Module 2.12 Assignment

CE7 Group 2 Members

1. Chua Lai Chwang
2. Dijay Kumar
3. Tan Yuan
4. Lovell Tan
5. Khai Razali
6. Shashipal Singh
7. Wong Teck Choy

What is Cloud Architecture Design - Reliability?

Chat generated reference

The primary goal of AWS cloud architecture design for reliability is to ensure high availability and resilience of cloud-based applications and services, minimizing downtime and maintaining performance under various conditions. Doing so delivers reliable service level, enhances user satisfaction, and supports business continuity by effectively managing risks associated with system failures.

It’s key design principles are:

- **Automatic Recovery**
  - Implement monitoring and automated recovery processes for proactive failure management.
- **Test Recovery Procedures**
  - Regularly design and test recovery plans to minimize downtime during outages.
- **Horizontal Scaling**
  - Use multiple smaller instances to enhance availability and resilience.
- **Automated Change Management**
  - Automate deployments to reduce human error and ensure consistent changes.
- **Multiple Availability Zones**
  - Deploy resources across AZs to protect against localized failures.
- **Load Balancing**
  - Distribute traffic evenly to enhance performance and provide redundancy.
- **Leverage Managed Services**
  - Use AWS managed services for built-in high availability and reliability.
- **Chaos Engineering**
  - Introduce controlled failures to identify weaknesses and improve resilience.

Cloud Architecture Design - Reliability Implementation

How do you manage service quotas and constraints?

1. **Understand Service Quotas and Constraints**:Be aware of the limitations on resource usage and API request rates. Understanding these constraints allows you to plan effectively and avoid unexpected service disruptions.

2. **Manage Quotas Across Accounts and Regions**:Actively track and manage service quotas for different AWS accounts and regions. This ensures that you don't accidentally hit limits, which can lead to service interruptions or delays.

3. **Design Architecture with Quotas in Mind**:When planning your system architecture, consider the fixed quotas and constraints. This might involve distributing workloads across multiple accounts or regions to balance resource usage and avoid hitting limits.

4. **Monitor Quotas Continuously**:Regularly monitor your resource usage and service quotas using tools like AWS CloudWatch. This proactive approach helps you stay within limits and identify potential issues before they become critical.

5. **Automate Quota Management**:Leverage automation tools to manage quotas efficiently. Automate processes like requesting quota increases or scaling resource usage based on demand, allowing for quick adjustments in response to changing conditions.

6. **Build in a Safety Margin**:Always maintain a buffer between your current resource usage and the maximum quota. This safety margin provides a cushion for handling unexpected traffic spikes or failover scenarios without breaching service limits.

How do you monitor workload resources?

1. **Comprehensive Monitoring**: Track the performance and health of all system components—servers, databases, applications—to maintain a complete view.
2. **Key Metrics**: Identify and measure critical metrics (e.g., CPU usage, error rates) to monitor system health effectively.
3. **Proactive Alerts**: Set up notifications for when metrics exceed thresholds, enabling quick response to potential issues.
4. **Automate Responses**: Implement automated actions, such as restarting services upon failure, to address problems immediately.
5. **Data-Driven Insights**: Analyze collected data to identify trends and patterns, guiding informed decisions and performance optimization.
6. **Regular Reviews**: Periodically assess your monitoring setup and metrics to ensure ongoing relevance and effectiveness.
7. **End-to-End Tracing**: Monitor requests across the entire system to pinpoint delays or issues, ensuring smooth operation.
8. **Centralized Dashboard**: Use a centralized monitoring dashboard to visualize all metrics and alerts in one place, making it easier to manage and understand system health at a glance.
9. **Scalability Monitoring**: Track metrics related to scalability, such as load balancing and auto-scaling events, to ensure your system can handle varying workloads effectively.
10. **Security Monitoring**: Include security-related metrics, such as unauthorized access attempts or unusual traffic patterns, to identify and respond to potential security threats.

How do you design your workload service architecture?

As outlined by the AWS Well-Architected Framework, the best-practices and principles for a high-reliability architecture design are as follows:

- **Employ a service-oriented architecture (SOA) or microservices architecture**
  - Break down applications into smaller, independent components for better scalability and reliability
  - Microservices allow for agile management and scaling of individual services
- **Implement automated recovery from failures**
  - Monitor key performance indicators (KPIs) to define acceptable performance levels
  - Set up alerts and automated responses to potential failures for quick recovery
- **Build and regularly test recovery procedures**
  - Prepare for inevitable failures by designing and testing recovery plans
  - Minimize impact on users during outages through effective recovery procedures
- **Utilize horizontal scaling**
  - Deploy multiple smaller instances instead of relying on a single large instance
  - Enhance availability by distributing workloads across instances
- **Manage changes through automation**
  - Reduce the risk of human error by automating deployments and configuration updates
  - Ensure consistent application of changes across the architecture
- **Implement idempotent responses in services**
  - Ensure repeated requests do not overwhelm the system
  - Maintain reliability and performance even under high load conditions

How do you backup data?

AWS Backup

- **Centralized Management**: Utilize AWS Backup to centralize and automate backups for services like EBS, RDS, and DynamoDB, simplifying the backup process.
- **Backup Plans**: Create backup plans with defined policies for backup frequency, retention, and lifecycle management.
- **Cross-Region and Cross-Account**: Enhance durability and disaster recovery by creating backups across different regions and accounts.
- **Compliance and Reporting**: Leverage AWS Backup's reporting and audit features to ensure backups meet compliance requirements and track backup activity.

Amazon Glacier & S3 Glacier Deep Archive

- **Cost-Effective Long-Term Storage**: Store data in Amazon Glacier or S3 Glacier Deep Archive for low-cost, long-term archival. Ideal for data that doesn't require immediate access, with retrieval times from minutes to hours.
- **Data Retrieval Options**: Choose from expedited, standard, or bulk retrieval options based on your needs, balancing speed and cost.

Amazon Aurora

- **Automated Backups**: Benefit from Aurora's automatic backups to Amazon S3, enabling point-in-time recovery.
- **Manual Snapshots**: Capture manual snapshots of Aurora clusters for long-term storage or specific recovery points.
- **Cross-Region Replication**: Set up cross-region replication for Aurora to maintain data availability and quick recovery in the event of regional outages.

Backup Best Practices

- **Encryption**: Ensure all backups are encrypted, both at rest and in transit, to protect data integrity.
- **Regular Testing**: Periodically test backup and recovery processes to ensure data can be restored quickly and accurately.
- **Retention Policies**: Define and enforce data retention policies to manage storage costs and regulatory compliance.

An essential practice of ensuring systems reliability is to regularly test the recovery of data from their backups.

AWS Reference :

1. How do you manage Service Quotas and constraints
2. How do you monitor workload resources
3. How do you design your workload service architecture
4. How do you back up data