For your **Security and Resilience** page, it’s essential to address both the **security measures** for protecting data and resources and the **resilience** of the architecture to ensure high availability, fault tolerance, and disaster recovery. Below is a suggested structure to help you articulate these concepts in your document.

**Security and Resilience for Document Migration Service**

**1. Security Overview**

**Objective**: To ensure the confidentiality, integrity, and availability of documents and metadata throughout the migration process from Oracle i-Prompt to AWS.

**Key Security Principles:**

* **Confidentiality**: Protecting sensitive data (documents and metadata) during transit and at rest.
* **Integrity**: Ensuring data remains consistent and unaltered throughout the migration process.
* **Availability**: Ensuring the service is highly available, with minimal downtime and fast recovery from failures.

**2. Security Measures in the Architecture**

**a. Data Encryption:**

* **Encryption at Rest**: All data stored in **S3** (document binaries) and **DynamoDB** (metadata) is encrypted using **AWS KMS (Key Management Service)** to ensure that sensitive data remains protected when stored in AWS.
  + **S3**: Server-side encryption (SSE) is enabled for all documents stored in **S3**.
  + **DynamoDB**: Data in **DynamoDB** is encrypted by default, ensuring that metadata is protected.
* **Encryption in Transit**: All communication between on-premises systems (**API-C**), **API Gateway**, and AWS services (such as **Lambda**, **S3**, **DynamoDB**) is encrypted using **TLS (Transport Layer Security)**. This ensures that data in transit is secure from potential interception.

**b. Identity and Access Management (IAM):**

* **Role-Based Access Control (RBAC)**: AWS **IAM roles** are used to grant permissions to different AWS resources, ensuring that only authorized users and services can access sensitive data.
  + **Lambda** functions have specific IAM roles that grant access to **S3** and **DynamoDB** for document processing.
  + **API Gateway** is secured using **IAM** policies to ensure that only authenticated and authorized users or services can access the API endpoints.
  + **Cross-Account Access**: In case other AWS accounts need access to your resources, **IAM roles** can be defined with appropriate permissions to enable cross-account access in a secure way.

**c. Authentication and Authorization:**

* **API Gateway Security**: Secure the endpoints of **API Gateway** with **API keys** or **AWS IAM Authentication**. This ensures that only authorized entities (like the on-premises **API-C**) can interact with your APIs.
* **Lambda Security**: Ensure that **Lambda** functions execute with the **least privilege** principle in mind, meaning they only have access to the resources they need to perform their tasks.

**d. Audit and Monitoring:**

* **AWS CloudTrail**: All API calls made to AWS services (including **Lambda**, **S3**, **DynamoDB**) are logged by **CloudTrail**, which provides an audit trail for all actions. This helps track any unauthorized or suspicious activity in the environment.
* **CloudWatch Logs**: Use **AWS CloudWatch** to monitor and log the execution of **Lambda** functions, **API Gateway** requests, and other AWS service activities. Alerts can be configured to notify the team of potential issues or breaches.

**e. Secure Data in Oracle i-Prompt:**

* **Database Access Control**: Ensure that **Oracle i-Prompt** is secured by enforcing strong authentication methods for database access.
* **Encryption**: If sensitive metadata is being stored in Oracle, ensure **encryption** is applied to data at rest within Oracle i-Prompt, mirroring the security posture in AWS.

**3. Resilience and High Availability**

**Objective**: To ensure that the document migration service is fault-tolerant, highly available, and capable of handling failure scenarios with minimal impact.

**a. High Availability and Fault Tolerance:**

* **API Gateway**: **API Gateway** automatically scales based on the incoming traffic. It’s designed to handle large numbers of requests with built-in redundancy to ensure availability even under high load.
* **Lambda**: AWS Lambda functions are stateless and can scale horizontally, ensuring that they can handle varying amounts of load. Lambda is deployed across multiple availability zones to provide resilience in case of failure in one zone.
* **S3**: **Amazon S3** is inherently highly available and durable, with objects stored across multiple **availability zones (AZs)**. This ensures that documents are always accessible even if one AZ experiences failure.
* **DynamoDB**: DynamoDB offers **multi-region replication** and automatic failover, ensuring that metadata (GUIDs, S3 URLs) is always available, even in the event of a region failure.

**b. Disaster Recovery (DR):**

* **Cross-Region Replication**: For critical documents and metadata, consider setting up **cross-region replication** for **S3** and **DynamoDB**. This ensures that if one AWS region becomes unavailable, another region can take over and continue operations with minimal data loss.
* **Lambda Resilience**: Ensure that **Lambda functions** are retried in case of transient failures. You can configure retries and dead-letter queues (DLQs) to capture and handle failed invocations, ensuring that no data is lost.
* **Backups and Snapshots**:
  + Regular **backups** should be taken for **DynamoDB** tables and **S3** buckets. **S3 versioning** should also be enabled to prevent data loss and facilitate recovery of older versions of documents.
  + Consider using **AWS Backup** or other automated backup solutions to ensure that all critical data is backed up periodically.

**c. Auto-scaling and Load Balancing:**

* **Lambda** automatically scales based on the number of incoming requests, ensuring that it can handle high volumes of document processing requests without manual intervention.
* **API Gateway** uses **AWS’s global edge network**, which automatically balances traffic to available API endpoints.

**4. Incident Response and Recovery Plan**

* **Incident Detection**: Leverage **AWS CloudWatch** to monitor application performance and automatically alert the team in case of issues (e.g., high error rates in Lambda, failed document uploads, etc.).
* **Incident Handling**: Define clear incident response procedures, including triggering **Lambda** for retries, rolling back changes in **DynamoDB** and **S3**, and informing stakeholders of the issue.
* **Recovery Procedures**:
  + For **S3**: Documents can be recovered by restoring them from cross-region replication or from backups.
  + For **DynamoDB**: Use **point-in-time recovery** (PITR) to restore data to a specific timestamp.
  + **Oracle i-Prompt**: Ensure database backups are taken regularly and stored securely.

**5. Security and Resilience Best Practices**

* **Least Privilege**: Always apply the **least privilege** principle for IAM roles to minimize unnecessary access to AWS resources.
* **Data Minimization**: Only store the necessary metadata in **DynamoDB** and **Oracle i-Prompt**. Limit the amount of data stored in each to ensure easier management and better security.
* **Compliance**: Ensure that your architecture complies with industry-specific regulations (e.g., GDPR, HIPAA) by using AWS services that meet these standards (e.g., **AWS Shield**, **AWS WAF**, etc.).

**Conclusion**

The **security** and **resilience** of your **document migration service** are paramount to ensure the confidentiality, integrity, and availability of your data. By leveraging AWS services like **IAM**, **S3**, **Lambda**, **DynamoDB**, and **CloudWatch**, along with proper disaster recovery and monitoring strategies, you can ensure that your solution remains secure, highly available, and fault-tolerant throughout its operation.