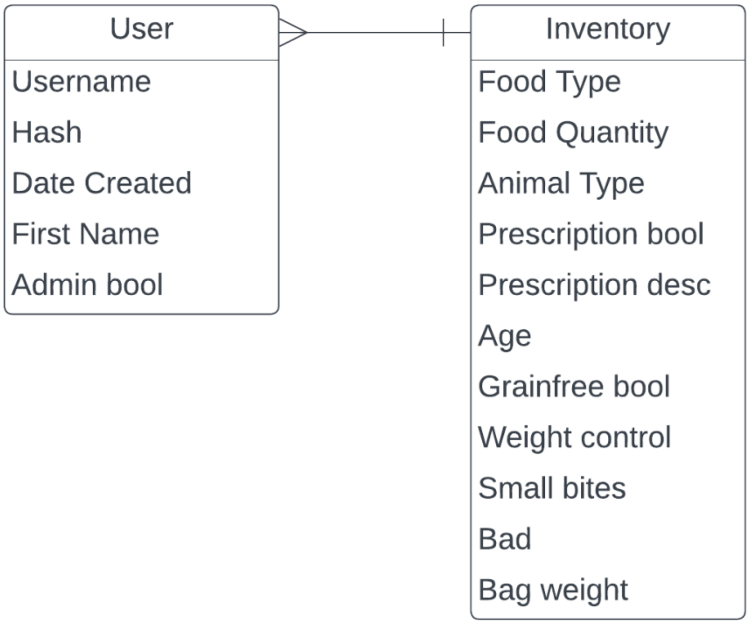
No Kill Louisville Inventory System

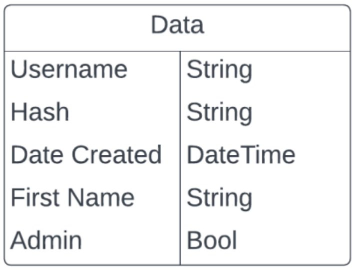
Software Detailed Design

P445 – Fall 2023

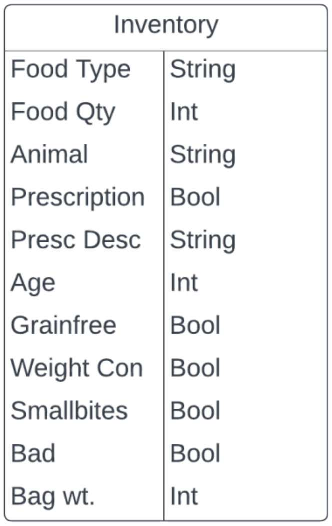
1. Data Structures And Objects



These data structures are really all that we need for our project. The user data is very limited as we really don't need to personally know the user's passwords so instead, we acquire said password and immediately hash it allowing passwords to stay anonymous to us. The inventory is what needs the most data and tables in the data base as there are just so many variables, given that a lot of them are bools. The database tables would look a little like this:



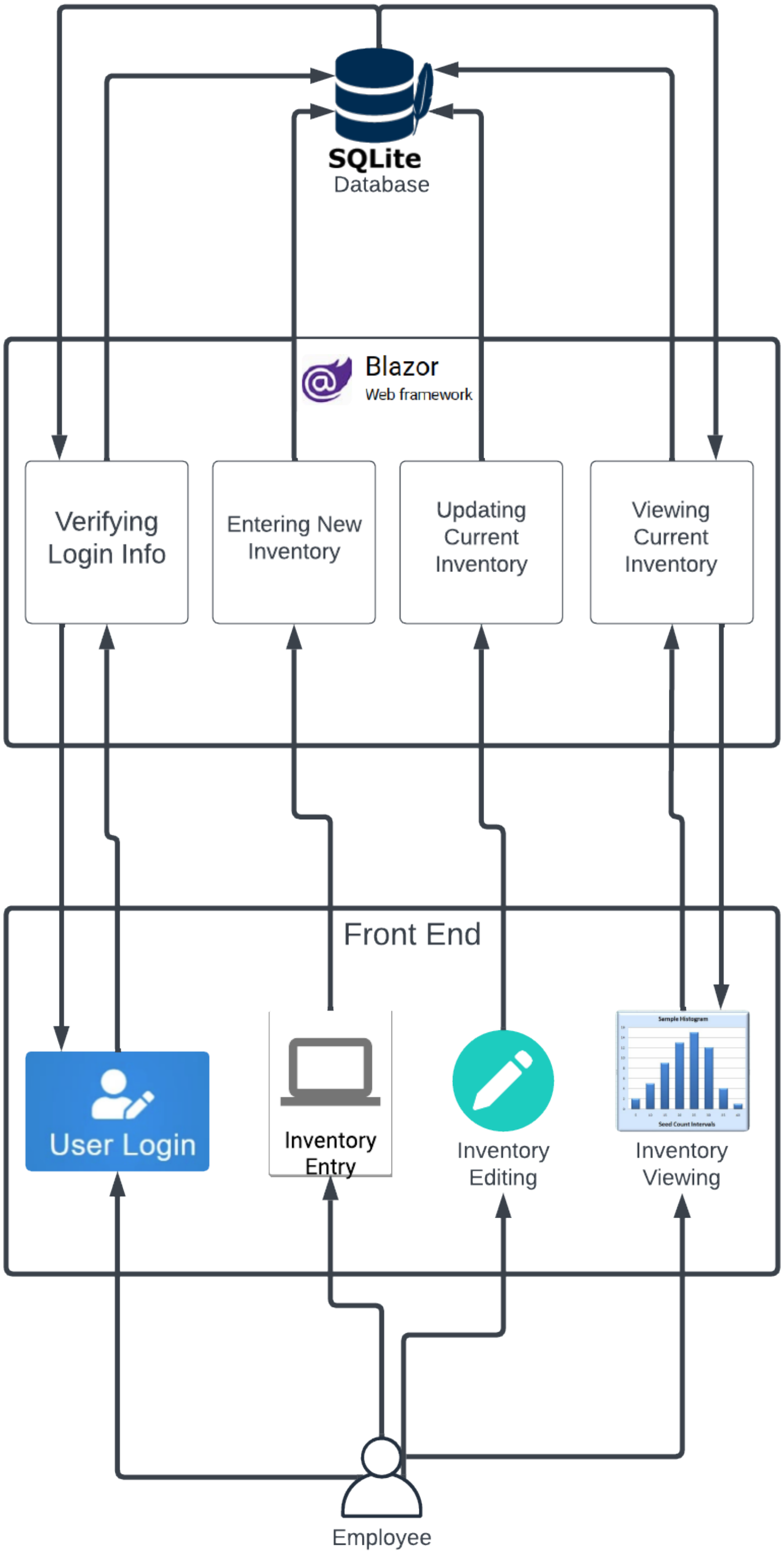
Data	
Username	String
Hash	String
Date Created	DateTime
First Name	String
Admin	Bool



Inventory	
Food Type	String
Food Qty	Int
Animal	String
Prescription	Bool
Presc Desc	String
Age	Int
Grainfree	Bool
Weight Con	Bool
Smallbites	Bool
Bad	Bool
Bag wt.	Int

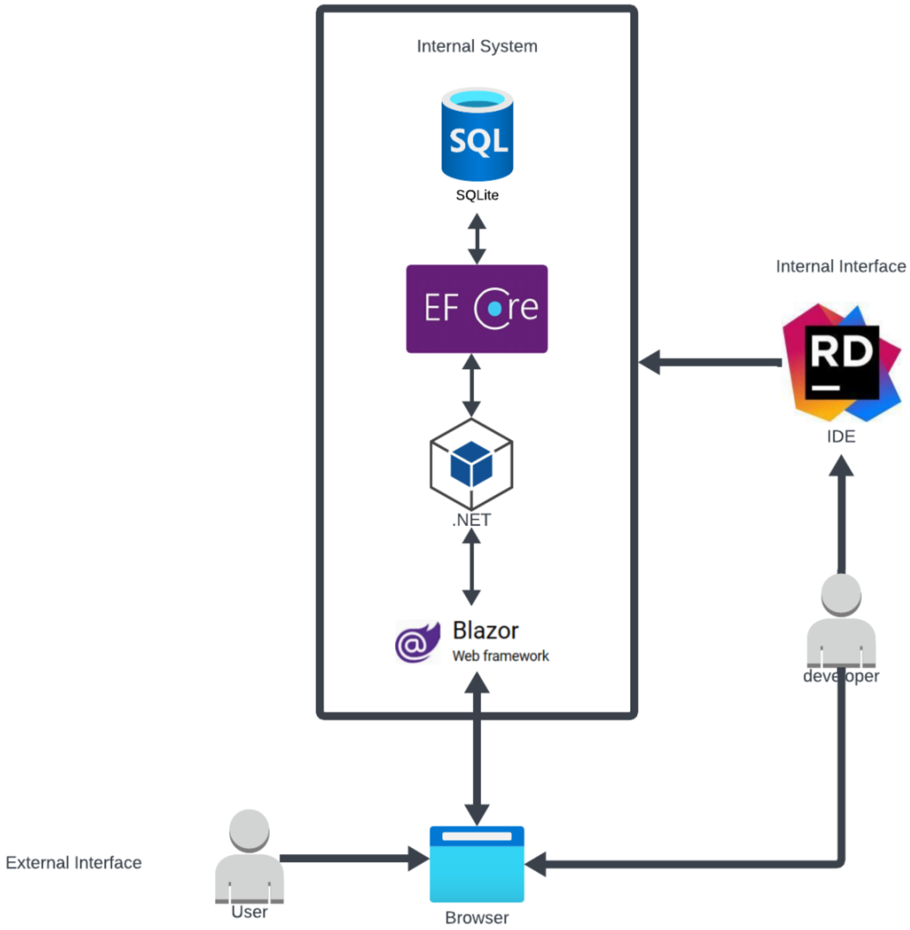
2. Architecture Design

This system's architecture is currently split into four parts, two of which connect to and from the database while the other two send to it. These include user login which goes to and from the database, inventory entry that just goes to, inventory editing which just goes to, and inventory viewing which goes to and from the database. This, of course, is all controlled form the employee through Blazor to access the back end.


SQLite
Database
Blazor
Web framework
Verifying Login Info
Entering New Inventory
Updating Current Inventory
Viewing Current Inventory
Front End
User Login
Inventory Entry
Inventory Editing
Inventory Viewing
Sample Histogram
Seed Count Intervals
Employee

Employee	Does not necessarily need to be employee just user in general only gets access to the front end as it should be.
Front End	The front end only contains the actual input elements and does not directly see backend info but some of them do return database values such as the user login, and inventory viewing.
Back End	The back end only gets connections from the front end but only actually connects to the front end from two spots, the user login and inventory viewing needs access to the backend enable to support the front end.

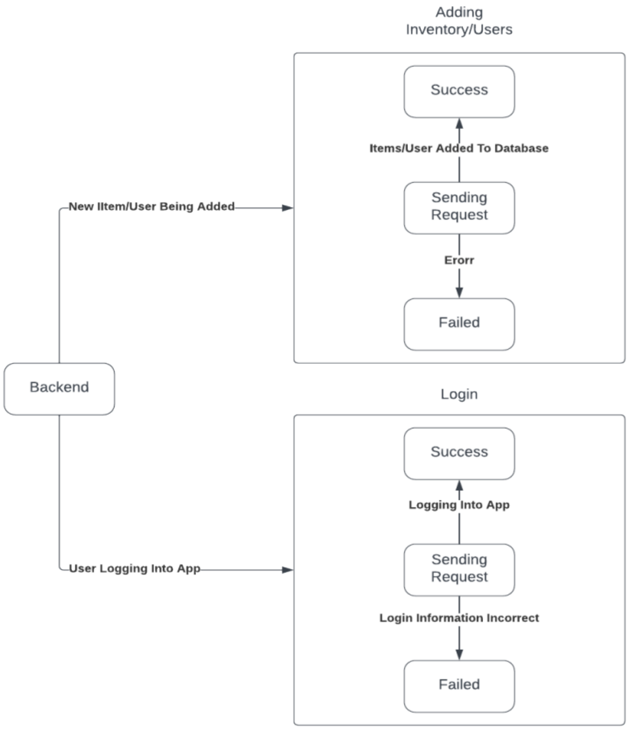
3. Interface Design



Our diagram shows how the developers will be accessing the internal system and the tools they will be using to do so along with how the user, or No Kill Employee, accesses it. The employees, or users, will be using a web browser to access the internal system where it will store the data of their logins and inventory counts. The developers will be accessing it by both the Rider IDE and the browser. The Rider IDE is used to develop, run, and debug the software. The browser will be used to test the user view to ensure everything looks and is running as we expect.

For the internal system we are using Blazor to create the web application. Blazor complies them to .Net and so that the application can run on Windows. EF Core will be used to interact with the database including storing, editing and viewing the data and SQLite is the database we are using to store the inventory count and login information of the users.

4. Procedural Design



The program only has 2 main procedures: inputting new items and new users to the database and logging in to the website. All other procedures are extensions of or supporting the two main procedures.

Logging in, adding users, and inputting new items are done with the front end and then the back end validates and sends all the data that has been input to the database. Error handling is also a procedure, as the program does need to respond to incorrect login data, or invalid inventory data.

5. Individual Responsibilities

Jacob Riggs – Back-End Design

Elizabeth Mittel – Front-End Design

Nicholas Gathof – Database Design

6. Key Personnel

Jeremy – Representative from No Kill Louisville

John Doyle – Professor working with No Kill Louisville

Ronald Finkbine – Class professor