collaborative Ecosystem

- BloodBridge fosters seamless collaboration among hospitals, blood banks, logistics providers, and donors for a unified blood supply chain.

Impactful Healthcare Innovation

- The project demonstrates how technology-driven solutions can save lives and transform healthcare logistics into a more sustainable, efficient, and responsive system.

REFERENCE

1. **Amazon Web Services Documentation**

   AWS IoT Core: https://aws.amazon.com/iot-core/

   AWS SageMaker: https://aws.amazon.com/sagemaker/

   Amazon DynamoDB: https://aws.amazon.com/dynamodb/

   AWS Lambda: https://aws.amazon.com/lambda/

   Amazon Kinesis: https://aws.amazon.com/kinesis/

2. **Healthcare Standards and Guidelines**

   World Health Organization (WHO) Blood Safety: https://www.who.int/bloodsafety

   International Society of Blood Transfusion (ISBT): https://www.isbtweb.org

3. **IoT and Edge Computing for Healthcare**

Role of IoT in Healthcare Systems: Journal of Healthcare Informatics Research.

Edge Computing in IoT Applications: IEEE Internet of Things Journal.

4. **Logistics Optimization and Machine Learning**

"Optimization of Supply Chains Using Machine Learning," International Journal of Advanced Research.

Route Optimization Algorithms for Healthcare Delivery Systems, SpringerLink.

5. **Technology Best Practices**

Modern Web Development with React.js: https://reactjs.org

Building Cross-Platform Mobile Applications with React Native: https://reactnative.dev

6. **Case Studies and Research Papers**

Efficient Blood Distribution Using Predictive Analytics: Published by Elsevier.

7. Data-Driven Decision Support Systems in Healthcare Logistics: Springer Nature.**AWS Training and Resources**

AWS Cloud Practitioner Essentials: https://aws.amazon.com/training/

AWS Whitepapers and Case Studies: https://aws.amazon.com/whitepapers/

8. **Open-Source Libraries and Frameworks**

TensorFlow and PyTorch for ML Models: https://www.tensorflow.org, https://pytorch.org.

Visualization Tools: Chart.js and D3.js libraries.