**Proposed Cloud Solution for Capack Galactic 2**

**Background and Introduction**

SparkIT-Game Company is releasing the latest series of the game Capack Galactic, Capack Galactic 2 Treasure Planet. The expectation is that the latest release of the game could become very successful. In anticipation of this demand on IT infrastructure SparkIT-Gamehas requested a cloud based automation system to automatically deploy and remove game servers based on player count in the game with the goal to keep gameplay as smooth as possible at all times.

**Overview of Business Requirements**

Systems should always be provided based on demand with at least a 10% cushion for additional growth. This includes the Primary Game cluster and the Arena Cluster.

The Primary game cluster will consist of the following VM’s per ever 40,000 active players.

* CGSP-2-core (for core services)
* CG2-web-front (for front-end web images)
* CG2-web-back (for back-end web services)
* CG-db (for data services)
* CG-2-Micropayment (for transaction services)

The Co-op (Arena) cluster will consist of the following VM’s per every 20,000 active arena players.

* CG-2-CoopPlayCore
* CG-2-CoopPlay-gw
* CoopPlay-Env for the cooperative play module

There should be at least 1 Co-op Arena Cluster for every 6 Primary Game Clusters that are created.

**Overview of Proposed solution**

Utilized a Chef Server in the Azure cloud to follow the logic of the business requirements to automatically spin the up necessary Virtual Machines as defined in the Chef Server cookbook. The same server will remove instances once they are no longer needed.

A template cookbook for the Chef server will be downloaded from Github (github.com, 2020). Modifications will be made to the cookbook to spin up the appropriate Virtual Machines based on the required conditions needed for SparkIT-Game.

A template kitchen.yml file will be downloaded from WGU’s Google Docs repository (docs.google.com, 2020). This template will be the key component used to create and modify the Virtual Machines.

The primary Chef commands that will be utilized to accomplish these goals will be attached to this proposal in the reference documents (VisualDiagram.pptx). Some commonly used kitchen commands will be:

* *kitchen create*
* *kitchen converge*
* *kitchen destroy*
* *kitchen list*

To identify when additional clusters should be spun up or removed from the Azure cloud the Automation Virtual Machine running Chef will use a combination of common chef and linux commands to identify system load and act accordingly (learn.chef.io, 2020).

Micropayment transactions confirmations and declines will be written to an Azure Data storage object at //data-currency-cg-2-pAPI.

Diagnostic data from the arena cluster will be written to an Azure Storage object at //data-automation-diagnostic-CG-2-game/

Diagnostic data from the arena cluster will be written to an Azure Storage object at //data-automation-diagnostic-CG-2-arena/

Reporting data detailing when a new cluster is activated or deactivated will be sent to a message queuing service in Azure that will send an email to SparkIT-Game helpdesk system at helpdesk@sparkIT\_game.com.

Template CookBook

<https://github.com/learn-chef/learn_chef_httpd.git>

Template YAML file

[https://docs.google.com/document/d/1caeYwKpxCipvwL8tU8ukMt1NEF1Ajl\_bXi8F7bVGLO4/edit#](https://docs.google.com/document/d/1caeYwKpxCipvwL8tU8ukMt1NEF1Ajl_bXi8F7bVGLO4/edit%23)

Gathering Data Commands

[https://learn.chef.io/modules/local-development/rhel/virtualbox/apply-a-cookbook#/](https://learn.chef.io/modules/local-development/rhel/virtualbox/apply-a-cookbook%23/)