**No Kill Inventory Software Requirements**

**Capstone Fall 2023**

**Introduction**

**Purpose**

The inventory system for No Kill Louisville is meant to address a problem: the lack of an established inventory system or rules. Currently, food is loosely sorted into categories and there is out of date information on the status and amount of inventory. We are aiming to create a web app for No Kill that allows them to streamline their inventory input & output process, and have accurate inventory records.

**Definitions**

No Kill Inventory System – Project name

IUS – Indiana University Southeast

No Kill Louisville – Pet food bank and project owner

C# - Microsoft .NET programming language

HTML - Frontend markup language

CSS – Frontend style language

Blazor – Microsoft .NET web framework that combines C#, HTML, and CSS

Entity Framework Core (EF Core) – Object-relational mapping framework for .NET

.NET Framework – Software framework created by Microsoft

SQLite – Relational database management system

**System Overview**

The program will consist of several pages a user can navigate to and perform various functions on as part of the frontend, along with a backend which will consist of a database and an authentication process. On the frontend, the user will have the ability to login, fill out a form to input new inventory items, and view & edit the current inventory records.

**Overall Description**

**Product Perspective**

Currently, a domain name and hosting service have not been chosen. It is being discussed with No Kill to find the best possible option based on cost and the needs for the project.

All the backend is written in C#, the frontend in HTML and CSS, and the database in SQLite. The project is intended to be lightweight, user friendly, and cost-effective. There will be no actual hardware to be maintained once a hosting service has been chosen, which will be more cost-effective for the organization.

**Product Functions**

The web app will take input from a user on a form and input the information provided into the database. A user will also be able to view the records in the database and see the current inventory in the system.

**Constraints, Assumptions, Dependencies**

The main constraint with the project is the cost of hosting and running the web app. The cost should be nominal, but it is dependent on what service is chosen to host and run the inventory system. It should not be difficult to migrate to other services in the event it is desired or needed in the future.

The largest dependency will be the hosting service, as all the frameworks and languages being used are either open-source and/or still being maintained and updated.

**Specific Requirements**

**External Interface, Functional, Performance, and Database**

There should be no need for external interfaces needed such as API’s. The only potential external interface would be authentication of users so that the inventory system is not available to the public. Authentication might be handled by a service provided by the hosting service or implemented in the web app itself.

**Software System Attributes**

The system itself will be hosted and ran by a hosting service, so maintenance and reliability will be on them, and not No Kill. Unless the problem is with the web app and its functionality itself. The main concern is setting up the infrastructure properly to ensure there are little to no issues once it is deployed for production use.