Phase 2

Project Title: *Data Backup and Recovery System Using IBM Cloud Object Storage*

1. **Architecture Overview:**
   a. **Data Storage with IBM Cloud Object Storage:** IBM Cloud Object Storage provides elastic scalability to handle growing backup demands. Configurable storage classes (Standard, Vault, or Cold Vault) optimize cost and performance based on backup frequency and recovery time objectives (RTO). Low-latency access ensures efficient data retrieval during recovery operations.

   b. **Database Integration with MongoDB:** MongoDB serves as the database system to manage and store the company's critical data. Tools like mongodump and mongorestore are used to create backups and restore the data. MongoDB's replication ensures high availability and minimizes downtime during recovery.

   c. **Automation and Event-Driven Recovery:** IBM Cloud Functions: A serverless architecture is employed to automate: o Regular backups of MongoDB data. o Recovery processes triggered by predefined failure conditions. Event Triggers: Automatic initiation of workflows based on scheduled tasks or system alerts.

   d. **Security:**
      - **Encryption:**
        - Data in transit is encrypted using SSL/TLS.
        - Data at rest in IBM Cloud Object Storage is secured with IBM Key Protect.
      - **Access Control:**
        - Role-based access control (RBAC) limits user permissions.
        - Integration with IBM Identity and Access Management (IAM) ensures secure operations.
      - **Audit Logs:** Continuous monitoring and logging of activities for compliance and forensic analysis.

2. **Service Integration:**
   a. **Integration with IBM Cloud Functions:**
      - IBM Cloud Functions automates critical workflows, such as scheduling MongoDB backups and triggering recovery processes.
      - Event-driven mechanisms ensure that backups are performed at predefined intervals, and recovery is initiated automatically during failure scenarios.
      -

   b. **IBM Cloud Object Storage SDKs and APIs:**
      - APIs and SDKs provided by IBM enable seamless interaction with Object Storage, allowing efficient upload, retrieval, and management of backup data.

- These tools streamline the integration of Object Storage with MongoDB, ensuring smooth data flow and minimal latency.

c. IBM Identity and Access Management (IAM):

- IAM manages user authentication and permissions across all integrated services, enhancing security during backup and recovery operations.
- Provides centralized control over service access and reduces the risk of unauthorized data handling.

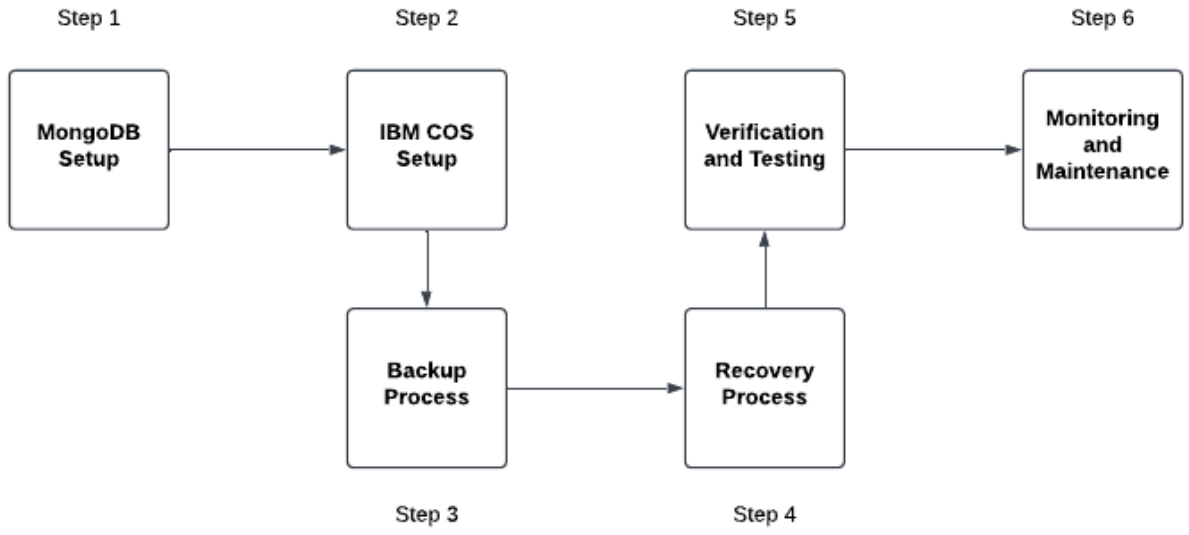
d. Monitoring and Alert Integration:

- IBM Cloud Monitoring integrates with Object Storage and Cloud Functions to track backup performance and detect failures in real time.
- Configurable alerts notify the team of anomalies, ensuring timely resolution of issues.

e. Optional Integration with IBM Watson AI:

- For advanced use cases, IBM Watson AI can be incorporated to analyze backup trends, predict potential system risks, and recommend preventive measures.
- This integration adds a layer of intelligence, making the system proactive rather than reactive.

3. Flow Diagram:



4. **Implementation Plan:**
   a. **MongoDB Setup**
      - Step 1.1: Install MongoDB on your system.
        Step 1.2: Configure MongoDB database and collections.
        Step 1.3: Verify MongoDB installation and database connection.

   b. **IBM Cloud Object Storage (COS) Setup**
      - Step 2.1: Create an IBM Cloud account.
        Step 2.2: Set up an IBM COS bucket.
        Step 2.3: Configure access credentials (API Key, Service ID).
        Step 2.4: Install and configure IBM Cloud CLI on your local system.

   c. **Backup Process**
      - Step 3.1: Create a local backup folder (C:\Mongo-Backups).
        Step 3.2: Use mongodump to export MongoDB data to BSON files.
        Step 3.3: Compress backup files (Compress-Archive).
        Step 3.4: Upload compressed backup to IBM COS using CLI commands.
        Step 3.5: Verify successful upload on the IBM COS dashboard.

   d. **Recovery Process**
      - Step 4.1: Download the backup file from IBM COS (Mongo-Backups.zip).
        Step 4.2: Extract the compressed file to the local backup folder.
        Step 4.3: Use mongorestore to import BSON files back into MongoDB.
        Step 4.4: Verify successful data restoration.

   e. **Verification and Testing**
      - Step 5.1: Validate database integrity.
        Step 5.2: Test data consistency across collections.
        Step 5.3: Perform query testing to ensure database performance.

   f. **Monitoring and Maintenance**
      - Step 6.1: Set up periodic backup scripts (using cron jobs or Task Scheduler).
        Step 6.2: Monitor backup jobs and IBM COS storage usage.
        Step 6.3: Document backup and recovery procedures.