AWS Certified Cloud Practitioner
Training Bootcamp

Design Principles – Loose Coupling

Design Principles – Loose Coupling

- What does loose coupling mean ?
- Breaking your application into smaller pieces or components in such a way that they are little to no dependent on each other leads to a loose coupling system
- How can loose coupling be implemented ? What are the options ?
- Let’s explore …

Loose Coupling – Well Defined Interfaces

- Communication between the components should be implemented through open-source mechanisms

- Using open-source communication interfaces (not vendor specific) leads to the possibility of developers to modify and adapt configuration on the fly, during or after project implementation

Loose Coupling – Service Discovery

- Implementing loose coupling means that you will have a lot of services, that need to either communicate with each other or with other services in your environment
- There needs to be a way to address or call any service, in a unique way, *"loosely"*, so that no interdependencies are created
- As an example, think of Load Balancers, you can call a load balancer by using the endpoint name (! Not IP address)

Loose Coupling – Asynchronous Integration

- Asynchronous integration refers to integrations between different services in your infrastructure

- What is asynchronous ?

- If two services can work independently of each other, but together as a system, this means that the system is asynchronous

- As an example, service A is the SNS (email), service B could be the SQS (queuing system)

Loose Coupling – Graceful Failure

- Graceful failure is also another method to increase loose coupling
- When a failure occurs, communication of the failure should be performed into the system and all components should be aware
- *Rerouting* of traffic to healthy services should take place (Route 53 can reroute client's traffic to a healthy EC2 instance that hosts your website)

Thank you