# Solution Architecture Review

—

Review Template

# Project Description

The purpose of this project is to develop an e-commerce web application for The Funny Hat Shop. The project includes building a solution architecture according to the IT architecture vision and business requirements.

The business outcome is to drive additional sales by implementing an e-commerce web site to sell its products online

The web site should be easy to use, but fast and friendly to use on any channel, such as mobile phones, tablets and personal computers.

The Funny Hat Shop made an agreement with an external Vendor for deliveries of the products to the customers. The solution needs to integrate with the delivery companies API’s.

# In Scope

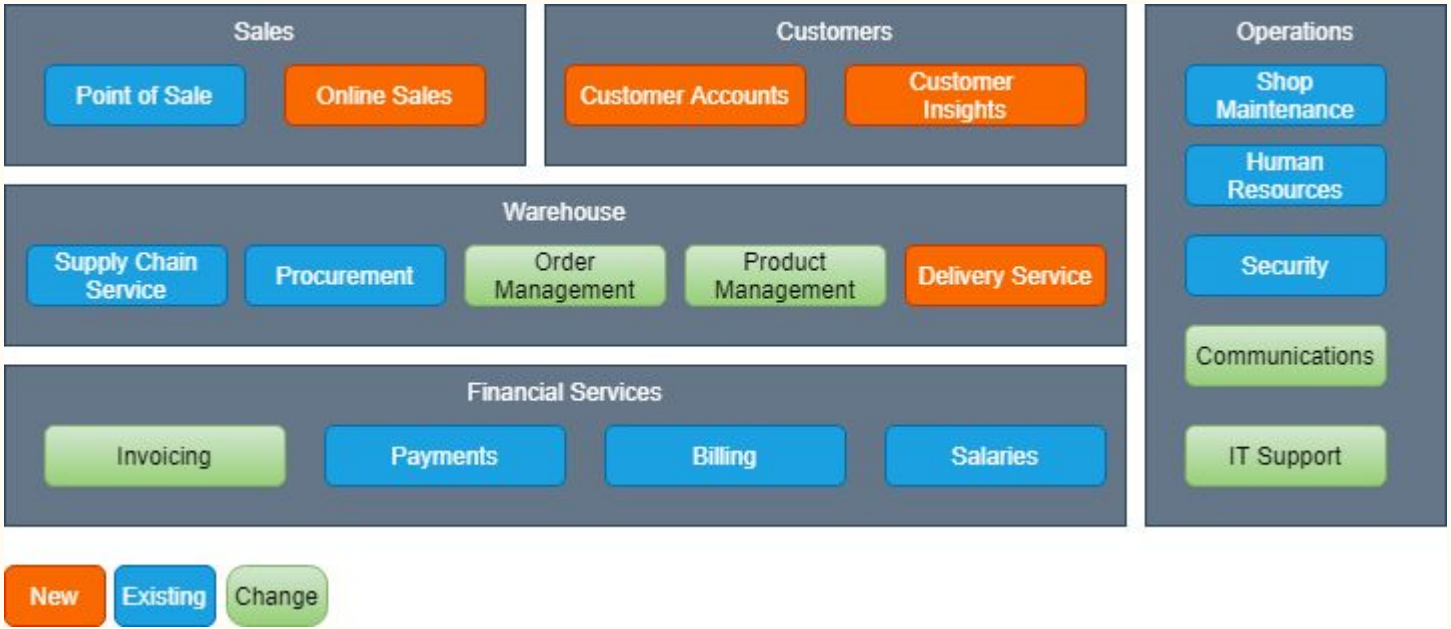
The following architecture building blocks are in scope of this project:

- Create an e-commerce web application hosting on the AWS platform
- Integration with the stock management and invoicing systems
- Stand up a e-mail service to communicate to customer
- Work with a new Delivery Vendor to pick up and deliver products sold on the web site
- Integrate with the Delivery Vendor API to automate status updates and delivery requests
- Add images to the stock management system for each product

# Out of Scope

Upgrading the stock management system is not in scope

# Business Architecture: Capability Map



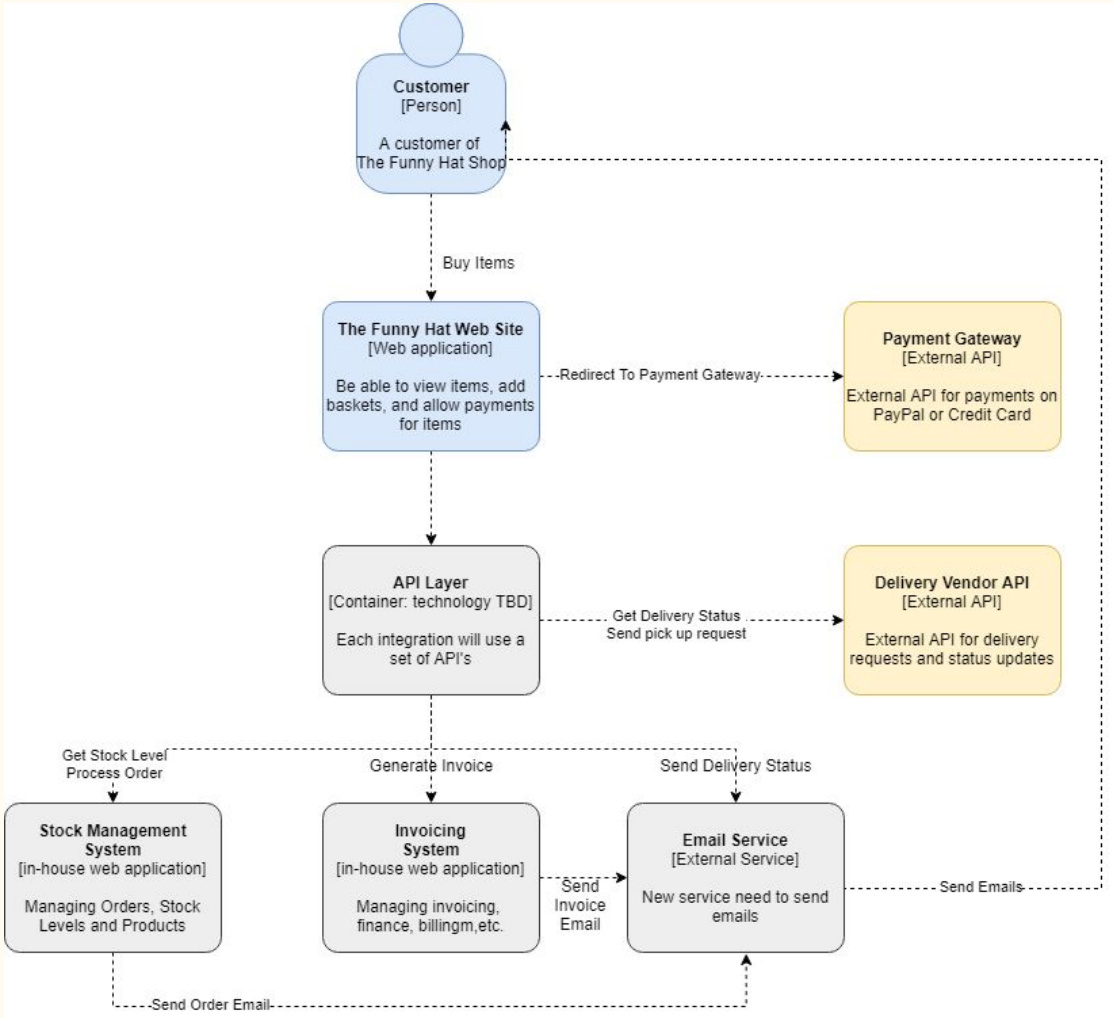
# Application Impacted

| Business Capability | Application/System | Buy/Build/Reuse | Application Roadmap |
|---|---|---|---|
| Online Sales | E-commerce Web Application | Build | New |
| Customer Accounts | CRM Database | Build | New |
| Customer Insights | BI Reporting | Build | New |
| Order Management | Stock Management System | Reuse | Change |
| Product Management | Stock Management System | Reuse | Change |
| Delivery Service | External Vendor - Funny Delivery | Buy | New |
| Invoicing | Invoice System | Reuse | Change |
| Communications | AWS SES Service | Build | New |
| IT Support | AWS CloudWatch | Build | New |

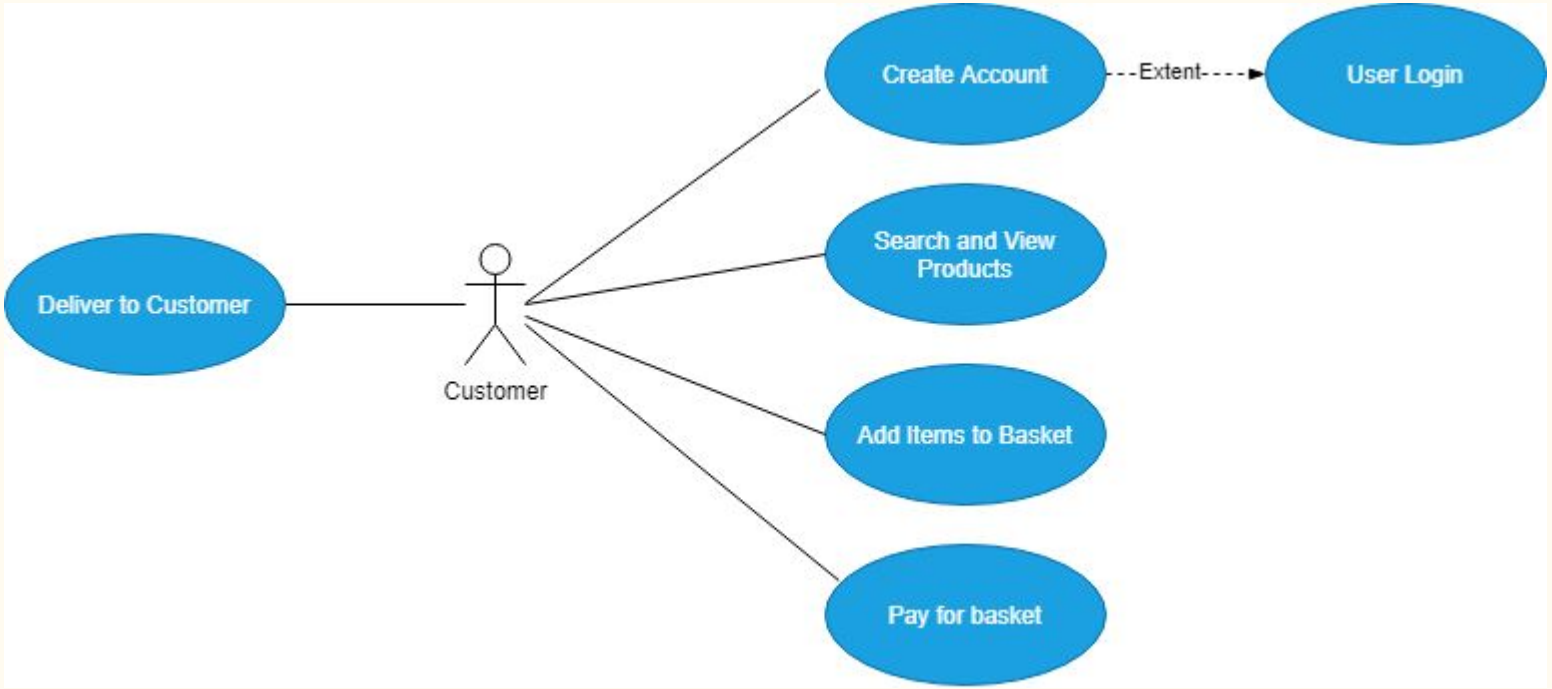
# AS-IS Context Diagram

No As-Is architecture exist

# TO-BE Context Diagram



# Use Case View



# Non-Functional Requirements

| Availability | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Hours of Operations** | Describe hours of operation for the Solution. E.g. 24/7 or weekdays only between 8am and 5pm | 24/7 |
| **Availability (%)** | Percentage of time that the application, process or capability needs to be available (i.e. 99.99%) | 99.99% |
| **SLA** | Service Level Agreement as measured by time (E.g. 30 minutes for completion of the process or batch). If external interfaces are part of the solution, provide the SLA for these individual interfaces (E.g. file delivered by 2 AM) | Deliver emails to customers not more than 10 minutes after order placed |
| **Maintenance/Upgrade Time** | When do you plan for upgrades on the system? Define a maintenance window (E.g. Sundays from 1:00 AM – 4:00 AM) | Planned downtime for Monday mornings 1 to 2 hours only at 1am |
| **Unplanned Downtime Impact** | Describe the impact of any unplanned downtime on the business or process | Sales will be impacted |
| **Batch Processing Times** | Define any batch process time windows (E.g. batches can only run between 2 AM and 4 AM) | Batches are expected to run to check delivery status. This should run every 10 minutes |

# Non-Functional Requirements

| Performance | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Response Time** | Measurement in time of the response of the system, to include the average expected response time and the maximum acceptable response time. This can either be the screen response time (E.g. Less than 2 seconds) | - Website response time less than 2 seconds<br>- Allow for a thousand hit per day<br>- Solution should allowed for auto scalability on evenings and weekends |
| **Data Latency** | Define the acceptable data latency of data required for the Solution. E.g. customer address can be a maximum of 1 day. | No data latency expected |

# Non-Functional Requirements

| Volumes | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Average Volume Estimates** | Measure of the average volume of the solution over a period of time such as daily or monthly (E.g. 1000 transactions per day) | 500 transactions per month, but website can be hit 1000 times per day |
| **Peak Estimates** | Define the peak time period for high volumes of data. (E.g. end of the month) Also define a peak time time period during a day or week (E.g. Between 5pm and 8pm or weekends) | Peak times expected in the evenings and weekends |
| **File Sizes & Interface Type** | Size in gigabytes, megabytes or kilobytes of any files that is required for processing E.g. interface size on a web services, image files or any other document files, etc.) | Product images are stored on AWS S3 Buckets for the stock management. Images are about 50kb average |
| **Backups / Archiving** | Describe the percentage of data that needs to be either purged or archived, and define the percentage over time. Also indicate when and how backups should be taken. | The solution should include replication for the database. Daily backups using AWS Backup top the AWS RDS database |
| **History** | Describe how long you need to keep data history. E.g. the solution needs 2 years of history for online transactional data | Keep sales for 10 Years |
| **Database Growth Projections** | Indication of the percentage increase or decrease of the size of the database over time. E.g. 5% increase over the first 12 months | 10% growth expected after 12 months |

# Non-Functional Requirements

| User Interactions | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Number of Total Users** | Total number of users expected to use the solution | The company expect 500 customer for the first 3 months, and 2000 after 12 months |
| **Number of Concurrent Users** | Number of concurrent users expected to use the system at the same time. | Allow for at least 10 concurrent users accessing the web site |
| **User Locations & User Totals** | Provide total number of users per each locations. Describe from where in the world will the solution be accessed or allowed to be accessed | User locations should only be from America, Europe and South Africa |
| **User Roles & Role Totals** | List the user roles that will access the system. Include the total number of users per role. E.g. 2x admin, 1000x customer, 1x store manager, etc. | 1x administrator, 2000 customers, 1x store manager |

# Non-Functional Requirements

| Business Continuity | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Business Continuity Plan** | Describe the high level Business Continuity Plan. This is usually a created by business and IT management or IT governance. | Utilize AWS high availability, by using multi-availability zones. Use AWS CloudFront for pushing content to 3 regions, us-east-1, eu-west-1, af-south-1 |
| **Minimum Acceptable Number of Users** | Give a percent that indicates the minimum number of acceptable users in a disaster recovery situation. E.g. total users are 200, but in a disaster recovery, allow only for 10% user base | Not applicable |
| **Minimum Acceptable Number of Transactions** | Give a percentage indicating the minimum number of acceptable transactions in a disaster recovery situation. For example 50% of the average transaction volume | Not applicable |
| **Minimum Acceptable Availability** | Describe the minimally acceptable hours of operation and/or availability of the system in a disaster recovery situation | Not applicable |
| **Acceptable Performance Degradation** | Give a percentage indicating the acceptable performance degradation that is acceptable in a disaster recovery situation. E.g. the existing UI performance is less than 2 seconds, however in a disaster recovery situation acceptable performance would be 4 seconds | Not applicable |

# Non-Functional Requirements

| Security | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Authentication** | Describe what authentication is required, and if required additional details should be provided to further clarify the authentication requirements (for example):<br>• Username / Password Requirements (Complexity, Length, Numerical Numbers, etc)<br>• Password change requirements (initial login, every 30 days, etc.)<br>• Password Storage | Use AWS Cognito to implement user authentication with Google and Facebook logins |
| **Authorization** | Details on the authorization requirements for the system. E.g. is the user or system authorized to access the database or API's | Ensure that only the web application can access the database. |
| **Attestation** | Details on the attestation requirements for the solution. E.g. managers need to review or approve user access, or process owner needs to review or approve user access | No attestation needed |
| **Audit Controls** | Details on any audit controls that are required for the solution | When user change address, add an audit table to capture the change |
| **Confidentiality of Data** | Details on the confidentiality of the data for the solution. E.g. is it customer personal information, secret information or public information | Customer data should be encrypted on the database |

# Non-Functional Requirements

| Security | | |
|---|---|---|
| | **Description** | **Answer** |
| **Integrity of Data** | Requirements in regards to the integrity of the data. E.g. the integrity of the public facing internet site would be considered high due to the reputational risk associated with unauthorized changes to the content. | Validate the the customer is a real person. Use captcha to validate that it is not a bot trying to add an account.<br>Use SMS to verify that the customer is valid. Use AWS Cognito |
| **Logging Requirements** | Detail on whether logging is required. Provide further details for logging:<br>• System Events to Log: Security, Configuration, Admin Events<br>• Log File Details: Date, Time, User Initiating Activity, Details, etc.<br>• Time period for keeping log information<br>• Required reviews of log information (timing / responsibility) | Build exceptional handling to add an error to the system events.<br>Developers should create a log file for error handling.<br>Keep log files for 7 days, then auto delete the log files |
| **In Transit / At Rest** | Describe the security requirements when data is in transit and when data is in storage or at rest. | Use X509 Certificates for data in transit.<br>Use AWS KMS for security at rest on S3 Buckets<br>Ensure AWS RDS uses the AES-256 encryption algorithm |
| **Functional Level Authorization** | Details on the Functional Level Authorization requirements for the solution.E.g. Is a user allowed to perform a specific function on the system | Not required |
| **Data Level Authorization** | Details on the Data Level Authorization requirements for the solution. | Not required |

# Non-Functional Requirements

| Operations and Monitoring | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Monitoring Requirements** | Describe the monitoring requirements for the solution | Add AWS CloudWatch to check system health and report using AWS SNS if any issues occur |
| **Operational Support Roles and Responsibilities** | Describe any operational support roles and responsibility requirements for the final solution | IT Operations should monitor support emails from AWS CloudWatch |

# Non-Functional Requirements

| Networking | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **LAN/WAN Requirements** | Describe any LAN or WAN requirements for the solution | Not applicable |
| **Cloud Networking** | Describe any Cloud Networking requirements. E.g. VPC on AWS or VNet on Azure. Include the address spaces if known | Create new VPC, use a public subnet for the web site, and private subnet for the database |
| **Network Bandwidth Requirements** | Describe the the network bandwidth requirements for the solution | Data : 5 kbps up and down<br>Image : 500 kbps up and down |
| **Inbound/Outbound Traffic** | Requirements on whether a firewall usage is required for the solution for inbound and outbound traffic | Configure security groups for inbound traffic to allow all on public subnet, and deny all on private subnet |
| **Internal / External Connectivity** | Describe any internal or external connectivity requirements for the solution. E.g. VPN, SFTP, API Gateways, etc. | Use AWS API Gateway to access Delivery Vendor API's. Vendor requires to whitelist the IP Address and accepts a secret and key. |
| **Load Balancing** | Describe whether a load balancer is needed for the solution | Use Application load balancer for handling web site traffic to multiple EC2 instances |

# Non-Functional Requirements

| User Interface Requirements | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **User Interface** | Details on the User requirements for the solution. If this solution is required to run on a laptop or desktop, describe the hardware requirements. | This is a web application that will only be accessed via browsers |
| **UI Channels** | Describe which channels will use the solution. Desktop, Mobile, Browsers, Tablets, IoT Devices, etc. | Web application should be access from the following browsers: Chrome, Internet Explorer, FireFox, Edge and Safari |

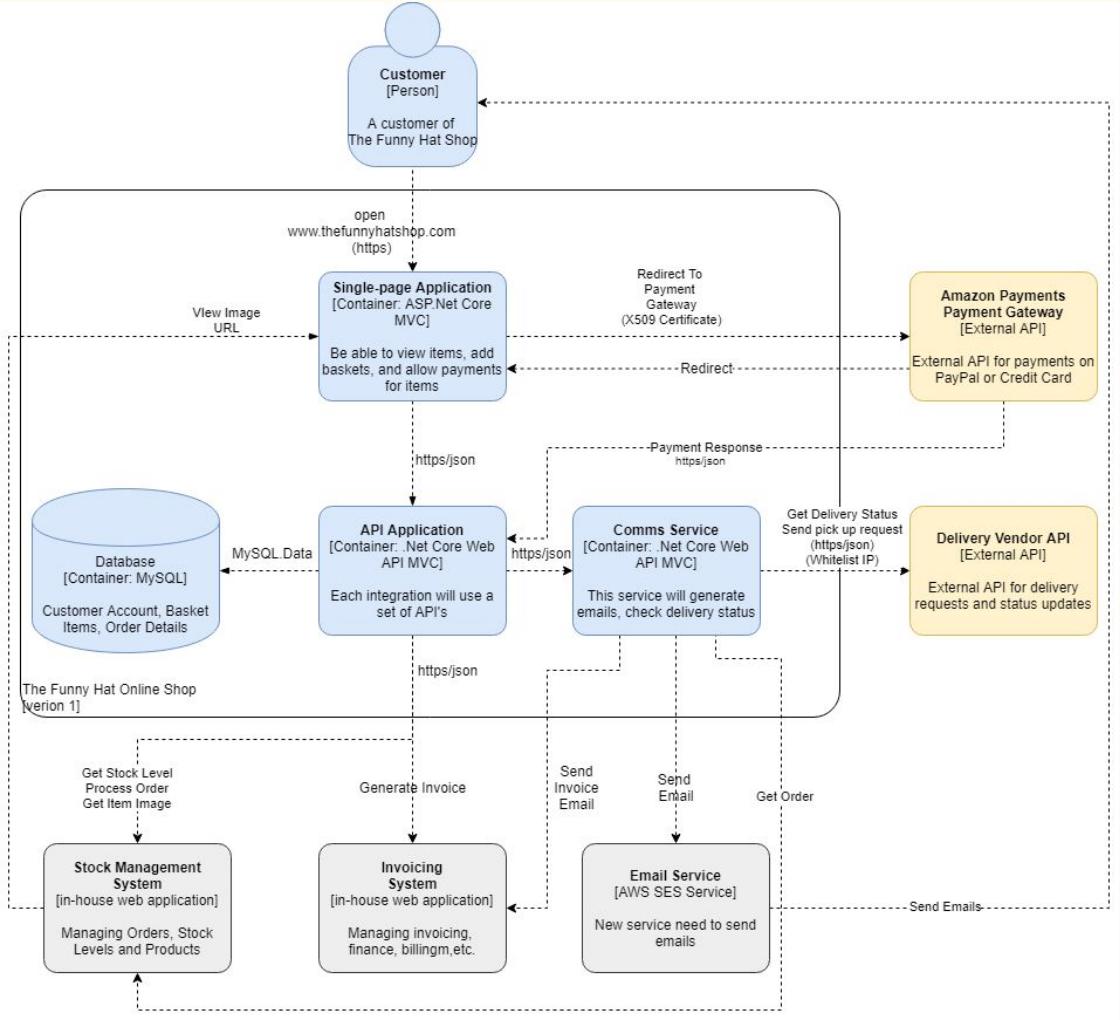
# Non-Functional Requirements

| Architectural Requirements | | |
|---|---|---|
| **Requirement** | **Description** | **Answer** |
| **Architecture Requirements** | Describe any specific architecture requirements for the solution:<br>·System Design - N Tier, Web Services, API, Containers, Kubernetes, etc<br>·Required Patterns, Pub/Sub, Queues, ESB, Microservices, etc. | Use the Microservices architecture patterns. Design should include using containerization technology, such as docker. API's should use RESTful web services using json files. |
| **Environment Requirements** | Details on the number of environments required and sizing for each of the environments. E.g. the application will require Development, Quality Assurance, User Acceptance Test, and Production environments. Production and UAT might have the same size, while Development, Quality Assurance can all be 10% of the Production environment | Create 4 environments: Dev, QA, UAT and Production. Each environment needs to be in its own VPC on AWS |
| **Virtualization Support** | Information on any requirements for virtualization support within the solution. E.g. on-premise use VMWare, Cloud use EC2 or ECS | Use EC2 to and apache to host the web application. |
| **Development Stack** | Detailed requirements on development stack that need to be supported as part of the build of the solution | Use the following development stack: C# ASP.Net Core 3.1, MVC Pattern, XUnit, Bootstrap, Docker, MySQL on AWS RDS, Apache Web Server, Angular and Javascript |

# Current State Solution Architecture Diagram

There is no current state. This is a new e-commerce web application.

# Target State Solution Architecture Diagram



# Architecture Objectives

- Create a secure private network on AWS using VPC’s for all environments
- Create a responsive single-page web application using bootstrap
- Use AWS Platform services for the solution, such as EC2, Elasticache for Redis, Amazon RDS on MySQL, ALB load balancer, Route 53 Hosted Zones and CloudWatch for monitoring
- Ensure auto scaling is enabled on the load balancer
- Deployments should use a DevOps pipeline for continuous integration and continuous deployments
- Adhere to security standards for both data in transit and data at rest by using X509 Certificates, AWS KMS encryption on S3 Buckets

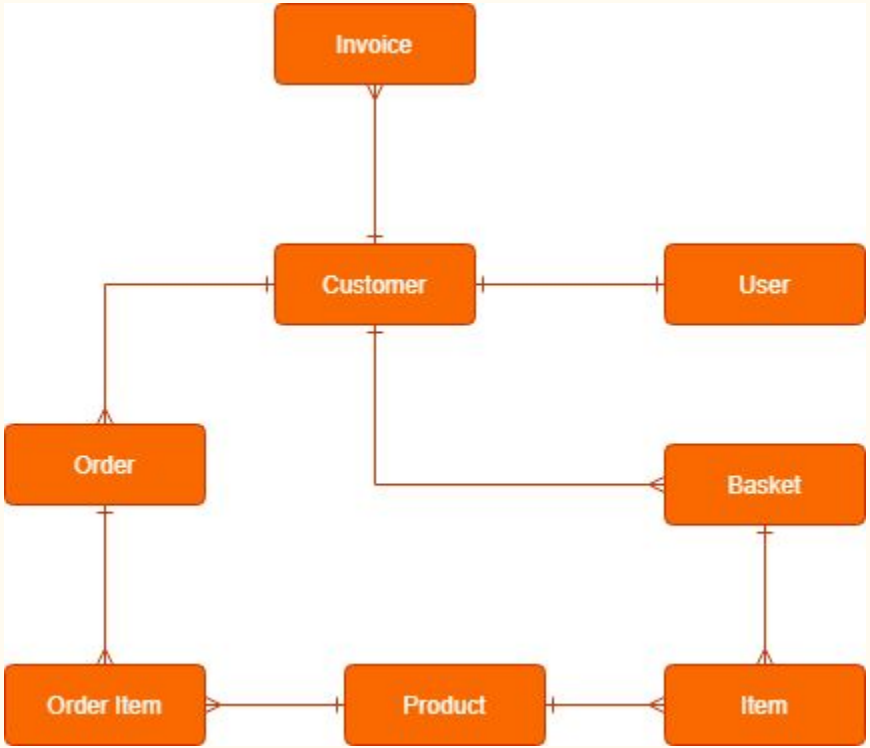
# Architecture Constraints

- The business only have a limited budget of $200,000 set out for development. The solution should consider as a constraint.
- The company relies on this solution to keep the business open and would like this solution in 3 months
- The company standard is to only use the Amazon AWS Cloud services

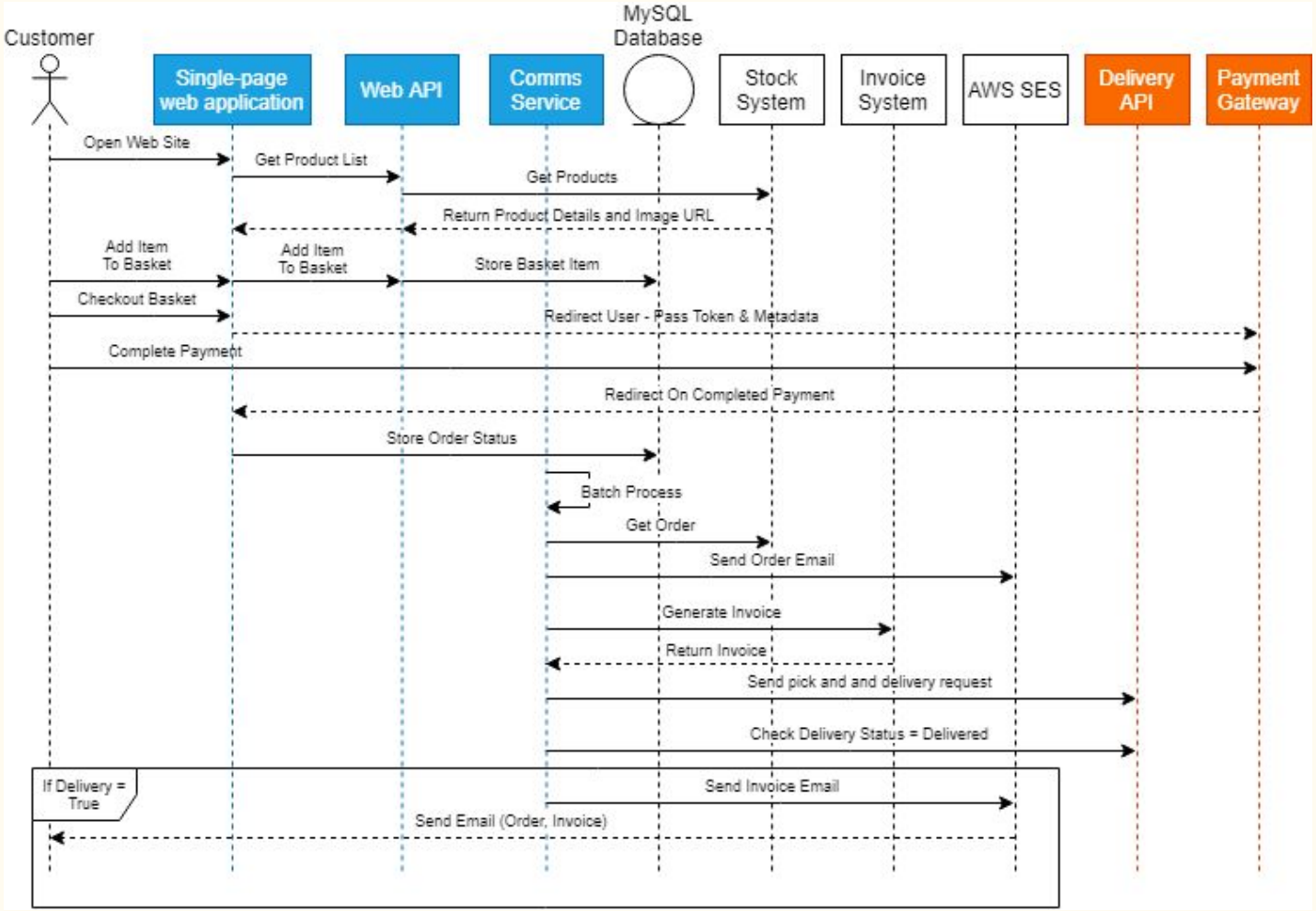
# Architectural Risks

| Risk | Mitigation | Owner |
| --- | --- | --- |
| Project Not Delivered in time | Business are willing to get contractors to speed up development | Business User (Sarah) |
| IT Operational Cost Might be high | Solution Architecture will include auto scaling capability | Solution Architect |

# Data Model Diagram

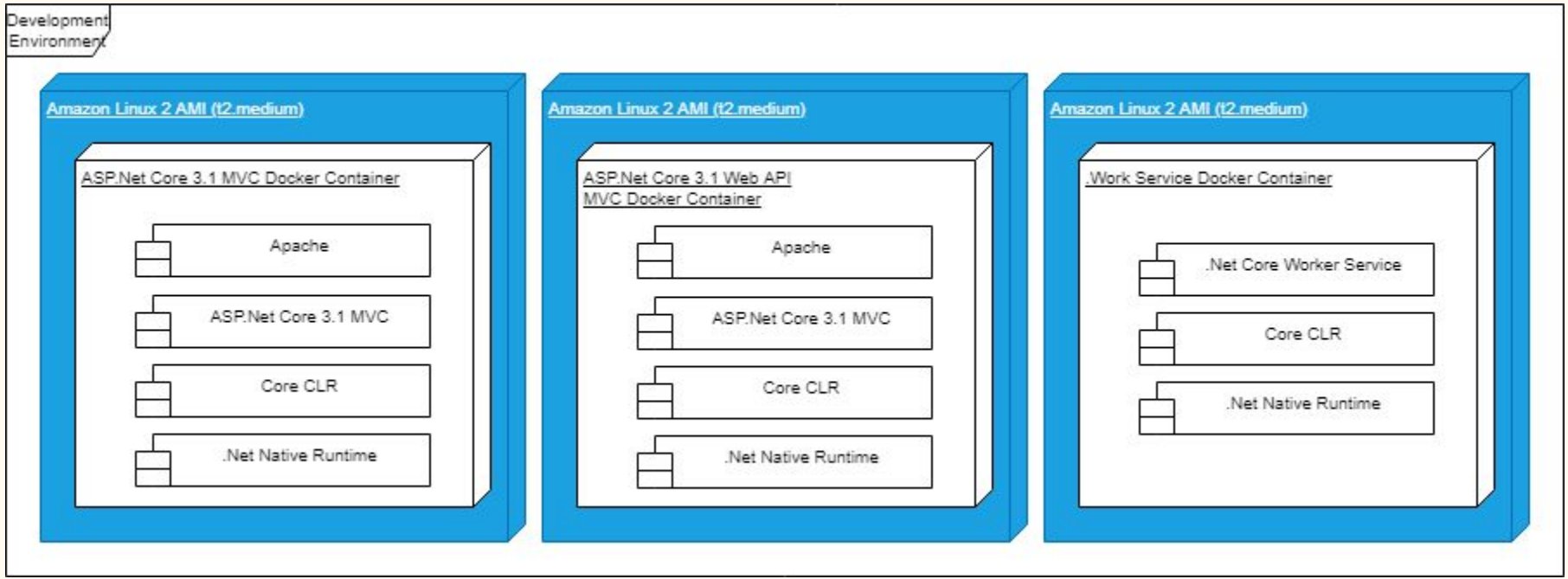


# Data Interaction Sequence Diagram

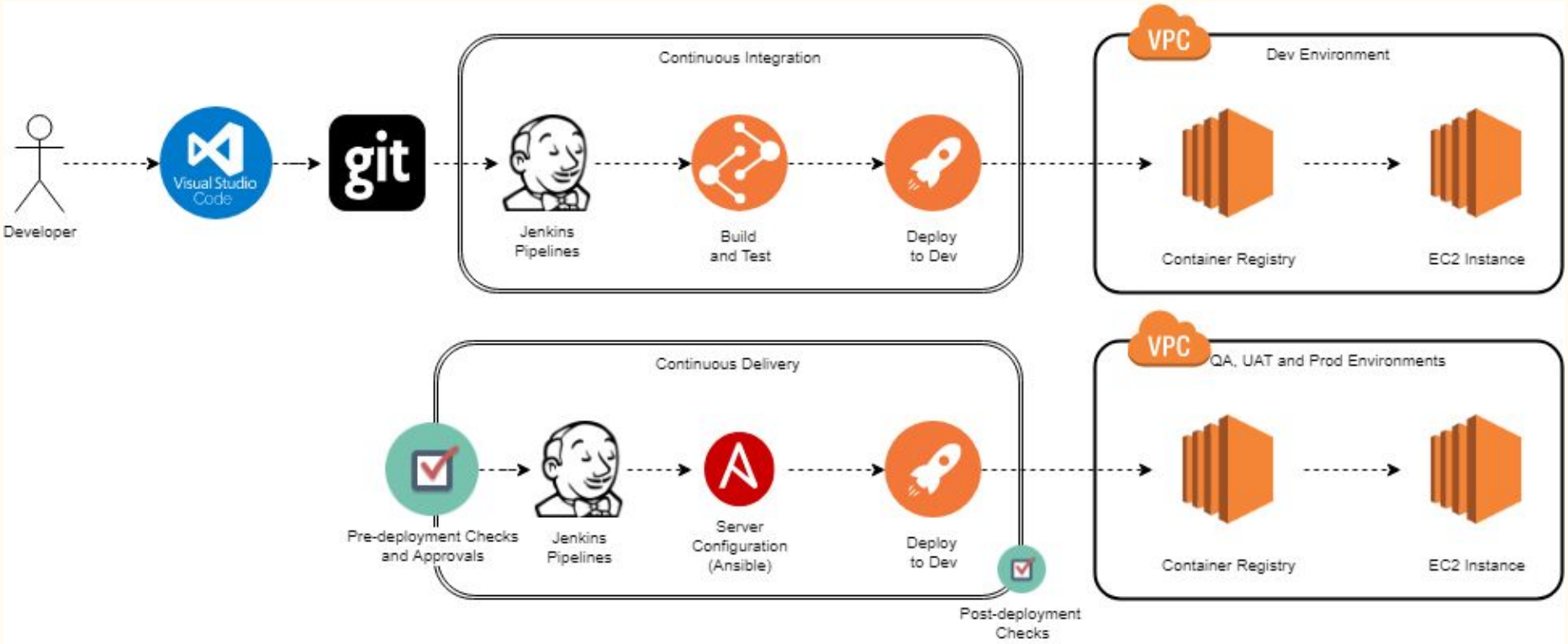


# Application Component Model

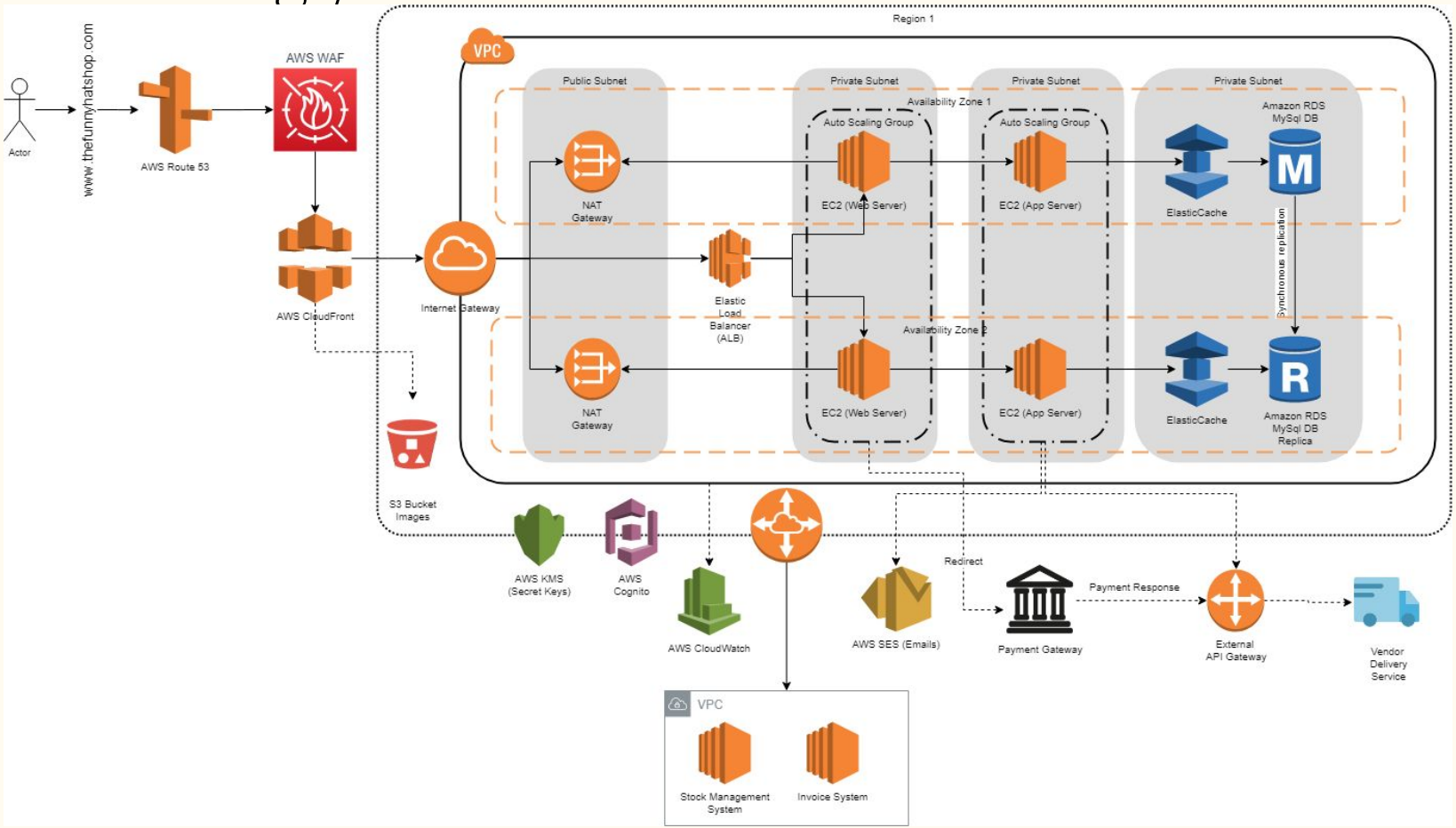
# Application Development Stack



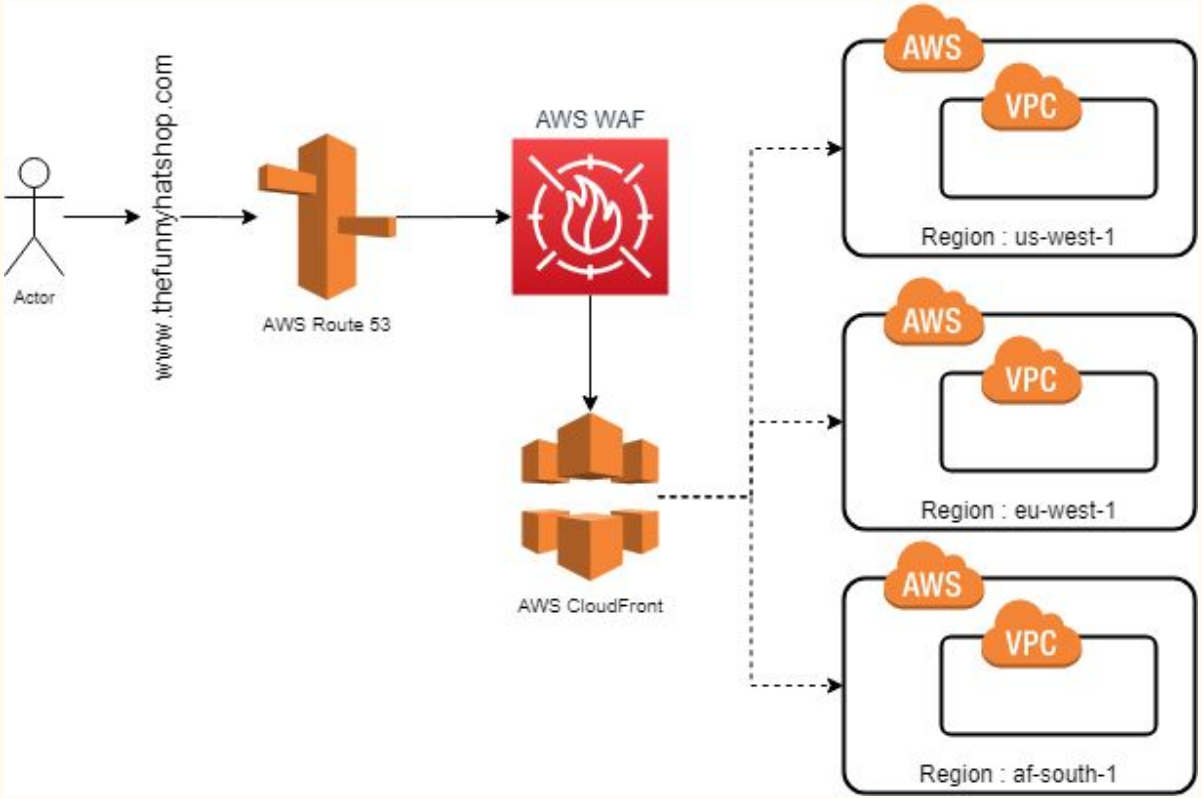
# Deployment Model (DevOps)



# Networking / Infrastructure



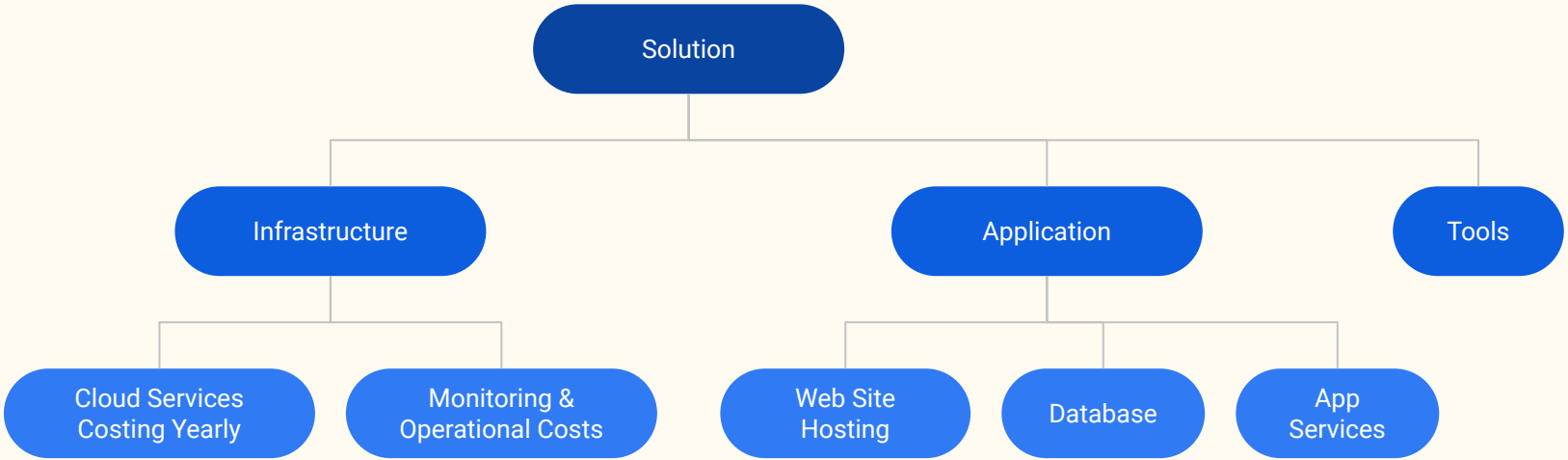
# AWS Regions


Actor
www.thefunnyhatshop.com
AWS Route 53
AWS WAF
AWS CloudFront
AWS
VPC
Region : us-west-1
AWS
VPC
Region : eu-west-1
AWS
VPC
Region : af-south-1

# Development Requirements

| | |
|---|---|
| Development | The Funny hat store doesn't have their own developers, and this project will be outsourced. (In a real-life example, teams are split into different areas of concerns, E.g. Systems of Insight. Systems of Record, Systems of Integration and Systems of Engagement) |
| Configuration | Create (Amazon Machine Images) AMI images for three container services using Ansible playbooks: Single-page web application, Web API application, and Comms Service. Use AWS KMS for secret keys and AWS Cognito service to control and authorize user access to the web site. |
| Databases | Create and design new database, thefunnyhatDatabase on MySQL AWS RDS. Database should include synchronous replication between different availability zones. Enable RDS encryption. Use db.m5.xlarge MySQL Instance. Use Multi-AZ deployment |
| File Storage | Attach an AWS Elastic Block Storage to each EC2 Instance for configuration files, exception files, and log files. Allocate 10gb per instance. |
| Data Migration | The only data capturing required is the images to be added to the stock management system. Use S3 Bucket to store the images, include a URL for each image on the stock management database. |
| Deployment Strategy | Refer to the logical deployment model, by using the DevOps deployment strategy |
| Reports | Create a custom sales report |

# Cost Model Breakdown

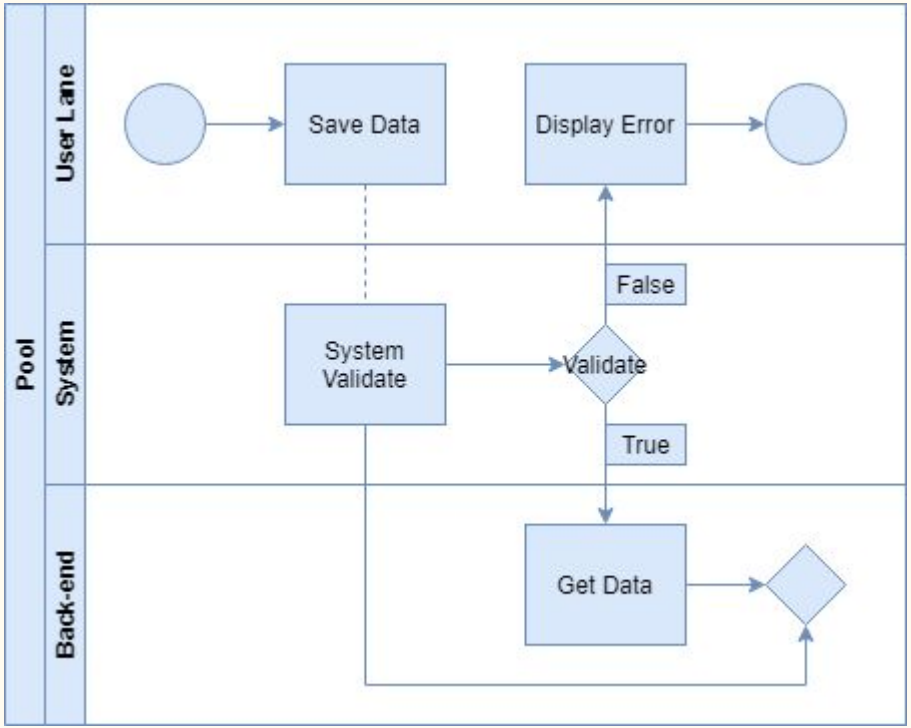


# Example Amazon AWS Cost - Per year

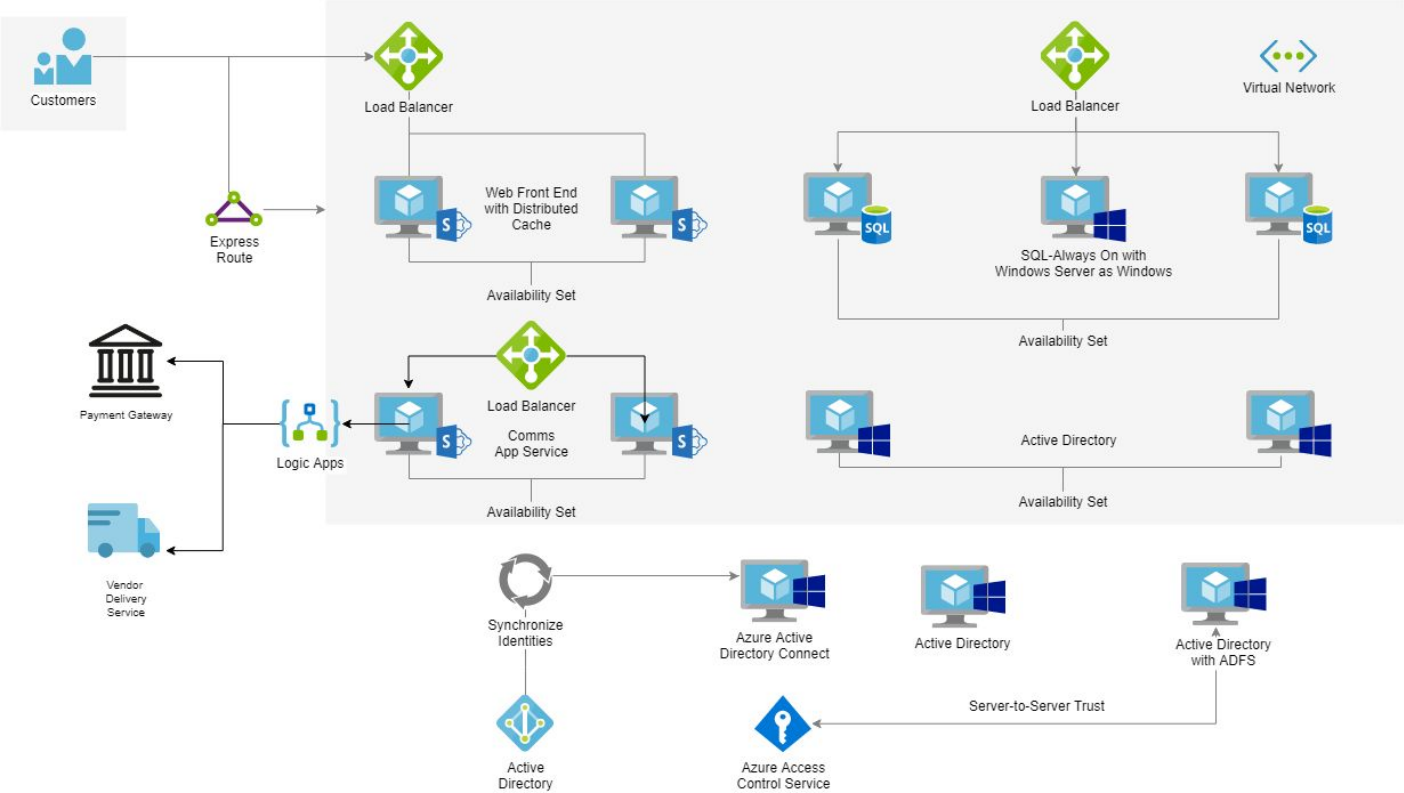
| Service | Upfront | Monthly | First 12 months to | Currency | Configuration summary |
|---|---|---|---|---|---|
| Amazon Aurora MySQL-Compatible | 0 | 424.8 | 5097.6 | USD | Change records per statement (0.38), (1 instances) db.r5.xlarge Memory optimized OnDemand, Storage amount (10 GB), Number of I/Os per month in millions (2) |
| Amazon EC2 | 0 | 87.8 | 1053.6 | USD | Operating system (Linux), Quantity (4), Storage for each EC2 instance (General Purpose SSD (gp2)), Storage amount (10 GB), Instance type (t2.medium) |
| Amazon Route 53 | 0 | 173.2 | 2078.4 | USD | Hosted Zones (3), Basic Checks Within AWS (5) |
| S3 Standard | 0 | 0.28 | 3.36 | USD | S3 Standard storage (10 GB per month) |
| Data Transfer | 0 | 205.61 | 2467.32 | USD | Data Transfer, Data transfer cost (205.61) |
| Amazon Virtual Private Cloud (VPC) | 0 | 109.5 | 1314 | USD | Number of Site-to-Site VPN Connections (3) |
| Amazon Virtual Private Cloud (VPC) | 0 | 467.1 | 5605.2 | USD | Number of NAT Gateways (6) |
| Amazon Elastic IP | 0 | 22.2 | 266.4 | USD | Number of EC2 instances (6), Number of EIPs per instance (1), Number of EIP remaps (3) |
| Amazon ElastiCache | 0 | 998.64 | 11983.68 | USD | (6 instances) Redis Memory optimized cache r4.large OnDemand |
| Amazon CloudWatch | 0 | 14.4644 | 173.57 | USD | |
| Amazon Simple Email Service (SES) | 0 | 2.92 | 35.04 | USD | Email messages sent from EC2 (5000 per month), Email messages sent from email client (5000 per month) |
| **Total** | | | **30078.17** | | |

# Additional Diagram Examples

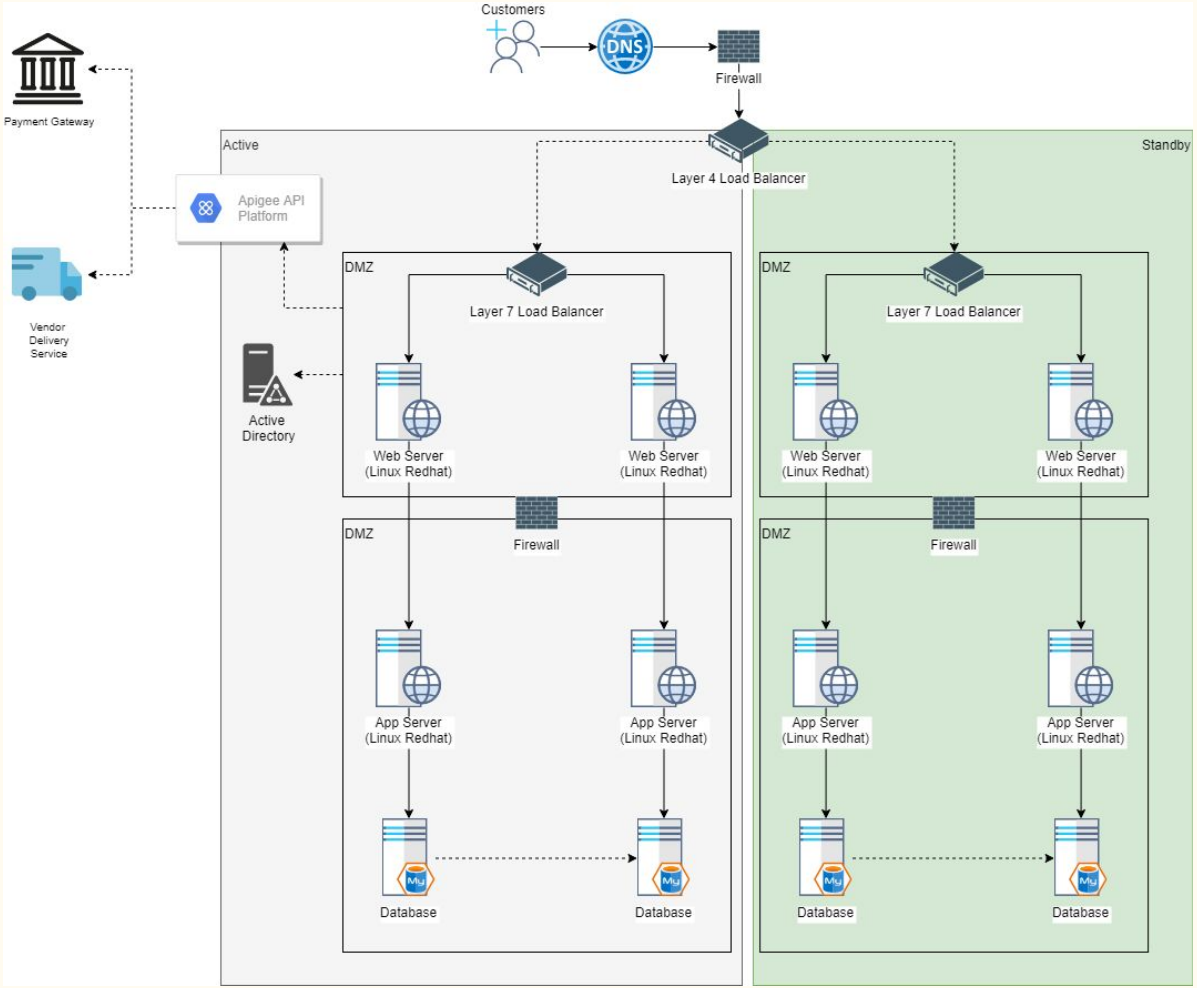
# BMPN - Business Process



# Azure Solution Architecture Diagram



# On-Premise Network Infrastructure



# Security Architecture Sample