**Commerce Bank Web App**

**Architecture Document**

**Team AAFK – Winter 2022**

**Table of Contents**

[***1. Introduction 1***](#_heading=h.gjdgxs)

[***2. Implemented Architecture (N-Layered) 1***](#_heading=h.30j0zll)

[**2.1 Model View Controller Architecture (MVC) 2**](#_heading=h.1fob9te)

[**2.4 ERD Architecture Description 3**](#_heading=h.3znysh7)

[**2.5 Azure Web Service Architecture 4**](#_heading=h.2et92p0)

[**2.6 Azure Architecture Description 4**](#_heading=h.tyjcwt)

[***3. Layer Classification 5***](#_heading=h.3dy6vkm)

[**3.1 MVC Layer 5**](#_heading=h.1t3h5sf)

[**3.2 Database Layer 5**](#_heading=h.4d34og8)

[**3.3 Web Layer 5**](#_heading=h.2s8eyo1)

# **1. Introduction**

This document describes the architecture and Design for our Commerce Bank Fundraiser page which was developed by Team AAFK and deployed live on Azure's web services. Our website Life’s Lemonade is a web application written in C#, HTML and CSS and deployed on Azure’s web service. This application allows users to create a fundraiser relatively easily as they are not required to have any technical skill. The web application is focused on simplicity and easy to use for the user.

# **2. Implemented Architecture (N-Layered)**

## **2.1 Model View Controller Architecture (MVC)**

Graphical user interface, application, timeline

Description automatically generated

**2.2 MVC Architecture Description**

The architecture used when developing our web application is the model view controller architecture used with .Net Core and its Entity Framework. This allows us to separate each component of our application into separate parts, which allows for better scalability and testing of our application. It will enable us to separate the UI interface from the data and the logic. Following this within our app allows for easier maintainability of our web application. This allowed us to divide each functionality and the database to ensure safety as each component is separate and not intertwined, keeping sensitive data safe.

**2.3 Entity Relationship Diagram (ERD)**

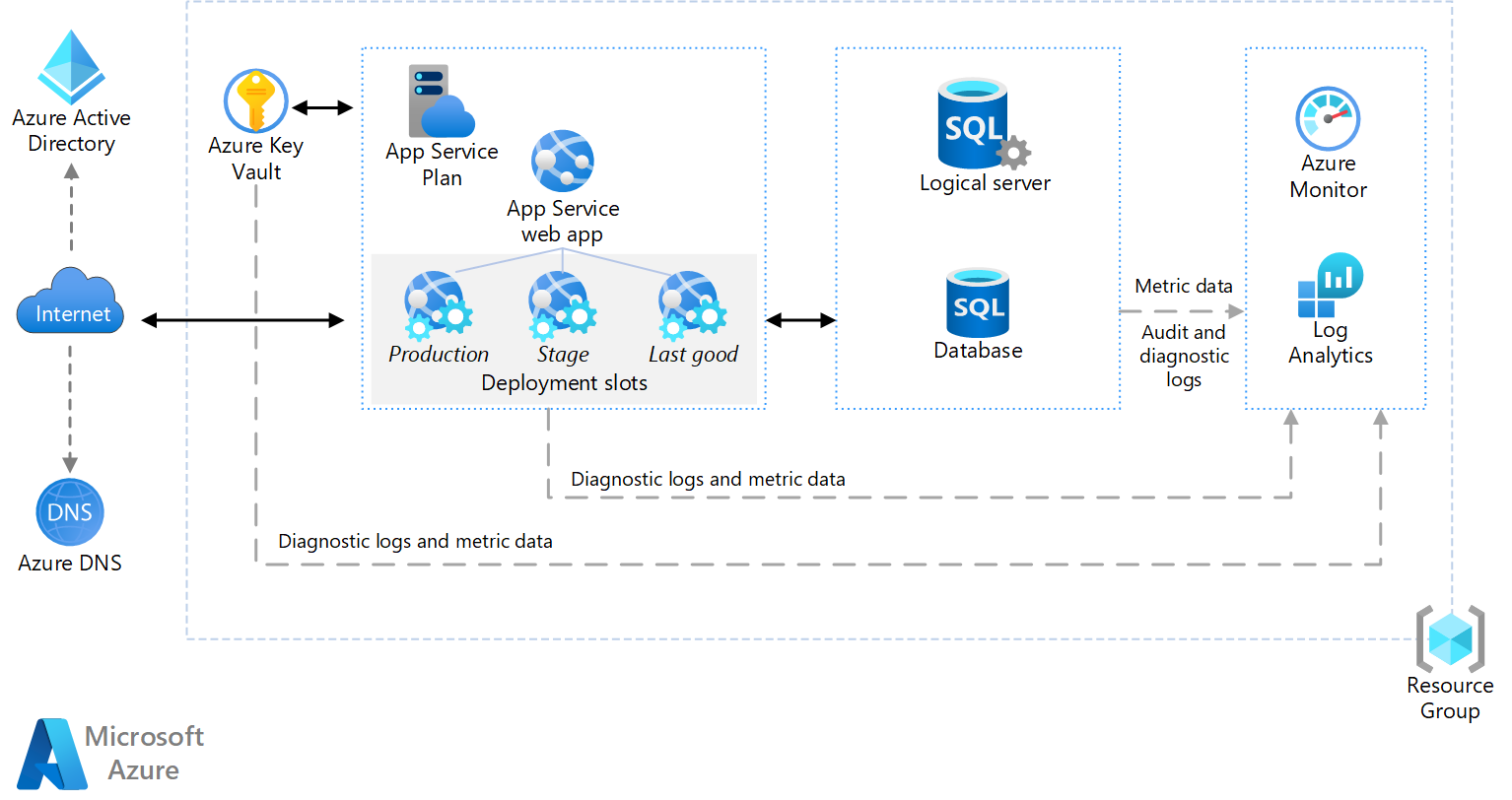
**A picture containing graphical user interface

Description automatically generated**

## **2.4 ERD Architecture Description**

The ERD shows the way in which the database is connected and how it connects to each component. For example, we keep track of the users and their donations by assigning them unique IDs and keep all data within the database. Most of the tables seen in the ERD are from the entity framework which keeps track of the users and their info like logins and payment info related to each user. The Database works in conjunction with the MVC to display the data to the user but the user has no direct access to the database this allows some safety functionality and security as the user cannot access the sensitive data.

## **2.5 Azure Web Service Architecture**



## **2.6 Azure Architecture Description**

Azure is where our web application was deployed on. The image above demonstrates the way the web application is taken into their service and bundled with the database we created alongside the MVC and is all stored within a resource group which can then deploy a website that is accessible through azure’s services and is ready to view on the internet by accessing Azure’s DNS and directory. Our MVC is what allows everything to function between the database and the UI as well as the brains or logic of our whole web application. Here we can see that the overall architecture implemented within the web application is that of a N-Layered architecture as it builds on the previous Architecture

# **3. Layer Classification**

## **3.1 MVC Layer**

**Purpose:**

This layer is used to separate functionality of all the components. The user interface (UI) is separate from the Controller or logic of the web application. The Database is also separate as the user does not have direct access to it.

**Specific Nature:**

This layer oversees taking the data inputted by the user and gives it to the database and then presents that data or info to the user accordingly. The MVC is also the way the application is structured as each controller oversees some individual component.

## **3.2 Database Layer**

**Purpose:**

This layer is in charge of storing data in a safe storage space as it can update and delete data accordingly. Keeps track of users and their fundraisers as well as any payments made.

**Specific Nature:**

This layer consists of storing all information about fundraisers created by each user as well as any donations they do. The database also allows for simple (CRUD) create, read, update, and delete functionality for each fundraiser.

## **3.3 Web Layer**

**Purpose:**

This layer oversees having a live web application that is connected to a database within a server which allows for the web application to be easily deployed online and be viewed anywhere by anyone.

**Specific Nature:**

This layer consists of keeping the application live as well as the database live by keeping them in a resource group within Azure and can be checked for any performance issues or any error that may appear through a log system and allows to check on users’ interactions. Can also be built upon as it allows for other frameworks and APIs to be used in conjunction to the web application.