IT Architecture Vision / Sprint Zero Architecture

The Funny Hat Shop - Online

Problem Description

Problem Statement

The Funny Hat Shop wants to sell their existing hats in stock online. During COVID-19, the company realised they are not making sales as they used to and believe an online e-commerce web site will restore their sales by selling their hats online.

Stakeholder	**Business Capability/Function**	**Business Problem/Concerns**
Web site	Sale	Sell existing stock of hats online
Customer	Basket	Customer register online before or after adding items to the basket
Warehouse Manager	Stock Management	Do not sell items not in stock – check stock system for stock levels
Customer	Account	Secure login – own username and password, or google or Facebook
Customer	Payment	On checkout basket –use secure payment gateway for payment (PayPal or Credit Card)
Customer	Communications Order	Send automatic emails when order is completed successfully to customers
Warehouse Manager	Stock Management Order	Send order to warehouse manager to pick items as per order
Delivery Company	Deliveries	Integrate solution with delivery company to request a pick up and delivery request
Customer	Communications Invoicing	On successful delivery, send automatic invoice email

Business Vision Statement

The Funny Hat Shop vision is to be able to sell all their stock they also sell in the shop on the internet. The solution should include cost effective options to host the e-commerce web site.

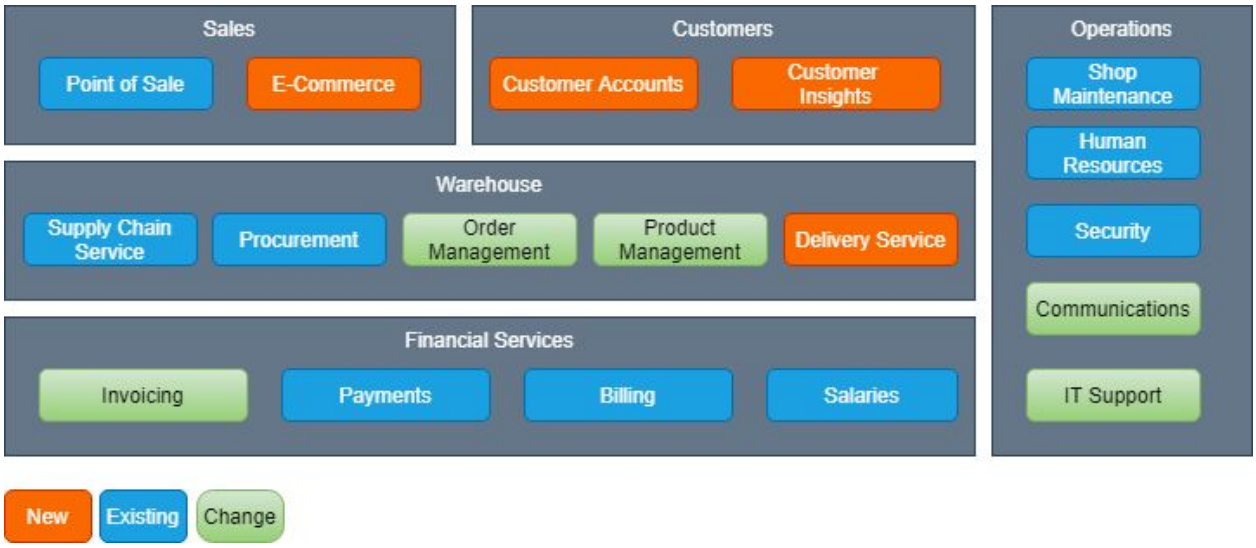
Change Drivers & Opportunities

Here is a list of key business drivers and opportunities to implement the e-commerce web site for The Funny Hat Shop:

- 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.
- The solution should integrate with the existing stock management system used in the warehouse.
- The web site can only be accessed from America, Europe and South Africa.
- The web site's availability should support a 24/7 uptime
- Consider responsiveness and develop using agile methodologies
- Privacy of customer data should be handle as confidential and security should be implemented

Business Objectives / Capability Impact

Figure 1 - Business Capability Model



Capability	**Description**	**New/Migrate/Change**
E-Commerce	Web site to sell hat products	New
Customer Accounts	Store and manage customer information	New
Customer Insight	Basic reporting of customer purchase history	New
Order Management	Order management will change due to orders being purchased online.	Change
Product Management	Products now have to also include images of the products	Change
Delivery Service	Integrate with an external delivery vendor solution and manage status of deliveries	New
Invoicing	Invoices will now be delivered automatically to customer email on successful delivery	Change
Communication	Include a email service to send automatic emails to customers	New
IT Support	IT support will include the support of the new e-commerce web site.	Change

Architecture Vision

To provide an e-commerce web application to support the sales of hat products for The Funny Hat Shop. The solution architecture should implement the solution using the following architecture principles:

- Business continuity - ensure the system has a disaster recovery plan included
- Ease of use - Keep the technology selection simple and easy to adapt and use
- Data security - Ensure the data is secure at rest and in transit
- Technology independence - the design should include application components that are independent of the infrastructure it runs on

Architecture Assumptions

The following architecture assumptions is made based on the drivers and objectives of the business problem:

- The stock management system is hosted on Amazon AWS with an API capability to access and update stock levels
- The delivery Vendor has an API capability to send and get delivery status information

Constraints & Risks

Find a list of constraints relating to the initial business requirements:

- The web site can only be accessed within America, Europe and South Africa

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

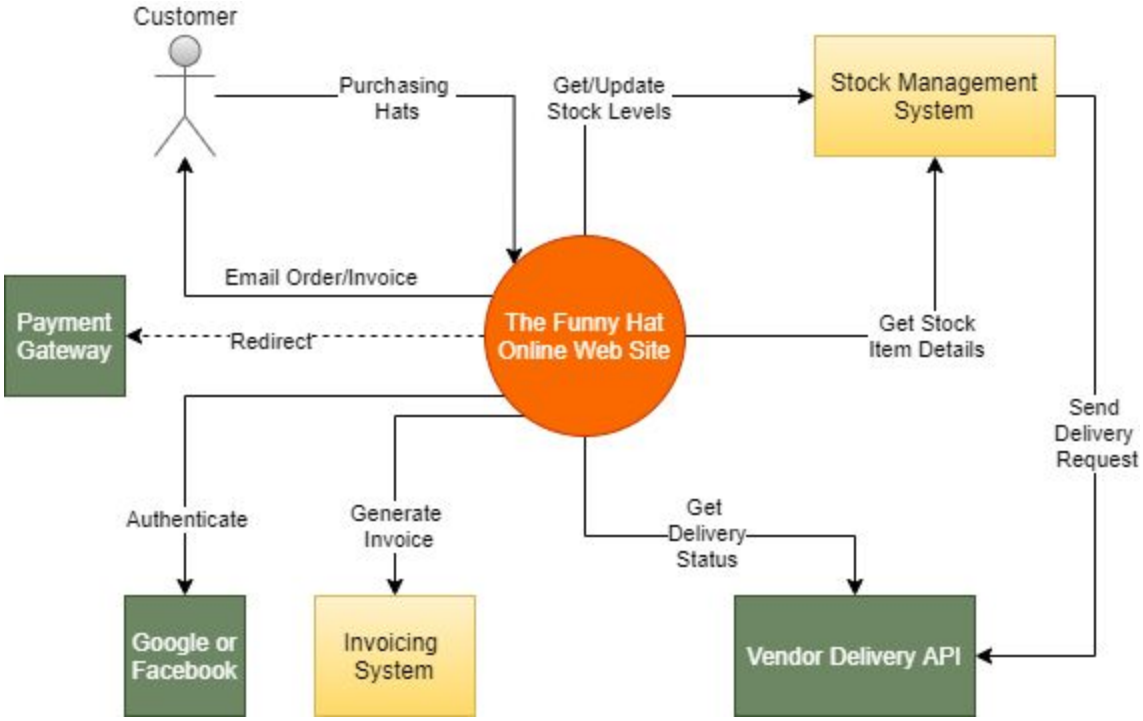
As-is Conceptual Architecture Diagram

There is no as-is architecture as this is a new solution integrating with existing systems.

To-Be Conceptual Architecture Diagram

Context Diagram

Figure 2 - Context Diagram



The above context diagram depicts the interaction between the different systems and actors.

The following interactions is expected and applications impacted:

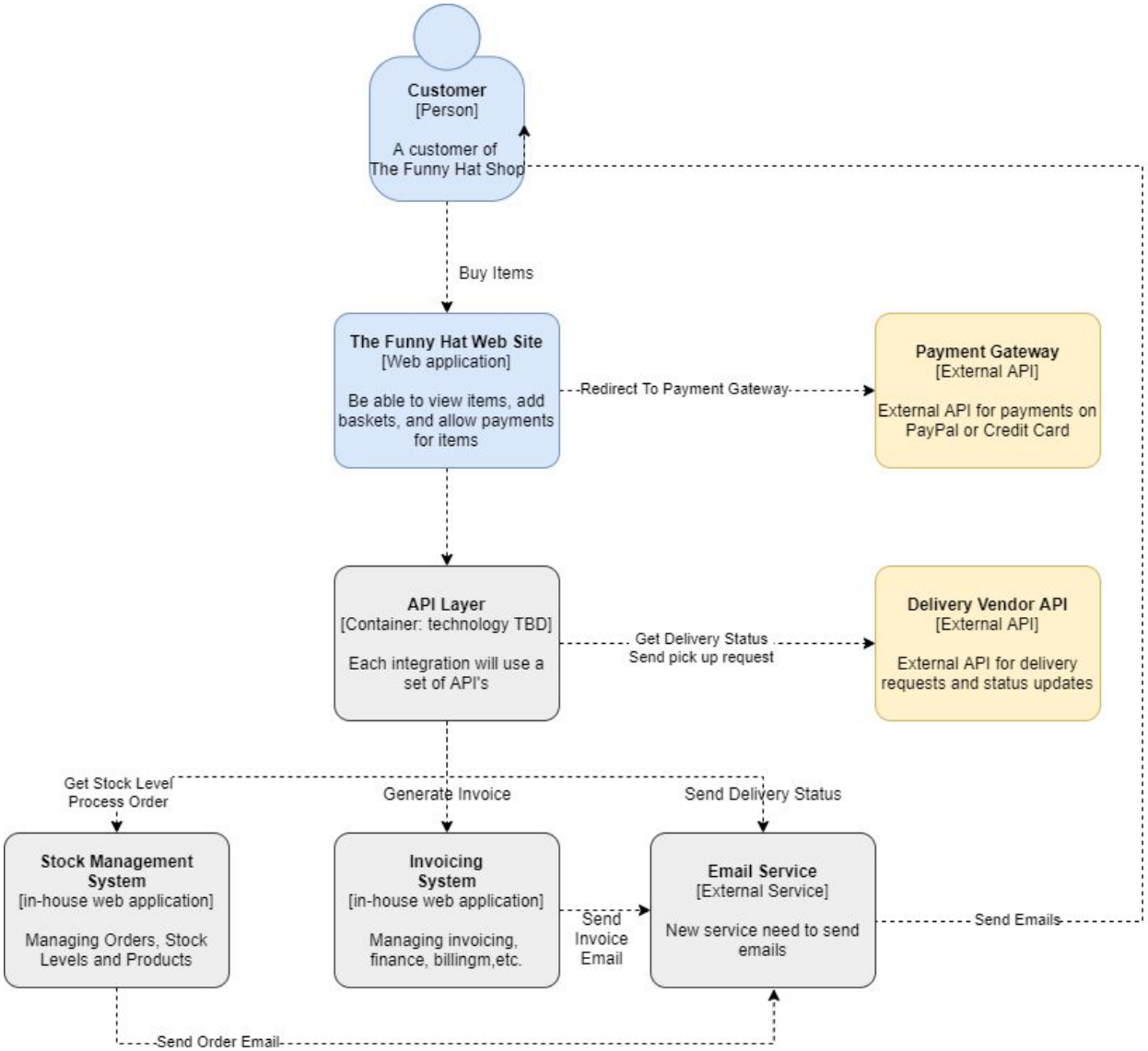
- Customers will interact with the online web site by adding items to a basket, complete a checkout process and pay for the items using a secure payment gateway
- The online web site will interact with the stock management system for the following interactions:
    - Get Stock Item Details to be displayed on the web site. This should include an image
    - Get Stock Level for a specific item
    - The stock management should update the stock level when an order is complete
- The stock management system will send a pick up and delivery request to the Vendor Delivery company's API
- There should be a scheduled job to check the delivery status and update the online web site when the delivery is done successfully.
- An email with order details should be generated when the order is generated on the stock management system

- An email with invoice details should be generated when the order is delivered to the customer by checking the Vendor Delivery Company's API for delivery status completed

C4 System Context Diagram

This is to indicate a different style of modelling architecture. You can either use the context diagram technique or C4 Systems Context Diagram.

Figure 4 - C4 Systems Context Diagram



High-Level Non-Functional Requirements

Availability

- Solution should be highly available 24/7 with 99.99% availability
- Batches are expected to run to check delivery status
- Planned downtime for Monday mornings 1 to 2 hours only at 1am.

Performance

- Solution should allowed for auto scalability on evenings and weekends
- The web site response time should less than 2 seconds
- Allow for a thousand hit per day

Volumes

- Product images are stored on AWS S3 Buckets for the stock management. Images are about 50kb average
- The Funny Hat Shop has 200 different products
- Allow for an initial 100gb of data storage with a 10% growth expected each year

User Interactions

- The company expect 500 customer for the first 3 months, and 2000 after 12 months
- Allow for at least 10 concurrent users accessing the web site
- User locations should only be from America, Europe and South Africa

Business Continuity

- Include backups with replication of the database
- Use multi availability zones in each region
- Customer data from that each country should be stored only in availability zones within those country's region

Security

- Authorize and authenticate user by using OAuth, using either Google or Facebook logins
- Secure data at rest and in transit on all services
- Implement audit controls on data changes
- Use X509 Certificate on the Web Application Container

Operations and Monitoring

- Implement monitoring for both the health of the web site and database
- The solution should include exception handling and alert on any errors

Networking

- Create a VPC on AWS for all the regions of the solution
- Create a public subnet for the web front end and private subnets for the back end database and other services
- Use a load balancer to balance the incoming traffic to the web application
- Create firewall rules for traffic to the Delivery Vendor API

User Interface Requirements

- Implement a responsive web design that will be accessed via mobile, tablet or web browsers

Architectural Requirements

- Use AWS Cloud platform
- Implement a Dev, QA, UAT and Production VPC on AWS
- Detail of the development stack will be in the solution options
- Solution should follow a Microservices architecture by implementing the solution components as containers
- Use DevOps pipelines to cater for automated deployments for the Continuous Integration and Continuous Deployment pipelines.

Proposed Solution Option

- Use EC2 to host the web application using apache as a web server
- Create application layer on EC2 for business rules and access to the data layer
- The auto scaling groups should be used to automatically scale in and out depending on the incoming load from the load balancer
- The application load balancer
- For caching use Elastic Cache to cache product data
- Images will be stored on the stock management S3 bucket
- The domain will be hosted through Route 53 Hosted Zone
- Data will be stored on MySQL on Amazon RDS
- AWS Cloudwatch will send alarms for any health issues on the EC2 instances

- Amazon SES will be used to send emails to customers
- Amazon CloudFront will be used for content delivery
- Amazon API Gateway to access Vendor Delivery API's

Costing Estimate

T-Shirt Sizing Exercise

Size	Estimate Work Range
S	0-2 weeks
M	2-4 weeks
L	4-8 weeks
XL	8-16 weeks

Building Block / Epics	Description	T-Shirt Size	Weeks
Web Application Front End	Building the web application with functionality to list products, add items to a basket	XL	10
Database	Create a database that will store basket items, customer account data and web site sales	M	2
Emails Service	Create an email service using Amazon SES to send order and invoice emails	M	2
Orders	Integrate with the Stock Management System to generate an order	S	1
Invoices	Integrate with the Invoicing System to generate an invoice when the order is delivered	S	1
Deliveries	Integrate with the Vendor Delivery API's to request pickup and delivery. Include API calls to get delivery status	L	4
Security	Build authentication for customers to log into the web site and create an account. Include the option to use facebook and Google logins	S	1
Infrastructure	Setup the AWS VPC, subnets, EC2 instances,	M	3

	and all other Cloud requirements, including the Dev, QA, UAT and Production environments		
Payments	Integrate with a payment gateway to allow customers to pay for the items in the basket	M	2
	Total		**24**

An estimate of 24 weeks is made by a collaborative session by the team. This is based on a team effort of 2 developers, 1 business analyst, the solution architect, QA tester and scrum master. This effort is not done by the architect but usually by the project manager or scrum master. The total team value per hour is $180 per hour. If the team works a 40 hour week for 24 weeks, the total amounts to 960 hours, and total amount to $172,800.00.

Cost Category	**QTY**	**Money Value ($)**
Development Cost	960	$172,800.00
Vendor Software (Integration)	1	$ 1,500.00
Infrastructure (AWS Cost)	1	$ 50,000.00
Any recurring costs for 12 months. E.g. Annual Fees, software licences, rentals, etc.	1	$ 50,000.00
Totals:		**$274,300.00**