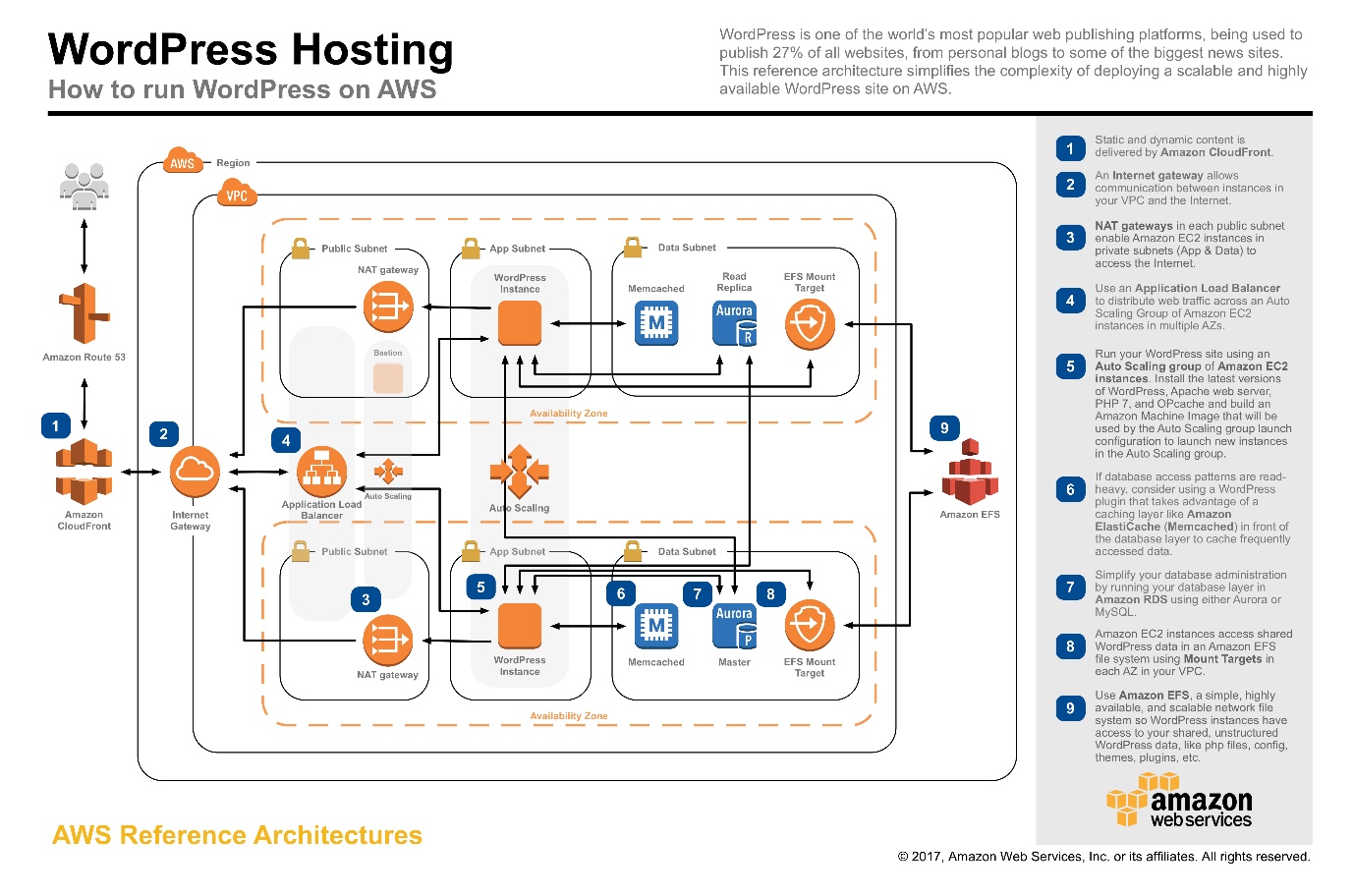
**Sandbox Project**

Welcome to your Sandbox Project, there are several tasks over multiple weeks which you will need to do.

We will be deploying Wordpress and supporting applications over the next several weeks.

[](https://raw.githubusercontent.com/aws-samples/aws-refarch-wordpress/master/images/aws-refarch-wordpress-v20171026.jpeg)

We'll be deploying this to Cloud of choice with a similar architecture.

We'll be using the \*.rack.gold domain. I will delegate a subdomain for you to use.

For each week, you need to do the following:

* A README text or Markdown file SHOULD contain a short description of additional changes to the infrastructure/architecture, how to deploy it (if anything has changed), etc.
* All code and documentation SHOULD be checked into the project Git repo
* A separate "feature" branch SHOULD be used in Git to store the artifacts for this phase of the project
* At the end of the week, a pull request from the feature branch SHOULD be created, targeting the "master" branch, signaling that the work is ready to be reviewed. Ensure that you request review from the Engineer assigned to review your task.

**Getting Started:**

* Make sure you have a sandbox google project.
* Create a github repository under this organization. You may pick any available name but the repo description should contain First Name - Sandbox Project
* You will be using a delegated a subdomain of rack.gold. The rack.gold zone can be found in Route53 under the 596834884942 AWS Account. Access this AWS account and delegate a subdomain for your self. Do not create any other DNS records on this zone. If you are not sure, please read about DNS Delegation and speak to an Engineer.
* The words MUST, MUST NOT and SHOULD have special meaning as per [RFC 2199](https://www.ietf.org/rfc/rfc2119.html). Please read it and adhere to it.

**Cloud Specific Mandatory tasks:**

**AWS**

* All AWS instances MUST be accessible via SSM. Please make sure you are familiar with how to configure SSM on VMs.

**Azure**

* WIP

**Week 1:**

This week is about deploying Wordpress and it's components on to a single VM.

**Requirements**

* Base OS is Ubuntu 2004
* Use the LEMP Stack

**Acceptance Criteria**

* Wordpress MUST be available on port 80 of instance's public IP. This requirement is waived if you have configured SSL for this server.
* Wordpress MUST be available on port 443 of the domain name you have configured for SSL. This requirement is waived if you have NOT configured SSL for this server. You SHOULD serve a redirect on port 80 of the webserver.

**Bonus Points**

* SSL enabled on nginx
* Monitoring and Logging agents installed on the VM. Make sure you install the appropiate agents for the Cloud that you are working on.
* Scriptable install

**Week 2:**

This week is about deploying Wordpress in front a load balancer.

**Requirements**

* Deploy the Google Cloud/AWS Components.
* Base OS is Ubuntu 2004
* Use the LEMP Stack

**Acceptance Criteria**

* Wordpress MUST only be reachable through the Loadbalancer.
* SSL MUST be enabled on the loadbalancer and it must be from a trusted Certificate Authority.
* Wordpress MUST be served properly with no missing content on the page. Please use the [Chrome DevTools](https://developer.chrome.com/docs/devtools/) to ensure that all the content(img,css,js, etc) loads properly.
* If you are using Terraform to deploy any infrastructure, you MUST use a remote state for it.

**Bonus Points**

* Deploy the Loadbalancer via Terraform. However, you MUST create your loadbalancers without using terraform modules.

**Week 3:**

This week is about deploying Wordpress using best practices.

**Requirements**

* Deploy the Cloud Components via Terraform
* Base OS is Ubuntu 2004
* Use the LEMP Stack
* Deploy a Cloud NAT(GCP)/NAT Gateways(AWS)

**Acceptance Criteria**

* Instances MUST NOT have external IPs
* All resources MUST be deployed through terraform
* All the criterias from Week 2 MUST be met.
* You MUST create your loadbalancers without using terraform modules.

**Bonus Points**

* Appropiate IAM roles/permissions SHOULD be assigned to your instances that enables the VMs to send metrics and logs to the Cloud Provider's Monitoring and Logging system.

**Week 4:**

This week is about Jenkins and automating some of the manual steps we are doing.

**Requirements**

* Deploy the required infrastructure to AWS/GCP via Terraform
* Base OS is Ubuntu 2004
* Jenkins must be installed on a VM. (Docker installs aren't accepted)
* Write a Jenkins pipeline that can run terraform plan and apply when there is a change in your repository.
* Configure Github app auth for jenkins so it can read the repositories. <https://github.com/jenkinsci/github-branch-source-plugin/blob/master/docs/github-app.adoc>

**Acceptance Criteria**

* Your Jenkins install MUST be secured by SSL and [Github Authentication](https://plugins.jenkins.io/github-oauth/) .

**Bonus Points**

* [Distributed build slaves](https://cloud.google.com/solutions/using-jenkins-for-distributed-builds-on-compute-engine) We'll deviate slightly from these docs. Ask me when you are about to deploy this.
* Packer Images/AMIs for your build slaves.
* Appropiate IAM roles are assigned.
* Use remote state with terraform.

**Week 5:**

This week is about breaking down the Wordpress monolith.

**Requirements**

* Base OS is Ubuntu 2004
* Replace MySQL with a Managed SQL Service from your Cloud Provider
* Create packer image/ami that preinstalls nginx + php + monitoring/logging agents, via shell scripts or Ansible.
* Write a Jenkins pipeline that can build the packer image/ami.
* Create a NFS share on Filestore(GCP), EFS(AWS),
* Create a startup script that adds the NFS Share to fstab.

**Acceptance Criteria**

* A managed MySQL instance MUST be used.
* The managed MySQL instance must be accessed by its private IP address.
* Wordpress files MUST be accessed from the NFS share.
* All the criterias from Week 3 MUST be met.

**Bonus Points**

* Appropiate IAM roles SHOULD be assigned.

**Week 6:**

This week is about implementing Shared VPC and Hashicorp Vault

**Requirements**

* Deploy Vault on AWS/GCP Virtual Machines.
* Deploy all your VMs on using Shared VPC (GCP)
* Store your wordpress secrets on Vault and use it in terraform.

**Acceptance Criteria**

* Vault MUST be deployed using KMS and GCS/S3/Azure Blob Storage.
* Vault MUST be deployed with [Auto Unseal](https://www.vaultproject.io/docs/concepts/seal) enabled. You must demostrate this by rebooting the vault server during the demo.
* Vault MUST be ran through systemd. Please read about systemd and how to work with it. Protip, installing vault from the hashicorp repositories meets this requirement.
* Vault MUST be secured by SSL. The Cert MUST be issued by a trusted CA.
* You MUST use any auth [method](https://www.vaultproject.io/docs/auth) of Vault except tokens and username/password. I recommend Cloud Auth + Github.
* Shared VPC networks MUST be used (GCP)
* Appropiate IAM roles MUST be assigned.

**Week 7:**

This week is about fully automating the Wordpress deployments.

**Requirements**

* Using Managed Instance Groups, provision atleast 2 VMs from the image last week along with a bootstrap script. (GCP)
* Using EC2 Autoscaling, provision atleast 2 VMs from the image last week along with a bootstrap script. (AWS)

**Acceptance Criteria**

* There MUST be instances covering at least 2 zones (GCP)
* There MUST be instances covering at least 2 availability zones (AWS)
* All the criterias from Week 6 MUST be met.
* Appropiate IAM roles MUST be assigned.