

# Black Bear Foodshare



## System Design Document

Olive Food Solutions

Dr. Scott Marzilli

Team Members: Dumas Wesley, Kaulenas Corey, Moody-Broen Makai, Sima Denis, Sholler  
Jakob

Black Bear Foodshare  
System Design Document

**Table of Contents**

1	Introduction.....	2
1.1	Purpose of This Document.....	3
1.2	References.....	3
2	System Architecture.....	3
2.1	Architectural Design.....	3
2.2	Decomposition Description.....	4
3	Persistent Data Design.....	5
3.1	Database Descriptions (if you use a database).....	5
3.2	File Descriptions.....	5
4	Requirements Matrix.....	6
	Appendix A – Agreement Between Customer and Contractor.....	7
	Appendix B – Team Review Sign-off.....	8
	Appendix C – Document Contributions.....	9


# 1 Introduction

This is a capstone project for Dr. T Scott Marzilli, Associate Provost of Student Success & Innovation, in partial fulfillment of the computer science BS degree for the University of Maine. Dr. Marzilli places great emphasis on the importance of providing the necessary resources and opportunities for students to succeed at the University of Maine. The impetus of this project comes from the passion Dr. Marzilli and our team share in providing judgement free opportunities for students dealing with food insecurity, as well as furthering the elimination of food waste on our campus. The purpose of this project is to develop an application to allow hosts of catered events to easily inform students of excess food on campus. These posts will include the food, photos of the food, and the location. The goal of this project is to reduce waste and increase sustainability on campus, while also complying with regulations to make sure the excess food is safe to consume.

## 1.1 Purpose of This Document

The purpose of this document is to inform our team, client, and interested parties of the intended design and functionality of the Black Bear Foodshare application.

## 1.2 References

- Olive Food Solutions, System Requirements Specification  
 System Requirements Specification

# 2 System Architecture

This section of the System Design Document contains section 2.1 Architectural Design, and section 2.2, Decomposition Description. This section outlines the necessary technologies that will be implemented in the design of the Foodshare application. Section 2.2 specifies the functions and methods that will be used to develop the application.

## 2.1 Architectural Design

Our system shall operate as an IOS only mobile application. It will be developed in XCode, using the proprietary IOS object oriented development language Swift. For the backend we will be using Firestore as our development tool. The backend will be programmed with Python as our language of choice. Digital Ocean will be the hosting service for the corresponding webserver, which will be programmed with JavaScript. Docker will be used as the container service for the application. Depending on time and technological constraint, an Android service may be developed alongside the IOS service. Due to the functionality of cross

platform development, our app would have to be simultaneously and separately programmed in Kotlin and Swift, because of our time constraints we are prioritizing IOS development. If the aforementioned constraints permit Android development, the Android application will be programmed with Kotlin.

- Frontend - Swift

The frontend is responsible for displaying all of the visual portions of the application. Such as food posts, surveys, and account information. If possible the user interface will mimic the current Umaine theme/aesthetic. The frontend will also be communicating with the backend to manage user actions and data flow. If the user clicks on the map details it will send info to the mapping client. If the user clicks for a post, checks an account etc it will communicate with the backend.

- Backend - Python Firestore

Utilizing Firestore for its high reliability, easy scalability and speed. Our backend needs scalability for future growth, speed for handling the logic behind the various subsections of code, (Uploading Posts, Viewing Posts, updating account settings, etc.).

- Database - Firestore

Firestore was chosen for the database due to its scalability and affordability, with support of 50k monthly active users and 1gb of storage. It covers more than enough to cover more potential users than planned. With the minimal data that is being saved, 1gb of data is enough, it does also support Google pricing for cloud storage if more is needed. Data such as user account information, Survey Information, While foodshares and their photos will be stored, they will only be temporary. Being removed after time to save on space and costs. Surveys and Foodshare data stored will be utilized for the improvement of the application.

- Server - Digital Ocean

Digital Ocean is what our system uses to host the webserver for the project.

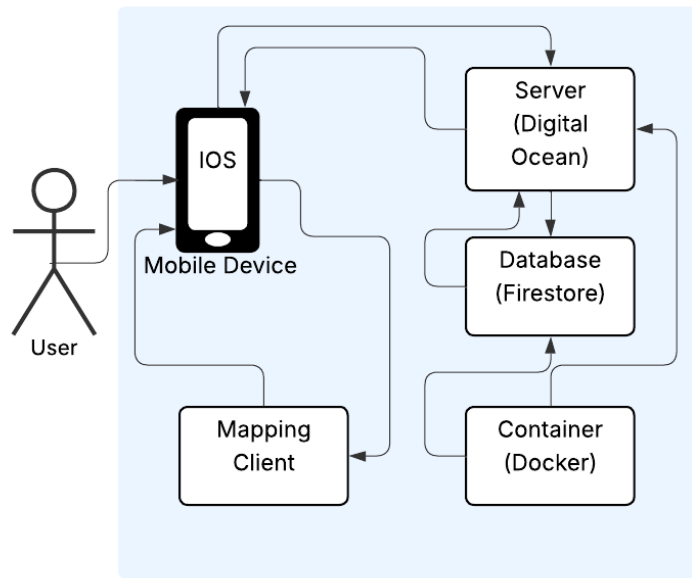
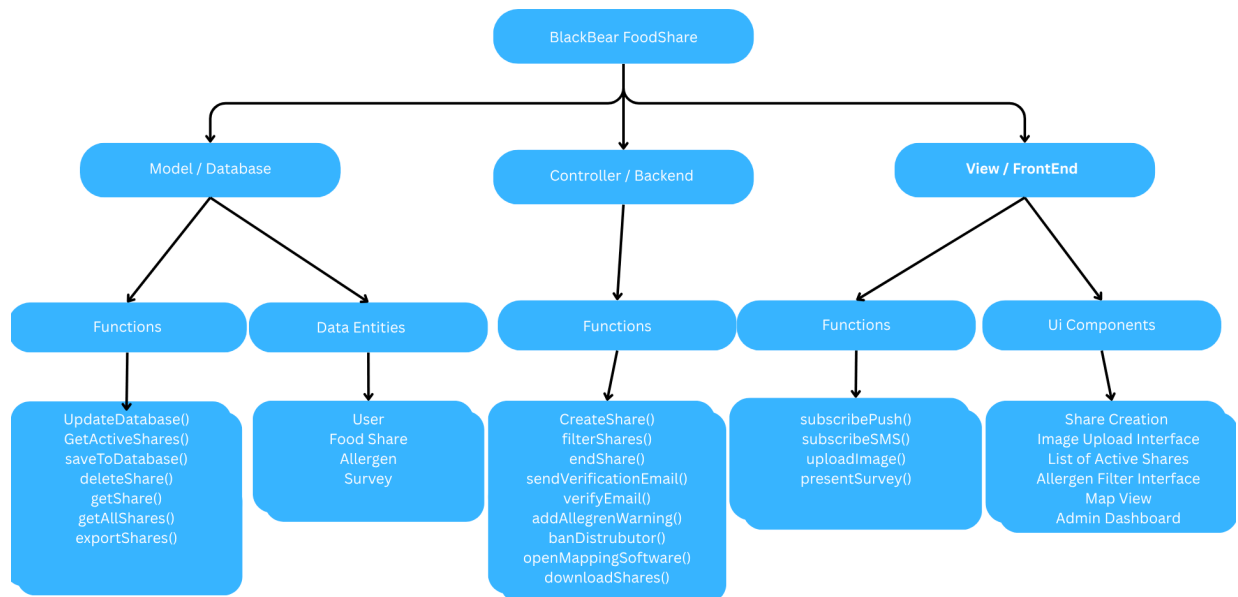


Fig 1. Technology Architecture Diagram

## 2.2 Decomposition Description



Our System follows an object-oriented design and utilizes the Model-View-Controller pattern. The pattern is split into three layers.

- Model Layer - Database; utilizing firestore.

The model layer is our database; it is where data will be stored, and any functions will be needed to modify or view the data from the database.

The data entities are the different types of data that will be in the backend, there

will be user data such as accounts, foodshare data, which will be the posts sharing food. Allergen data which is for safety, and Survey data which will be used for analytics

- View Layer - Frontend; utilizing swift for IOS

The View layer is the front-end visuals where the user will be. Anything that needs to be visualized will be in the view layer such as Ui Components. There will also be functions that reflect user actions such as signing up for notifications, or uploading images for food shares.

- Controller Layer - Backend; utilizing python and firestore.

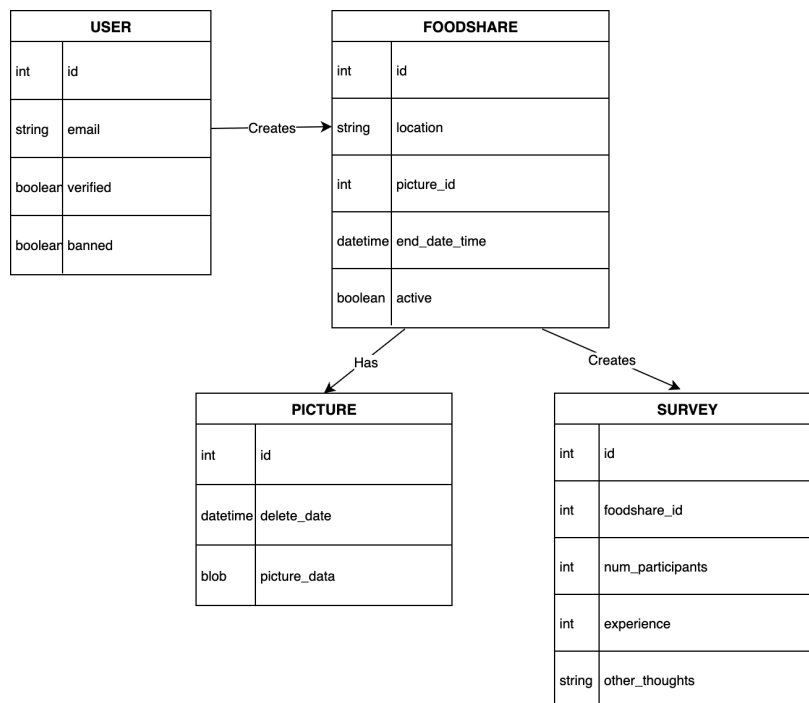
The Controller layer handles the data processing between the view layer which is where the user will be, and the model layer or database. Any data processing such as filtering food shares, email verification, creating food shares, opening a map for finding the food share, and adding allergens to a food share are all things the controller layer handles.

### 3 Persistent Data Design

This section shall outline the hierarchical relationships between our functions and fields. Necessary file descriptions will also be outlined in section 3.2.

#### 3.1 Database Descriptions (if you use a database)

Our system will use a NoSQL database. We will have fields for the User, Foodshare, Picture, and Survey. The User, Foodshare and Survey will be permanent, while the pictures will be deleted from the database after two weeks.



### 3.2 File Descriptions

The database will use standard file types like jpeg, zip, json, and csv. Our system will use no non-standard file structure.

## 4 Requirements Matrix

The requirements matrix shows how each functional requirement in the System Requirements Specification corresponds with the components in the System Design Document. Its purpose is to ensure that every requirement is addressed in the system design and has methods or functions to complete the requirements. It will be referenced during implementation to determine the details of how each of these different requirements should be implemented in practice to work together to create the overall product. A 'user' is anyone who is using the application, a 'host' is someone who creates a food share, which any 'user' is able to do, and

Functional Requirement use case number	Functional Requirement name	System Component
UC-001	Users shall be able to sign up for push notifications.	subscribePush() subscribeSMS()
UC-002	Create food share	createShare() updateDatabase()
UC-003	Hosts shall be able to upload and post images inside their posts	uploadImage()
UC-004	View current food shares	getActiveShares()
UC-005	Ending food shares	endShare() presentSurvey()
UC-006	User Verifies Email	sendVerificationEmail() verifyEmail()
UC-007	User Filters food shares by Allergy.	filterShares()
UC-008	Host Adds Allergen warning.	addAllergenWarning() saveToDatabase()
UC-009	User accesses location map service	openMappingSoftware()
UC-010	View food share	getShare()

UC-011	Admin Views food share Data	getAllShares() exportShares() downloadShares()
UC-012	Admin Deletes food share	deleteShare() updateDatabase()
UC-013	Admin Bans Food Distributor	banDistributor() updateDatabase()

## Appendix A – Agreement Between Customer and Contractor

The customer and team agree that the document represents a finalized version that meets all agreed upon criteria and requirements. Both parties have thoroughly reviewed the document and all of its contents and agree that the information and structure meet the standards of the requirements. Any feedback or revisions have been addressed and remedied. The document is considered complete and ready for use.

If changes to the System Design Document are required, the team will begin by preparing a draft of the document with the changes. After the draft has been prepared, each team member will review the draft and note any problem with it they might have. Once all the problems have been addressed, the team will review the document once more and sign off on it, to show that they are satisfied with the state of the document. Once that has been completed, we will submit the draft to the client for review and approval.

Wesley Dumas

Signature: *Wesley Dumas* Date: 11/17/25

Corey Kaulenas

Signature: *Corey Aras Kaulenas* Date: 11/17/25

Makai Moody-Broen

Signature: *Makai Moody-Broen* Date: 11/14/25

Denis Sima

Signature: *Denis Sima* Date: 11/17/25

Jakob Sholler

Signature: *Jakob Sholler* Date: 11/16/25

Dr. Scott Marzilli

Signature:  Date: 11/17/25

Comments:

## Appendix B – Team Review Sign-off

All team members have thoroughly reviewed this document and reached full agreement on all of its content. No major concerns have been raised, and any minor concerns or clarifications raised by individuals are noted in the comments below their signature. Additionally, all members of the team have agreed upon the formatting and presentation of this document with no major complaints.

Wesley Dumas

Signature: *Wesley Dumas* Date: 11/17/25

Comments:

Corey Kaulenas

Signature: *Corey Aras Kaulenas* Date: 11/17/25

Comments:

Makai Moody-Broen

Signature: *Makai Moody-Broen* Date: 11/14/25

Comments:

Denis Sima

Signature: *Denis Sima* Date: 11/17/25

Comments:

Jakob Sholler

Signature: Jakob Sholler Date: 11/16/25

Comments:

## **Appendix C – Document Contributions**

Sholler Jakob, Contributions:

- Decomposition description

- 2.1 bullet points

Dumas Wesley, Contributions:

- Requirements Matrix

- Appendix A, B

Moody-Broen Makai, Contributions:

- Introduction

- Purpose

- Architectural design intro and 2.1 paragraphs

- Logo

Kaulenas Corey, Contributions:

- Database Description

- File Description

Sima Denis, Contributions: