NoteJam Architecture

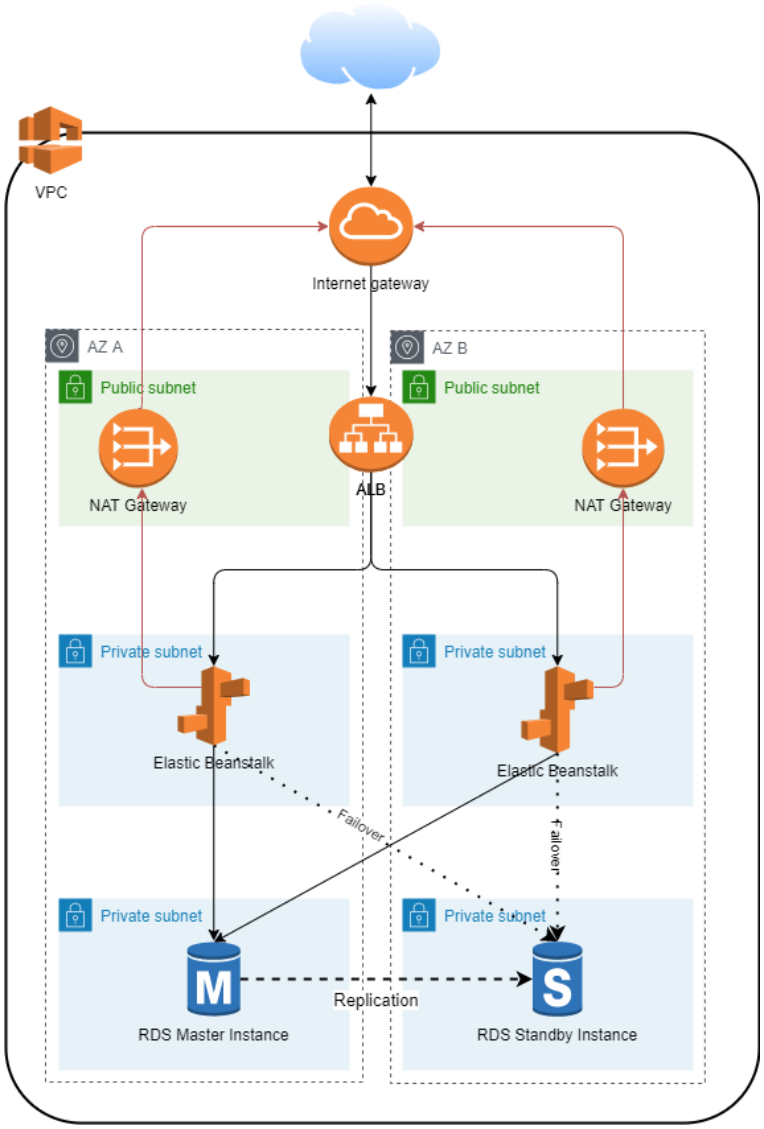
Scenario:

Notejam is currently a monolithic application using a single server with embedded database. Customer wants a pilot in AWS refactored to meet their requirements.

Requirements:

1. The Application must serve variable amount of traffic.
2. Most users are active during business hours. During big events and conferences, the traffic could be 4 times more than typical.
3. The Customer takes guarantee to preserve your notes up to 3 years and recover it if needed.
4. The Customer ensures continuity in service in case of datacenter failures.
5. The Service must be capable of being migrated to any regions supported by the cloud provider in case of emergency.
6. The Customer is planning to have more than 100 developers to work in this project who want to roll out multiple deployments a day without interruption / downtime.
7. The Customer wants to provision separated environments to support their development process for development, testing, production in the near future.
8. The Customer wants to see relevant metrics and logs from the infrastructure for quality assurance and security purposes.

Proposed Production Architecture



Components

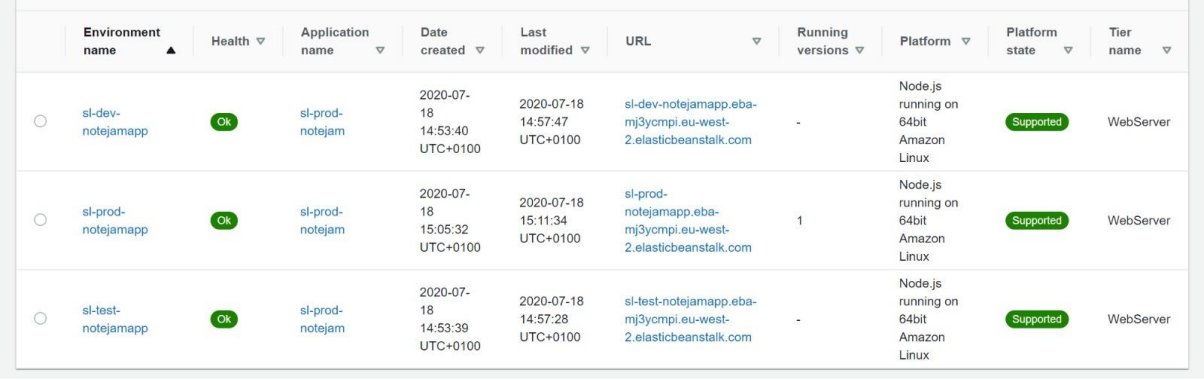
VPC:

- NAT Gateway for internet access
- Public subnet in each AZ for load balancers
- Private subnet in each AZ for Elastic Beanstalk application layer
- Private subnet in each AZ for RDS Database

Application Tier:

Decision Point: Elastic Beanstalk (EBS) was chosen for the application layer as it provides easier centralised management of underlying resources. In order to meet the requirement of allowing

several releases a day, EBS facilitates this by allowing application releases by simply uploading an application .zip file which is automatically deployed to all nodes. It centralises performance monitoring and additionally, there are no extra charges for EBS over the underlying resources that are provisioned.



Environment name	Health	Application name	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name
sl-dev-notejamapp	Ok	sl-prod-notejam	2020-07-18 14:53:40 UTC+0100	2020-07-18 14:57:47 UTC+0100	sl-dev-notejamapp.eba-mj3ycmpi.eu-west-2.elasticbeanstalk.com	-	Node.js running on 64bit Amazon Linux	Supported	WebServer
sl-prod-notejamapp	Ok	sl-prod-notejam	2020-07-18 15:05:32 UTC+0100	2020-07-18 15:11:34 UTC+0100	sl-prod-notejamapp.eba-mj3ycmpi.eu-west-2.elasticbeanstalk.com	1	Node.js running on 64bit Amazon Linux	Supported	WebServer
sl-test-notejamapp	Ok	sl-prod-notejam	2020-07-18 14:53:39 UTC+0100	2020-07-18 14:57:28 UTC+0100	sl-test-notejamapp.eba-mj3ycmpi.eu-west-2.elasticbeanstalk.com	-	Node.js running on 64bit Amazon Linux	Supported	WebServer

Spec:

- Environments: 3 – Production, Dev, Test
- T2.micro instances (on demand) – Prod environment
    - Autoscaling Group - min 1, max 4 instances
- T2.micro instances (spot) - Test & Development environments
    - Autoscaling Group – min 1, max 4 instances

Decision point: Spot instances were chosen for test & dev environments for cost savings. For the purposes of this demo only, micro instances were chosen.

- Deployment method: rolling health based
- Max batch size: 1
- Min in service: 1

Decision point: Rolling method was chosen as it allows app updates without downtime. Health based ensures deployment pass a health check before continuing. The batch size and minimum in service values ensure the application stays in service throughout the update process.

Database Tier

Decision Point: Database was moved to RDS (MySQL) in order to meet redundancy and scalability criteria. The database is better suited to handling large volumes of traffic, can be replicated to another AZ for redundancy, and can scale using features such a read replicas.

Spec:

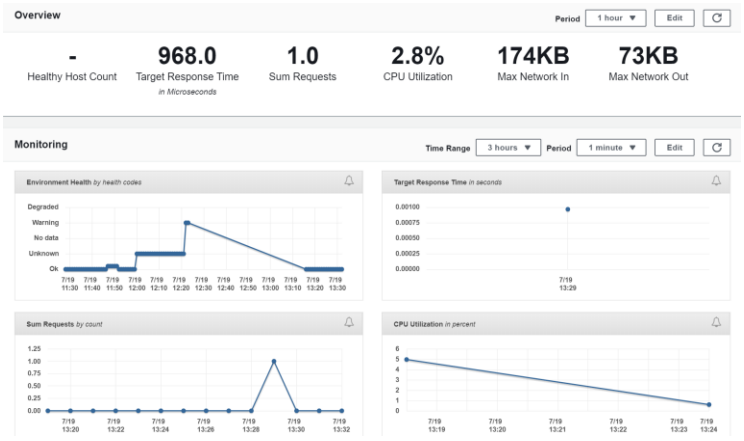
- T2.Small instance – for the purpose of this demo, a small size was chosen. T2.Small is the smallest which supports at-rest encryption.
- Encryption – Enabled
- Multi AZ – Enabled for Production. Single instance for Test & Dev environments.
- Storage
    - GP2 – Production
    - Standard (magnetic) – Test & Dev

- Backup retention period - 35 days (maximum rds retention period)

Decision Point: SSD storage was chosen to handle production demands, where as magnetic storage was chosen for test & dev environments to facilitate cost savings where higher performance may not be required. This can be adjusted after assessment.

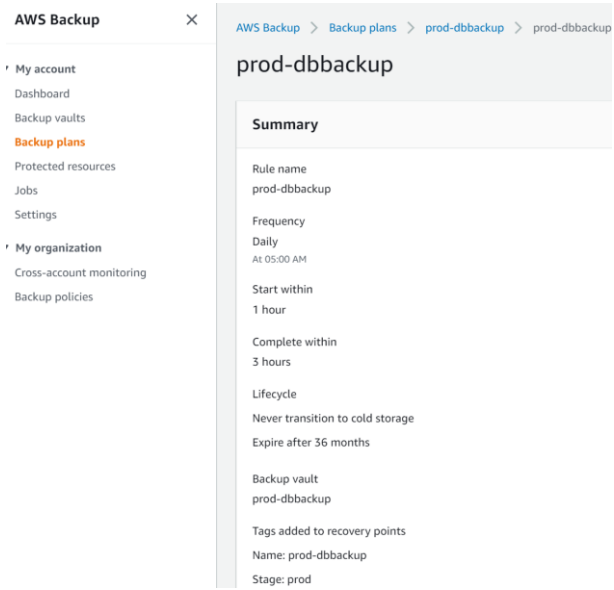
Monitoring

Monitoring is provided within an easy to use dashboard within Elastic Beanstalk



Backups

Decision Point: RDS backups have a maximum retention of 35 days. Therefore, AWS Backup was chosen to facilitate the 3 year retention period for the Production database.



Requirements matrix

Requirement	Requirement met	Met by
The Application must serve variable amount of traffic.	Yes	Auto scaling group scales up to meet increased demand, and back down when demand decreases. More instances provide more network capacity. Nat Gateway scales with demand.
Most users are active during business hours. During big events and conferences, the traffic could be 4 times more than typical.	Yes	Auto scaling group scales to meet demand
The Customer takes guarantee to preserve your notes up to 3 years and recover it if needed.	Yes	AWS Backup can retain the backups for 3 years.
The Customer ensures continuity in service in case of datacenter failures.	Yes	The application is spread across 2 AZs, with the production database replicated to another AZ for redundancy.
The Service must be capable of being migrated to any regions supported by the cloud provider in case of emergency.	Yes	Infrastructure can be reployed to another region by editing the region variable in the terraform script. RDS database would need to be restored from snapshot to new region.
The Customer is planning to have more than 100 developers to work in this project who want to roll out multiple deployments a day without interruption / downtime.	Yes	Elastic Beanstalk provides easy application deployment to developers. The roll deployment method ensures no downtime. See Future Improvements section to improve this further.
The Customer wants to provision separated environments to support their development process for development, testing, production in the near future.	Yes	Separate Production, Development & Test environments are provisioned by the terraform script.
The Customer wants to see relevant metrics and logs from the infrastructure for quality assurance and security purposes.	Yes	These are provided in an easy to use dashboard within Elastic Beanstalk.

Future Improvements:

- Implement WAF for advanced web threat protection
- Utilise VPC endpoints for Beanstalk & RDS for additional security
- Implement a CICD pipeline with Code Commit & Code Deploy or other similar tools to further streamline the application development and deployment process
- Investigate containerising the application, potentially use Fargate to further abstract the infrastructure.

Code repository:

https://github.com/simonlush/notejam