Elaboration Phase Spec

Hannah Sadikovic, Matthew Roth, Faysal Bulbul, Chris Humphrey, Logan Coleman

Table of Contents

1. System Requirements 3

2. Trace Matrix 5

3. Use Cases 10

4. Use Case Diagrams 83

5. Class Diagrams 88

5. Windows Navigation Diagram 90

6. Database Design and Data Definitions 90

7. Systems Architecture 97

8. Design Procedures for Security Concerns and Non-functional Requirements 99

9. Gantt Chart 101

10. Prototypes 102

System Requirements

The system requirements are the basic needs that the system will require and should be referred to in the design phase when designing the system. The following list the requirements for each major business problem.

Our Story

* Their story should be easily visible to users
* The mission statement should be easily visible to users
* There should be links to additional information about the organization

Visibility

* There should be links to social media
* The website needs to be easily found when searched for

Dynamic Updates

* There should be a system that will allow their data to be updated automatically
* There should be a dynamic calendar
* There should be a dynamically updated itemized list for donation suggestions

Events

* There needs to be an easy system for communicating with RSOs and other organizations that host events
* The events should be updated within the dynamic calendar
* There should be information on the website that informs users how they can contact the pantry about leftover food donations

Donation

* There should be a link to an Amazon Wishlist
* There should be a link to the Funds for Student Affairs
* There should be information accessible to users about the days of operations of the pantry for the Drop-off box
* There should be information on the website that highlights the types of donations the pantry accepts
* There should be an itemized list that suggests which donations the pantry is in need of

Volunteer

* There should be a way for people to sign up as volunteers
* There should be an easy way for volunteers to log their hours
* There should be a way for volunteers to login to their portal

Communication

* There should be an easy way for RSOs to communicate with staff and volunteers of the pantry

Inventory

* Their inventory data should be dynamically updated
* There should be emphasis on the website on the types of donations they accept and do not accept
* Inventory data should be updated by staff through their portals

Sustainability Emphasis

* There should be emphasis on the website that part of their mission is sustainability
* There should be a way for staff to track how many pounds of compost they have created
* There should be a way for staff to dynamically update how many pounds of compost they have created

Additional Resources

* There should be a link for additional campus resources available to users
* There should be a way for users to submit recipe ideas
* There should be a link for Grab-n-Go Access

Technological Improvements

* There needs to be more hardware in addition to the singular iPad
* There should be a switch from Google-based applications to Microsoft applications
* The software Cardinal Cupboard chooses should be secure

Staff Needs

* There should be a way for staff to calculate how many donations they are receiving
* There should be a staff-only section accessible through the portal
* The staff should be able to access the schedule
* The staff should have access to forms submitted by potential volunteers

Contact Us

* There should be a way for users to easily contact Cardinal Cupboard
* There should be a way for users to subscribe to emails

User Login

* There should be a login available for staff/ volunteers

Trace Matrix

The trace matrix represents the joining of use cases, and business requirements. The left side column represents the use cases. The top row represents the business requirements. Each requirement is numbered. Each number corresponds to the same number in the top row of the trace matrix. To improve readability, the requirements were named with a number. Each X represents a unison of a requirement, and a use case.

1. Their story should be easily visible to users
2. The mission statement should be easily visible to users
3. There should be links to additional information about the organization
4. There should be links to social media
5. The website needs to be easily found when searched for
6. There should be a system that will allow their data to be updated automatically
7. There should be a dynamic calendar
8. There should be a dynamically updated itemized list for donation suggestions
9. There should be an easy system for communicating with RSOs and other organizations that host events
10. The events should be updated within the dynamic calendar
11. There should be information on the website that informs users how they can contact the pantry about leftover food donations
12. There should be a link to an Amazon Wishlist
13. There should be a link to the Funds for Student Affairs
14. There should be information accessible to users about the days of operations of the pantry for the Drop-off box
15. There should be information on the website that highlights the types of donations the pantry accepts
16. There should be an itemized list that suggests which donations the pantry is in need of
17. There should be a way for people to sign up as volunteers
18. There should be an easy way for volunteers to log their hours
19. There should be a way for volunteers to login to their portal
20. There should be an easy way for RSOs to communicate with staff and volunteers of the pantry
21. Their inventory data should be dynamically updated
22. There should be emphasis on the website on the types of donations they accept and do not accept
23. Inventory data should be updated by staff through their portals
24. There should be emphasis on the website that part of their mission is sustainability
25. There should be a way for staff to track how many pounds of compost they have created
26. There should be a way for staff to dynamically update how many pounds of compost they have created
27. There should be a link for additional campus resources available to users
28. There should be a way for users to submit recipe ideas
29. There should be more hardware in addition to the singular iPad
30. There should be a switch from Google-based applications to Microsoft applications
31. The software Cardinal Cupboard chooses should be secure
32. There should be a way for staff to calculate how many donations they are receiving
33. There should be a staff-only section accessible through the portal
34. The staff should be able to access the schedule
35. The staff should have access to forms submitted by potential volunteers
36. There should be a way for users to easily contact Cardinal Cupboard
37. There should be a way for users to subscribe to emails
38. There should be a login available for staff and volunteers
39. There should be a link to access Grab n Go

Diagram

Description automatically generated

Improving Cardinal Cupboard

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 24/10/2021 | 1.0 | First Vision | Chris Humphrey, Faysal Bulbul, Matthew Roth, Logan Coleman, Hannah Sadikovic |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Use Case Specification: <Login Student>

# Use-Case Name

Login Student

## Brief Description

This use case shows how a student will login to the Cardinal Cupboard website.

# Flow of Events

## Basic Flow

* Student navigates website
* Student chooses Login
* Student enters username
* Student enters password
* Student chooses Submit
* Use Case Ends

## Alternative Flows

None

# Special Requirements

There are no special requirements associated with this use case

# Pre-conditions

## < Pre-condition One >

A student must have a U-link account.

# Post-conditions

## < Post-condition One >

A student can access resources such as volunteering and donating

# Extension Points

There are no extension points associated with this use case

Chart, box and whisker chart

Description automatically generated

**Use Case Specification: <Subscribe Mailing List>**

1. **Use-Case Name**

Subscribe Mailing List

**1.1 Brief Description**

This use case describes how a person subscribes to the Cardinal Cupboards mailing list.

1. **Flow of Events**

**2.1 Basic Flow**

* Guest navigates website
* Guest chooses resources tab
* Guest chooses mailing list
* Guest enters email
* Guest chooses Subscribe
* Use case ends

**2.2 Alternative Flows**

There are no alternative flows for this use case

**3 Special Requirements**

**3.1 < First Special Requirement >**

There are no special requirements associated with this use case

1. **Pre-conditions**

There are no pre-conditions associated with this use case

1. **Post-conditions**

**5.1 < Post-condition One >**

A Guest will be subscribed to the mailing list and receive updates regarding the cupboard to their email.

1. **Extension Points**

There are no extension points associated with this use case

Chart

Description automatically generated with medium confidence

Use Case Specification: Calculate Values

# Use-Case Name

Calculate Values

## Brief Description

The role of this use case is for employees to calculate the total amount of donations to Cardinal Cupboard

# Flow of Events

## Basic Flow

* User clicks finances
* User clicks total amount

## Alternative Flows

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User is logged in as an employee

# Post-conditions

Total amount of money donated is calculated

# Extension Points

N/A

A picture containing chart

Description automatically generated

**Use Case Specification: <Submit Volunteer Application >**

1. **Use-Case Name**

Submit Volunteer Application

**1.1 Brief Description**

This use case shows how a student can sign up to volunteer

1. **Flow of Events**

**2.1 Basic Flow**

* Student navigates website
* Student logs in
* Student chooses volunteer tab
* Student chooses form
* Student enters email
* Student chooses Submit
* Use Case Ends

**2.2 Alternative Flows**

There are no alternative flows for this use case

1. **Special Requirements**

**3.1 < First Special Requirement >**

A volunteer must have an account at the University of Louisville.

1. **Pre-conditions**

**4.1 < Pre-condition One >**

A student must be logged in

1. **Post-conditions**

**5.1 < Post-condition One >**

A Student will be able to volunteer at the Cardinal Cupboard.

1. **Extension Points**

There are no extension points for this use case.

Chart, box and whisker chart

Description automatically generated

Use Case Specification: <Submit Recipe>

## Brief Description

This use case describes the action that a user takes when they want to submit a recipe form on Cardinal Cupboard’s website. The actor is any user of the system.

# Flow of Events

## Basic Flow

* User navigates website
* User chooses “Resources”
* User chooses “Recipes”
* User uploads recipe
* User submits recipe
* Use-Case ends

## Alternative Flows

### < First Alternative Flow >

Since the recommended recipes submitted are based on the current food items stocked by the pantry, formatted as a dynamic itemized list, the user may wish to view the list before submitting a recipe. The basic flow would be:

* User navigates website
* User chooses “Resources”
* User views Itemized List

### < Second Alternative Flow >

The user may also wish to view recipes already submitted so they may submit a new one. The basic flow would be:

* User navigates website
* User chooses “Recipes”
* User views “Recommended Recipes”

# Special Requirements

## < First Special Requirement >

The system must allow the user to upload a recipe as a word document or pdf.

# Pre-conditions

## < Pre-condition One >

There must be a dynamic list of current food items in the pantry available for users to click on and read in order to determine recommended recipes.

# Post-conditions

## < Post-condition One >

The forms will be made available for viewing by the staff members through the login portal.

## < Post-condition Two >

There must be a dynamic itemized list of currently stocked food items available for the user to view.

# Extension Points

None

Diagram

Description automatically generated

* User navigates website
* User chooses “Resources”
* User chooses “Recipes”
* User uploads recipe
* User submits recipe
* Use-Case ends

Use Case Specification: <View Recipe>

# Use-Case Name

## Brief Description

This use case describes how the user will view recommended recipes submitted by users on Cardinal Cupboard’s website. The user “chooses” a recipe to view it. The user is a staff member who must be logged in to access the recipes and can then filter out the recipes to search for a certain recipe.

# Flow of Events

## Basic Flow

* User navigates website
* User chooses “Recipes”
* User logs in
* User chooses recipe
* User reads recipe
* Use-Case ends

## Alternative Flows

### < First Alternative Flow >

If the user wishes to filter the recipes to view a certain one, an alternative flow may be:

* User filters recipes
* User chooses recipe
* User reads recipe
* Use-Case ends

# Special Requirements

## < First Special Requirement >

The system should allow for a LIFO (last-in-first-out) order of recipes to be listed for the user to view as the most recent recipes submitted relate to the current inventory the pantry is stocked with.

## < Second Special Requirement >

The system should filter out and highlight the recipes that have not been viewed yet.

# Pre-conditions

## < Pre-condition One >

There must be at least one submitted recipe in order for a user to view it.

# Post-conditions

## < Post-condition One >

The system will filter out the already viewed recipes and highlight the currently unviewed recipes.

# Extension Points

None

Diagram

Description automatically generated

* User navigates website
* User chooses “Recipes”
* User logs in
* User chooses recipe
* User reads recipe
* Use-Case ends

Use Case Specification: <Add Recipe>

# Use-Case Name

## Brief Description

This use case describes the action that a staff member would take to choose a recipe from the submitted forms for recommended recipes and add the recipe to a list of saved recipes.

# Flow of Events

## Basic Flow

* User navigates website
* User logs in
* User views inventory
* User views recipes
* User chooses recipe
* User adds recipe
* Use-Case ends

## Alternative Flows

### < First Alternative Flow >

The user can either save the recipe to a file of saved recipes that can then be modified or deleted, or they can directly add the recipe to the final list of recipes viewable by any user. The basic flow allows the user to do the latter. The alternative flow may be:

* User chooses recipe
* User saves recipe
* Use-Case ends

#### < An Alternative Subflow >

From the previous alternative flow, the user can take the following steps in any order:

* User adds recipe
* User edits recipe
* User edits recipe
* User adds comment
* Use-Case ends

# Special Requirements

## < First Special Requirement >

The recipes that are added to a saved list must be filtered in a LIFO (last-in-first-out) order so that users viewing the list can see recipes more relevant to current food items stocked by the pantry.

# Pre-conditions

## < Pre-condition One >

A recipe must have been created. The user must have already viewed and chosen a specific recipe to add.

# Post-conditions

## < Post-condition One >

The recipes added by the user are saved to a list of “acceptable recipes” that can be viewed by any user on the website.

# Extension Points

Graphical user interface, diagram

Description automatically generatedNone

* User navigates website
* User logs in
* User views inventory
* User views recipes
* User chooses recipe
* User adds recipe
* Use-Case ends

Use Case Specification: <Modify Recipe>

# Use-Case Name

## Brief Description

This use case describes the action a user would take to modify a saved recipe before adding it to the final list available to all users. Modifying the recipe includes changing, deleting, or adding certain elements of the recipe.

# Flow of Events

## Basic Flow

* User views recipe
* User chooses recipe
* User saves recipe
* User navigates “saved recipes”
* User changes recipe
* User saves recipe
* Use-Case ends

## Alternative Flows

### < First Alternative Flow >

An alternative flow may be:

* User navigates “saved recipes”
* User adds comment
* User saves recipe
* Use-Case ends

# Special Requirements

None

# Pre-conditions

## < Pre-condition One >

The user must already have saved a recipe for later to be able to modify it.

# Post-conditions

## < Post-condition One >

After modifying the recipe, the system allows the user to either further modify the recipe, save the recipe for later use, or add the recipe to the final list of recipes available to all users.

# Extension Points

None

Graphical user interface, diagram

Description automatically generated

* User views recipe
* User chooses recipe
* User saves recipe
* User navigates “saved recipes”
* User changes recipe
* User saves recipe
* Use-Case ends

Use Case Specification: <Delete Recipe>

# Use-Case Name

## Brief Description

This use case describes how a user would delete a recommended recipe from the list of submitted recipes.

# Flow of Events

## Basic Flow

* User views recipe
* User deletes recipe
* Use-Case ends

## Alternative Flows

None

# Special Requirements

None

# Pre-conditions

## < Pre-condition One >

There must be a submitted recipe available for the User to delete.

# Post-conditions

## < Post-condition One >

The recipe will be permanently deleted from the website.

# Extension Points

None

* Graphical user interface, diagram

  Description automatically generated User views recipe
* User deletes recipe
* Use-Case ends

Use Case Specification: <View Application>

# Use-Case Name

## Brief Description

This use case describes how a user would view a submitted volunteer application. The actor is a staff member.

# Flow of Events

## Basic Flow

* User navigates website
* User logs in
* User chooses “Applications”
* User views an application
* Use-Case ends

## Alternative Flows

None

# Special Requirements

**< First Special Requirement >**

The application must be submitted as a file that can be opened up as a word document or pdf.

# Pre-conditions

## < Pre-condition One >

There must be at least one application available for the user to view.

# Post-conditions

## < Post-condition One >

After viewing, the applications are ready to be added to an accepted list, modified, or deleted to a rejected list.

# Extension Points

None

* Diagram

  Description automatically generated User navigates website
* User logs in
* User chooses “Applications”
* User views an application
* Use-Case ends

Use Case Specification: <Add Application>

# Use-Case Name

## Brief Description

This use case describes how a user would add a volunteer application to a list of accepted applications.

# Flow of Events

## Basic Flow

* User views application
* User chooses application
* User adds application
* Use-Case ends

## Alternative Flows

### < First Alternative Flow >

There might be some applications that the user wishes to add to the list, but at a different time, or possibly review with other staff members before adding. The steps taken after step 2 would be:

* User saves Recipe
* User navigates “Saved Recipes”

#### < An Alternative Subflow >

From the previous steps, the user can take the following steps in any order:

* User adds recipe
* User modifies recipe
* User deletes recipe
* User adds comment

# Special Requirements

## < First Special Requirement >

The applications mut be forms when submitted.

# Pre-conditions

## < Pre-condition One >

There must be applications already submitted for the use case to be performed.

# Post-conditions

## < Post-condition One >

The applications will be added to a list that can be modified and is only accessible by other volunteers.

# Extension Points

None

Graphical user interface, diagram, application

Description automatically generated

* User views application
* User chooses application
* User adds application
* Use-Case ends

Use Case Specification: <Delete Application>

# Use-Case Name

## Brief Description

This use case describes how a user would delete a volunteer application from the submitted applications files. When an application is deleted, it is moved to a “rejected applications” place. The actor is a staff member.

# Flow of Events

## Basic Flow

* User chooses “Applications”
* User reads application
* User deletes application
* Use-Case ends

## Alternative Flows

None

# Special Requirements

## < First Special Requirement >

When an application is deleted, it is not permanently deleted, and is rather moved to a “rejected applications” section of the website accessible only by staff members.

# Pre-conditions

## < Pre-condition One >

There must be submitted applications available for the user to access.

# Post-conditions

## < Post-condition One >

The application will be moved to the “rejected applications” section of the website.

# Extension Points

None

* Diagram

  Description automatically generated User chooses “Applications”
* User reads application
* User deletes application
* Use-Case ends

**Use Case Specification: Contact Cardinal Cupboard**

Use-Case Name

Brief Description

The role of this use case is to enable a user of the website to contact Cardinal Cupboard directly via email or telephone. The purpose of this use case is that every business should have a way for customers to get into contact with them.  Another purpose is to allow users the ability to get their questions answered. By allowing users a way to contact them, Cardinal Cupboard will create an environment that is conducive to high user satisfaction.

Flow of Events

Basic Flow

* User clicks website
* User scrolls down
* User clicks contact us
* User chooses method of contact
* User enters email
* User enters phone number
* Cardinal Cupboard volunteer responds to contact
* Use case ends

Alternative Flows

None

Special Requirements

None

Pre-conditions

Pre-condition One

Cardinal Cupboard must have a method of contact.

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

None

**Use Case Specification: Access Grab & Go**

Use-Case Name

Brief Description

The role of this use case is to enable a visitor of the website to be able to access the grab and go function. The purpose is so that visitors have a way of quickly building a grab bag in order to get food.

Flow of Events

Basic Flow

* The user will click access grab and go
* The user will fill out a form
* Use enters foods they want in the grab bag
* The data is collected and processed
* Volunteers make the grab bag
* User picks up grab bag
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

The website must be operational.

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

The website must be operational

**4.2            The user must have valid credentials**

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

An extension point details how a use case can be combined with another, and extended.

Name of Extension Point

**Use Case Specification: View History**

Use-Case Name

Brief Description

The role of this use case is to allow visitors of the Cardinal Cupboard website a way to view the history of the organization. The purpose of this use case is to help inform visitors. Users of the website will want to know the history behind the organization they are about to get help from. By displaying the history, Cardinal Cupboard will be creating more informed users, and in the process attract more people.

Flow of Events

Basic Flow

* User clicks on website
* User clicks view history
* User views history
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

First Special Requirement

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

Cardinal Cupboard must have a history page

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

Name of Extension Point

Box and whisker chart

Description automatically generated with medium confidence

**Use Case Specification: View Mission Statement**

Use-Case Name

Brief Description

The role of this use case is so that visitors to the website can view the mission statement of Cardinal Cupboard. The purpose of this use case is that the mission statement is an extremely important part of any organization. Without one, prospective “customers” will have no idea what the business stands for. By including a mission statement, Cardinal Cupboard can appeal to more people.

Flow of Events

Basic Flow

* User clicks website
* User clicks Mission Statement
* User views Mission Staement
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

First Special Requirement

None

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

Cardinal Cupboard must have a mission statement

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

Name of Extension Point

A picture containing box and whisker chart

Description automatically generated

**Use Case Specification: View Additional Information**

Use-Case Name

Brief Description

The role of this use case is to ensure that visitors of the website can view additional information if they need it. The purpose of this use case is to help guide visitors to more resources, and to get them help if they need it. Every website has something similar to this. By allowing users access to more information, Cardinal Cupboard will make its audience more informed.

Flow of Events

Basic Flow

* User clicks onto website
* User views homepage
* User clicks view additional information
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

First Special Requirement

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

Cardinal Cupboard must have a view additional information page

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

An extension point details how a use case can be combined with another, and extended.

Name of Extension Point

**Use Case Specification: View Social Media**

1. **Use-Case Name**
2. **Brief Description**

The role of this use case is so that a visitor to the website can view the social media links of Cardinal Cupboard. The purpose of this use case is that many young people, the primary college audience, use social media. By having the social media links, Cardinal Cupboard can appeal to young people, and grow their audience.

1. **Flow of Events**
2. **Basic Flow**

* User scrolls down
* User chooses social media
* User clicks Twitter
* Use case ends

1. **Alternative Flows**

* User scrolls down
* User chooses social media
* User clicks Instagram
* Use case ends

1. An Alternative Subflow
2. *Second Alternative Flow*
3. **Special Requirements**

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

1. **Cardinal Cupboard must have social media**

1. **Pre-conditions**

Preconditions are the previous state of the system that must be present for the use case to be operational.

1. **Cardinal Cupboard must have social media**
2. **Post-conditions**

A Postcondition is the state of the system right after the use case is performed.

1. **The user will be taken to the social media**
2. **Extension Points**
3. **Name of Extension Point**

**Use Case Specification: View Dynamic Updates**

Use-Case Name

Brief Description

The role of this use case is to allow users to view the dynamically updating content across the many pages of the Cardinal Cupboard website. The purpose of this use case is that dynamically updating content will allow users to immediately know information with just a glance. In addition, it is a great way for Cardinal Cupboard to show off the progress they have made with helping students. By including dynamically updating content, Cardinal Cupboard will have a more satisfying website.

Flow of Events

Basic Flow

* User clicks onto website
* User views homepage
* User scrolls down
* User sees the stats on compost
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

The code used to calculate the values must be operational

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

Cardinal Cupboard must have data available to be displayed

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points

An extension point details how a use case can be combined with another, and extended.

Name of Extension Point

**Use Case Specification: <View Dynamic Calendar>**

* 1. **Use-Case Name**

View Dynamic Calendar

**1.1 Brief Description**

This use case shows how a user will view the dynamic calendar of the Cupboard.

1. **Flow of Events**

**2.1 Basic Flow**

* Guest navigates website
* Guest chooses resources tab
* Guest chooses calendar
* Use case Ends

**2.2 Alternative Flows**

There are no alternative flows associated with this use case

**3. Special Requirements**

There are no special requirements associated with this use case

1. **Pre-conditions**

There are no pre-conditions associated with this use case

1. **Post-conditions**

**5.1 < Post-condition One >**

A Guest can view important dates for events and availability of the Cupboard.

1. **Extension Points**

There are no extension points associated with this use case.

Chart, box and whisker chart

Description automatically generated

Use Case Specification: View Events

# Use-Case Name

View Events

## Brief Description

The role of this use case is to show how guests can view what kind of events Cardinal Cupboard is participating in.

# Flow of Events

## Basic Flow

* User will hit service

## Alternative Flows

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User must be at the landing page

# Post-conditions

User can view events that Cardinal Cupboard is taking part in.

# Extension Points

N/A

Chart

Description automatically generated with low confidence

Use Case Specification: Add Events

# Use-Case Name

Add Events

## Brief Description

The role of this use case is to show how employees can add the events Cardinal Cupboard is participating in.

# Flow of Events

## Basic Flow

* User will click dynamic calendar
* User will click add event
* User adds event details
* User clicks finish

## Alternative Flows

* User modifies event
* User deletes events

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User is logged in as an employee

# Post-conditions

New event is added to the dynamic calendar.

# Extension Points

N/A

A picture containing chart

Description automatically generated

Use Case Specification: Modify Events

# Use-Case Name

Modify Events

## Brief Description

The role of this use case is to show how Employees can modify created events on the dynamic calendar to Cardinal Cupboard.

# Flow of Events

## Basic Flow

* User clicks on dynamic calendar
* User clicks on an event
* User clicks modify event

## Alternative Flows

User adds an event

User deletes an event

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User must be logged into the system as an employee.

User at the main volunteer page.

# Post-conditions

Event is modified to the new information

# Extension Points

N/A

Table

Description automatically generated with low confidence

Use Case Specification: Delete Events

# Use-Case Name

Delete Events

## Brief Description

The role of this use case is to show how Employees can delete events created on the Cardinal Cupboard Dynamic Calendar

# Flow of Events

## Basic Flow

* User clicks on dynamic calendar
* User clicks on event
* User clicks delete event

## Alternative Flows

User adds an event

User modifies an event

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User must be logged into the system as an employee.

User at the main volunteer page.

# Post-conditions

Event is deleted off the dynamic calendar.

# Extension Points

N/A

Chart

Description automatically generated

**Use Case Specification: <View Inventory >**

1. **Use-Case Name**

View inventory

**1.1 Brief Description**

This use case describes how a staff member will view the inventory

1. **Flow of Events**

**2.1 Basic Flow**

* Staff navigates website
* Staff chooses login
* Staff enters username
* Staff enters password
* Staff chooses Submit
* Staff chooses inventory
* Staff views inventory
* Use case ends

**2.2 Alternative Flows**

There are no alternative flows for this use case

1. **Special Requirements**

There are no special requirements for this use case

**4. Pre-conditions**

**4.1 < Pre-condition One >**

Must be a staff member

1. **Post-conditions**

**5.1 < Post-condition One >**

Staff can view the inventory. Staff can see what things they have and also things which are needed for the Cupboard.

1. **Extension Points**

There are no extension points for this use case

A picture containing diagram

Description automatically generated

**Use Case Specification: <View Itemized List>**

1. **Use-Case Name**

View Itemized List

**1.1 Brief Description**

This use case will describe how a Guest (Any User) will view the itemized list on the Cardinal Cupboard website.

1. **Flow of Events**

**2.1 Basic Flow**

* Guest navigates website
* Guest chooses resources tab
* Guest chooses itemized list tab
* Guest views itemized list
* Use Case ends

**2.2 Alternative Flows**

There are no alternative flows for this use case

1. **Special Requirements**

There are no special requirements for this use case

1. **Pre-conditions**

There are no pre-conditions associated with this use case

1. **Post-conditions**

**5.1 < Post-condition One >**

A user can view the itemized list and see what the Cupboard currently has in stock, what they need most of, and what they need least of to allow users to donate the most appropriate items at the most appropriate time.

1. **Extension Points**

None

Chart, box and whisker chart

Description automatically generated

Use Case Specification: <View Types of Donations>

# Use-Case Name

View Types on Donations

## Brief Description

This use case describes how a user would find and view the different types of donations.

# Flow of Events

## Basic Flow

User logs on as guest.

User selects the donation tab.

User views the page.

Use case ends.

## Alternative Flows

### < First Alternative Flow >

User logs on.

User selects the donate button.

User selects “Other Ways to Donate”

Use case ends.

# Special Requirements

## < First Special Requirement >

One of the ways to donate is Amazon Wishlist, if the user selects it, they will need an Amazon account to purchase anything.

# Pre-conditions

## < Pre-condition One >

User signs on.

# Post-conditions

## < Post-condition One >

Users may choose to follow through with a donation or not.

# Extension Points

N/A

A picture containing chart

Description automatically generated**Types of Donation Basic Flow**

User logs on as guest.

User selects the donation tab.

User views the page.

Use case ends.

Use Case Specification: <View Donation Emphasis>

# Use-Case Name

View Donation Emphasis

## Brief Description

This use case describes how a user would find and view the donation emphasis.

# Flow of Events

## Basic Flow

User clicks to website.

User selects the donation tab.

User scrolls down.

User views Donation Emphasis.

Use case ends.

## Alternative Flows

### < First Alternative Flow >

User signs on.

User clicks to search.

User keys “Donation Emphasis”

User presses enter.

Use case ends.

# Special Requirements

N/A

# Pre-conditions

## < Pre-condition One >

User signs on.

# Post-conditions

N/A

# Extension Points

N/A

Table

Description automatically generated**Donation Emphasis Basic Flow**

User clicks to website.

User selects the donation tab.

User scrolls down.

User views Donation Emphasis.

Use case ends.

Use Case Specification: <View Sustainability Emphasis>

# Use-Case Name

View Sustainability Emphasis

## Brief Description

This use case describes how a user would find and view the Sustainability Emphasis.

# Flow of Events

## Basic Flow

User clicks to website.

User views homepage.

User scrolls down.

User views Sustainability Emphasis.

Use case ends.

## Alternative Flows

### < First Alternative Flow >

User signs on.

User clicks to search.

User keys “Sustainability Emphasis”

User presses enter.

Use case ends.

# Special Requirements

N/A

# Pre-conditions

## < Pre-condition One >

User signs on.

# Post-conditions

N/A

# Extension Points

N/A

A picture containing table

Description automatically generated**Sustainability Emphasis Basic Flow**

User clicks to website.

User views homepage.

User scrolls down.

User views Sustainability Emphasis.

Use case ends.

Use Case Specification: <View Days of Operation>

# Use-Case Name

View Days of Operation

## Brief Description

This use case describes how a user would find and view Cardinal Cupboard’s days of operation.

# Flow of Events

## Basic Flow

User signs on.

User scrolls down.

## Alternative Flows

### < First Alternative Flow >

User signs on.

User clicks to search.

User keys “Days of Operation”

User presses enter.

# Special Requirements

N/A

# Pre-conditions

## < Pre-condition One >

User signs on.

# Post-conditions

N/A

# Extension Points

N/A

Table

Description automatically generated**Days of Operation Basic Flow**

User clicks to website.

User views homepage.

User scrolls down.

User views Days of Operation.

Use case ends.

Use Case Specification: Find Info on Drop Box

# Use-Case Name

Find Info on Drop Box

## Brief Description

The role of this use case is to show how guests can learn how the Drop Box works at Cardinal Cupboard

# Flow of Events

## Basic Flow

* User clicks resources

## Alternative Flows

* User clicks home
* User clicks about
* User clicks donate
* User clicks services

# Special Requirements

Special requirements are an event necessary for the event flow, but not usually specified in the event flow.

# Pre-conditions

User must be at the main landing page or any of the sub pages of it.

# Post-conditions

# Extension Points

N/A

A picture containing table

Description automatically generated

Use Case Specification: <Donate with Amazon Wishlist>

# Use-Case Name

Donate with Amazon Wishlist

## Brief Description

The user will have options of different ways to donate. For this use case, it allows the user to donate using Amazon Wishlist, as opposed to monetary donation or donating supplies in person.

# Flow of Events

## Basic Flow

User signs on as guest.

User navigates to donate page.

User selects Amazon Wishlist link.

Use case Ends.

# Special Requirements

## < First Special Requirement >

The user will have to have an Amazon account in order to purchase an item off the Wishlist. However, they do not need one just to view the Wishlist.

# Pre-conditions

## < Pre-condition One >

The user logs onto the system.

The Amazon Wishlist has items for the user to look through and potential purchase.

# Post-conditions

## < Post-condition One >

User is taken offsite to the Amazon Wishlist and can choose whether to buy something or not.

# Extension Points

N/A

Diagram

Description automatically generated**Amazon Wishlist Basic Flow**

User signs on as guest.

User navigates to donate page.

User selects Amazon Wishlist link.

Use case Ends.

Use Case Specification: <Make a Monetary Donation>

# Use-Case Name

Make a monetary donation

## Brief Description

This use case communicates how a user would make a monetary donation.

# Flow of Events

## Basic Flow

User logs on as guest.

User selects the donate button.

User selects amount to donate.

User keys in card information.

User selects submit.

Use case ends.

## Alternative Flows

### < First Alternative Flow >

User logs on as guest.

User selects the donation page.

User selects monetary donation link.

Use case ends.

# Special Requirements

## N/A

# Pre-conditions

## < Pre-condition One >

User logs onto the system.

# Post-conditions

## < Post-condition One >

User is taken to the payment system page and may choose to donate or not.

# Extension Points

N/A

A picture containing table

Description automatically generated**Monetary Donation Basic Flow**

User logs on as guest.

User selects the donate button.

User selects amount to donate.

User keys in card information.

User selects submit.

Use case ends.

**Use Case Specification: <Communicate Drop-off>**

1. **Use-Case Name**

Communicate Dropoff-box

## 1.1 Brief Description

This use case describes how a Guest can communicate with the Cardinal Cupboard to drop off food items

1. **Flow of Events**

**2.1 Basic Flow**

* Guest navigates website
* Guest chooses resources tab
* Guest chooses “Drop off-box”
* Guest chooses Contact
* Use Case Ends

**2.2 Alternative Flows**

*2.1< First Alternative Flow >*

* Guest navigates website
* Guest chooses resources
* Guest chooses drop-off
* Guest chooses Contact
* Guest chooses call
* Guest Calls Cupboard
* Use Case Ends

## 2.3 < An Alternative Sub flow >

* In step 4, of the basic flow, a user can add notes about where and when they are dropping off the item/items.

## *2.4 Second Alternative Flow >*

* Guest navigates website
* Guest chooses resources
* Guest chooses “Drop off”
* Guest chooses Contact
* Guest chooses form
* Guest enters credentials
* Guest chooses Submit
* Use Case ends

1. **Special Requirements**

**3.1 < First Special Requirement >**

A staff member must be checking forms constantly and marking down when and where users are dropping off items.

1. **Pre-conditions**

**4.1 Pre-condition One >**

A Guest must have food ready to be picked up

1. **Post-conditions**

**5.1 < Post-condition One >**

A Guest will know what days and time they can drop off items.

1. **Extension Points**

None

Chart, box and whisker chart

Description automatically generated

**Use Case Specification: <Communicate RSO>**

1. **Use-Case Name**

Communicate RSO

**1.1 Brief Description**

This use case shows how a user will get in touch with an RSO.

1. **Flow of Events**

**2.1 Basic Flow**

* Staff navigates website
* Staff chooses login
* Staff enters username
* Staff enters password
* Staff chooses Submit
* Staff contacts event
* Staff chooses email
* Staff sends email
* Use case ends

1. **Special Requirements**

None

1. **Pre-conditions**

No pre-conditions are associated with this use case

1. **Post-conditions**

**5.1 < Post-condition One >**

A user will have more information on RSO events.

1. **Extension Points**

None

A picture containing diagram

Description automatically generated

**Use Case Specification: Backup System**

Use-Case Name

Brief Description

The role of this use case is to securely backup the system in case of a failure. The purpose of this use case is to ensure that a reliable backup of the system is always available in the event of an emergency. By securely backing of the system often, Cardinal Cupboard will have the security that is expected by modern website users. Modern users also don’t like downtime, and by making secure backups, there will be little to not downtime.

Flow of Events

Basic Flow

* The admin logs into the system using their credentials.
* The admin starts the backup process
* The website data is backed up
* Use case ends.

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

The website must be operational

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

The admin must have valid credentials

**4.2        The website must be operational**

**4.3        The program used for backup must be operational**

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

The system backup is ready in the event of an emergency

Extension Points

An extension point details how a use case can be combined with another, and extended.

Recover System

## Box and whisker chart Description automatically generated with low confidence

**Use Case Specification: Recover System**

Use-Case Name

Brief Description

This use case details how an administrator will recover the system using a backup. This will allow the system to safely recover in the event of a failure.

Flow of Events

Basic Flow

* Administrator logs in
* Administrator chooses a backup
* Administrator recovers the system
* Use case ends

Alternative Flows

First Alternative Flow

*An Alternative Subflow*

Second Alternative Flow

Special Requirements

Special requirements are an event necessary for  the event flow, but not usually specified in the event flow.

The system must have a backup to recover

Pre-conditions

Preconditions are the previous state of the system that must be present for the use case to be operational.

The system must have a backup to recover

Post-conditions

A Postcondition is the state of the system right after the use case is performed.

Post-condition One

Extension Points   
An extension point details how a use case can be combined with another, and extended.

Backup System

A picture containing diagram

Description automatically generated

Use Case Diagrams

Use case diagrams are made up of actors, use cases, and the relationships between them. The actor is represented by the stick figure, the use cases are represented by ovals, and the lines that connect the two are the associations between them.

For example, the first diagram shows the choices a staff member, the actor, would have for related use cases: view application, add application, delete application, view recipe, add recipe, modify recipe, and delete recipe. The class APPLICATION is explained in the Class Diagrams section of the document and broken down into its relevant use case in the diagram, along with the other classes.

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

![Diagram

Description automatically generated]()

![Diagram

Description automatically generated]() ![Diagram

Description automatically generated]()

Diagram

Description automatically generated

Class Diagram

 A class diagram is a representation of the use cases and describes the structure of a system by showing the use cases classes, attributes, methods, and their relationship among objects. It shows a developer how the system should work and run smoothly. With that being said, the attributes and methods must be specified exactly how the designer would want the system to look like and behave. For the class diagram, we used a method called verb-noun analysis where we named the class and put that name in the first row of the class in the form of a noun from the appropriate use case for the class being made. We then identified all the nouns and listed them in the second row of the class. Finally, we identified all the verbs and listed them in the third column. After we had all our classes, we had to find relationships between them. There are two main classes that all the sub classes go back to. They are the Login class and the Load site class. An arrow is connected from all sub classes to either the Login Class or the Load Site Class. The Login class is a prerequisite for all sub classes that people who deal with the backend of the website, staff members. The Load site class is a prerequisite for all sub classes that deal with the front-end of the website, guests as well as staff members.

Timeline

Description automatically generated

Windows Navigation Diagram

The windows navigation diagram shows the basic structure of the website. It starts with the broadest part, the home page, and becomes more specific from there. The diagram shows how the entire website is connected to itself.

Diagram, engineering drawing

Description automatically generated

Database Design and Data Definitions

The following diagram is a relational database diagram designed in 3NF (third normal form). Below that are the data descriptions about the attributes of each class, displayed in table format.

Graphical user interface, application, Word

Description automatically generated

**Login Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **UserID** | A username that uniquely identifies a student or staff member | Max 40 bytes | string | **Primary Key** |
| **Password** | A password that uniquely identifies a student or staff member | Max 40 bytes | string | N/A |

**Scheduling table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **AmountHours** | The amount of hours that were volunteered. | Max 40 bytes | string | **Primary Key** |
|  |  |  |  |  |
|  |  |  |  |  |
| **Name** | The name of the volunteer who performed the service | Max 40 bytes | string | N/A |
| **Date** | The date the service was performed | Max 40 bytes | string | N/A |

**Drop off Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Description** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **PhoneNumber** | Phone number of a guest | 11 bytes | Long unsigned int | **Primary Key** |
| **Email** | Email address of a guest | Max 40 bytes | string | N/A |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| **Items** | Food items being dropped off | Max 255 bytes | string | N/A |
| **FirstName** | First name of guest | 40 bytes | string | N/A |
| **LastName** | Last name of guest | 40 bytes | string | N/A |

**RSO Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Description** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **PhoneNumber** | Phone number of RSO contact | Max 11 bytes | Long unsigned int | **Primary Key** |
| **RSOName** | Name of RSO | Max 40 bytes | string | N/A |
| **EventName** | Name of an RSO event | Max 40 bytes | string | N/A |
| **Email** | Email of Head of RSO event | Max 40 bytes | string | N/A |

**Subscription Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Description** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **Email** | Email of the Guest User that is subscribing to the mailing list | Max 255 bytes | string | **Primary Key** |
| **FirstName** | First name of user | Max 40 bytes | string | **N/A** |
| **LastName** | Last name of user | Max 40 bytes | string | **N/A** |

**Inventory Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Description** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** |
| **FoodName** | Description of type of food item | Max 40 bytes | string | **Primary Key** |
| **FoodItems** | Food items the pantry currently has in stock | Max 255 bytes | string | **N/A** |
| **NonFoodItems** | Non-Food Items that the pantry has in stock | Max 255 bytes | string | N/ A |
| **Quantity** | Quantity of inventory item | Max 255 bytes | string | N/A |
| **ExpirationDate** | Expiration date of food items | Max 20 bytes | string | N/A |

**Itemized List Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Description** | **Attribute Size** | **Attribute Type** | **Primary/Foreign Key** | |
| **FoodName** | Description of type of food item | Max 40 bytes | string | **Foreign Key** |  |
| **NeededFoodItem** | Food item Cardinal Cupboard needs | Max 255 bytes | string | **N/A** | |
| **NeededNonFoodItem** | Non-food item Cardinal Cupboard needs | Max 255 bytes | string | N/A |  |

**Applications table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **UserID** | Username of staff Member | **Max 10 bytes** | string | Primary Key |
| **Password** | Password of staff member | **Max 10 bytes** | string | N/A |
| **Applications** | The applications that staff can add, modify, or delete. | **Max 1MB** | string | N/A |

**Forms table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **FormID** | The identifier for the form being submitted. | **Max 20 bytes** | Long unsigned int | **Primary Key** |
| **RecipeForm** | The recipe form users can submit. | **Max 100 bytes** | string | **N/A** |
| **ApplicationForm** | The volunteer application form users can submit. | **Max 100 bytes** | string | N/A |
| **FirstName** | First name of user. | **Max 40 bytes** | string | N/A |
| **LastName** | Last name of user. | **Max 40 bytes** | string | N/A |
| **Phone Number** | Phone number of user. | **Max 11 bytes** | Long unsigned int | Foreign Key |
| **StudentID** | Student ID number | **Max 10 bytes** | Long unsigned int | Foreign Key |

**Recipe table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **RecipeID** | The Recipe Name | **Max 40 bytes** | String | Primary Key |
| **Inventory** | The inventory that both guest users and staff can view. | **Max 10GB** | String | N/A |
| **Recipes** | The recipes that staff can add, modify, or delete. | **Max 5 GB** | String | N/A |

**Contact Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **PhoneNumber** | Student’s phone number | **Max 11 bytes** | long unsigned Int | **Primary Key** |
| **FirstName** | Student’s first name | **Max 40 bytes** | String | N/A |
| **LastName** | Student’s last name | **Max 40 bytes** | String | N/A |
| **Email** | Student’s email | **Max 40 bytes** | String | N/A |

**Amazon Wishlist Table:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | | **Attribute Size** | | **Attribute Type** | | **Primary or Foreign Key?** | |
| **AmazonWishlist** | Cardinal Cupboard’s wishlist with items people can donate from. | | **Max 255 bytes** | | string | | **Primary Key** | |
| **ItemSelected** | Items selected from the wishlist | **Max 40 bytes** | | string | | N/A | |  |
| **Quantity** | Number of items on wishlist | **Max 255 bytes** | | Long unsigned int | | N/A | |  |
| **Item** | Description of item on the wishlist | **Max 255 bytes** | | string | | N/A | |  |

**ItemDonation Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** | |
| **Email** | Contact Information for the guest donating. | **Max 40 bytes** | String | Primary Key | |
| **FoodDonation** | Food that is donated to the cupboard. | **Max 255 bytes** | String | N/A |  |
| **NonFoodDonation** | Items that are not food donated to the cupboard | **Max 255 bytes** | String | N/A |  |
| **PerishableID** | Determines if a FoodDonation is perishable or non-perishable. If perishable, this is used to find an expiration date | **2 bytes** | Boolean | N/A |  |
| **ExpirationDate** | The date a food item expires | **Max 16 bytes** | String | N/A |  |

**MonetaryDonation Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **MonetaryDonation** | Money donated to the cupboard. | **Max 40 MB** | Long Unsigned Int | N/A |
| **Email** | Contact Information for the guest donating. | **Max 40 bytes** | String | Primary Key |

**Dynamic Calendar Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **Event Name** | Name of an event | **Max 255 bytes** | String | N/A |
| **Event Description** | Describes the event | **Max 255 bytes** | String | N/A |
| **Event Time** | Time event starts and ends | **8 bytes** | int | N/A |
| **Event Date** | Day(s) event is on | **Max 255 bytes** | String | N/A |

**Drop Box Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **Student ID** | The ID U of L gives to each student. | **7 bytes** | Long unsigned Int | **Primary Key** |
| **Student Email** | Student’s email | **Max 255 bytes** | String | N/A |
| **ItemsSelected** | The items a student can select to pick up | **Max 255 bytes** | string | N/A |

**Compost Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Attribute Definition** | **Attribute Size** | **Attribute Type** | **Primary or Foreign Key?** |
| **UserID** | The ID of staff Member | 7 bytes | Long unsigned Int | **Primary Key** |
| **CompostQuantity** | The amount of compost created | Max 20 bytes | Long unsigned int | **N/A** |

 System Architecture

The system we envision consists of 5 components that the user’s input is put through. The first system is the design viewpoint, which describes the system architecture we need, and the second viewpoint is the realization viewpoint, which describes the system architecture we will use. For cloud storage, we will use Microsoft OneDrive, which is provided by the university. The Database server we will use is SQL. Server (also provided by the University). The application server we will use is Microsoft azure, and finally, the content management system we chose is Joomla, which is an open-source CMS.

*Design Viewpoint*

Diagram

Description automatically generated

*Realization Viewpoint*

Diagram

Description automatically generated

Design Procedures for Security Concerns and Non-functional Requirements

The procedures we need to take to fulfill for the nonfunctional requirements are:

* Connect to University of Louisville Firewall.
* Implement a SQL server database
* Implement the Microsoft Azure application server.
* Implement the Microsoft OneDrive cloud server
* Create the website with the Joomla content management system

The most important nonfunctional requirement is the ability to keep the system secure. This is fulfilled by the University of Louisville firewall, as well as the Microsoft OneDrive cloud server. The firewall will protect the data from unauthorized access, and OneDrive will allow the safe storage of data. In addition, OneDrive is much more secure than Google. We also need Microsoft SQL server to store our data. We need to redesign the website to be more secure. For this we recommend the Joomla content management system. This CMS is good because it is both free, and highly secure. Finally, we would need Microsoft Azure as our application server.

During the design phase, we can easily address each requirement. Firstly, we would need to create the website itself using the Joomla content management system and implement the SQL server to it. Next, we would connect the website to the University of Louisville firewall.  Finally, we would implement Microsoft Azure. During detailed design phase, we need to start with the essential tasks, and follow the order described above.

Chart

Description automatically generated

User Interface Prototypes

The following images are high-level prototypes of what the system must look like in order to complete each use case.  These prototypes do not connect to any databases, and only reflect the minimum of programming. **Due to the backup system, and restore system use cases requiring external programs, no diagrams need to be made for them.**

*Landing Page*

A picture containing text, indoor, screenshot

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

The Landing Page allows the user to choose from the tabs “About,” “Donate,” “Services,” and “Resources. The Landing Page contains a description of the cupboard, information on hours of operation, location, current Covid-19 guidelines, composting statistics, additional resources, and contact information.

*Member Login Page*

*Graphical user interface, application

Description automatically generated*

The member login page allows the user to either login as a volunteer through the staff login or sign up if they are not already volunteers through the student login, or find more information if they are a parent or guest user.

*Member Page*  *(Scheduling)*

*Calendar

Description automatically generated*

*The scheduling page allows users to view their volunteer schedule and track their logged hours.*

*Inventory Page*

*Graphical user interface, application

Description automatically generated*

*The Inventory page shows information on the cupboard’s inventory.*

*Finances Page*

*Graphical user interface, application

Description automatically generated*

*The Finances Page shows information on the cupboard’s finances.*

*Submitted Applications Page*

*Graphical user interface, application

Description automatically generated*

*The submitted applications page allows users to add, modify, or deleted applications.*

*Events Page*

*Graphical user interface, calendar

Description automatically generated*

Calendar

Description automatically generated

Graphical user interface, application

Description automatically generated

*The events page allows users to get in contact with RSO organizations and check student events.*

*Submitted Recipes Page*

*Graphical user interface, table

Description automatically generated*

*The Recipes Page, navigated through the student login, allows for recipes to be viewed, added to “Accepted Recipes,” “Saved Recipes,” and “Rejected Recipes.”*

*About Us Page*

*Graphical user interface, text, website

Description automatically generatedGraphical user interface, text, application, Word

Description automatically generated*

Graphical user interface, text, application

Description automatically generated

*The About Us Page provides information on the cupboard’s background, mission statement, vision statement, and other information, as well as contact information.*

*Donation Page*

*Graphical user interface, website

Description automatically generated*

*The Donations Page allows users to donate through the Amazon Wishlist, the student affairs fund link, or view information on the Drop-Off Box.*

*Grab and Go Page*

Graphical user interface

Description automatically generated

*The Grab & Go page allows student users to submit requests for item pick up from the Grab and Go program.*

*Recipes Page*

Graphical user interface

Description automatically generated

*The Recipes page allows users to submit suggested recipes based on a dynamically updated itemized list of inventory.*