SPRINT 1

Details:

Started in Fall 2016, the Rutgers Student Food Pantry is one of the over 20 food pantries in New Brunswick. It exclusively serves Rutgers students and is located at the College Ave Student Center. The pantry is stocked by donations from Rutgers Against Hunger, Replenish, and private donations.

Point of contact:

- Nick Lluen (worker at the Food Pantry, and member of this group)
- Food Pantry Coordinator: Barbara Blackwell (Nick's boss)
- Ross (Nick's co worker)

Problems:

- Slow sign in times at arrival of food pantry
- Not able to account who has already been to the food pantry that week
- No efficient way to keep track of inventory in the food pantry
- No scanner to connect to inventory
- Focus on optimization for student experience
- Not many people know about the food pantry's existence

Capacity:

- Only limits customer base to Rutgers students (37,364 students)
- 30-70 daily customers

Projected Tech Stack:

- SQL for database of attendees
- Better data management for food pantry
- Frontend (ruby on rails? Or ReactJS TBD) with simple UI
- Must be accessible by RU students only!!
- Python Back End
- Group Python skills are better after minor project
- Qualtrics sign-up form

A. Requirements Engineering

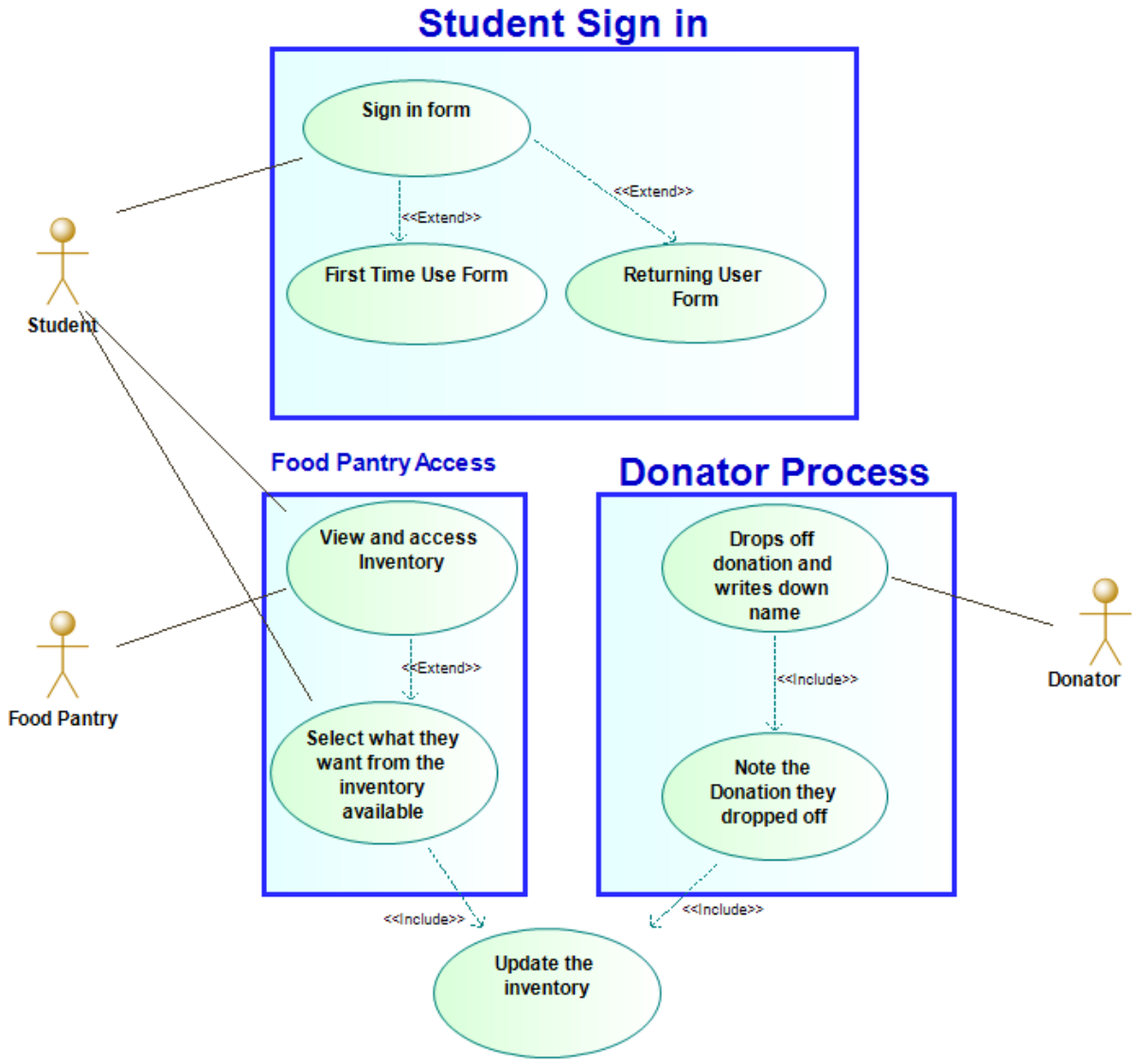
Non Functional Requirements:

- Be able to service the various users interacting with the food pantry this includes but is not limited to:
    - **Donors**: people who donate food to the food pantry, is not limited to just one person or just one food item donated
    - **Employees**: the people who work at the Food Pantry
    - **Food Recipients**: RU students that are taking out food out of the pantry
- Easy and fast process for donors to log their donation.
- Must be working and reliable at least 99% of the time.

Functional Requirements:

- Inventory should be accessible by both Employees and Students, but not necessarily Donators.
- Verify that the person taking food is a Rutgers student
    - Has valid netid and associated password
- Each time food is taken out of the pantry, the recipient must log what is taken out, this should update the inventory in the database.

B. System Modeling



System	Food Pantry System
Use Case	Sign In to Food Pantry
Actors	Student
Data	The user provides their netid and password
Stimulus	The user hits the enter button and the system validates whether the netid and password is valid by checking the Rutgers student database.
Response	The system redirects the student to either a first time form if it is the user's first time here or confirms that the user has logged in.

Comments	The user's information is validated from a mock Rutgers student database that was created by our team.

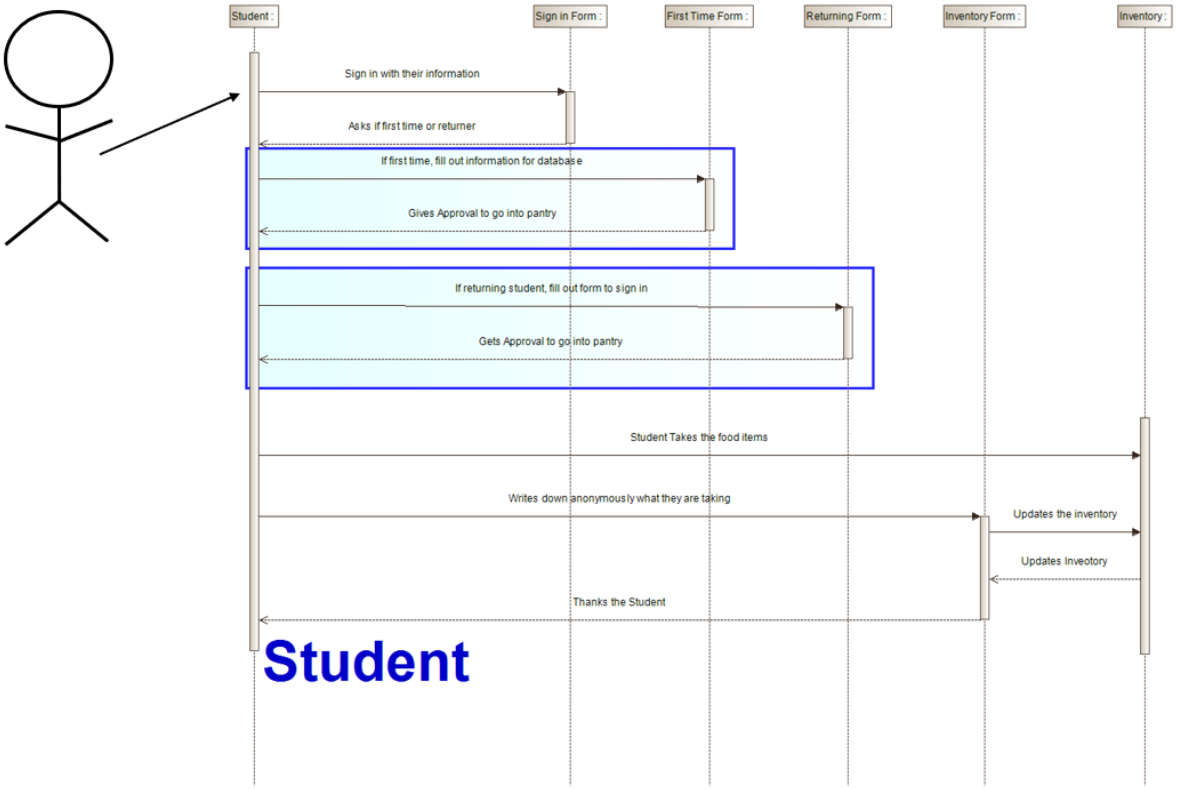
System	Food Pantry System
Use Case	First Time Use Form
Actors	Student
Data	The user provides their netid and password
Stimulus	The user hits the enter button and the system validates whether the netid and password is valid by checking the Rutgers student database.
Response	The system redirects the student to either a first time form if it is the user's first time here or confirms that the user has logged in.
Comments	The Food Pantry made it clear that for their business needs it is very important to collect the information of first time student

System	Food Pantry System
Use Case	Donation Form
Actors	Donator
Data	The user provides their information such as name, email, phone number, donation, donation quantity, etc.
Stimulus	The user hits the enter button and the system validates whether the netid and password is valid by checking the Rutgers student database.
Response	The food pantry database takes the survey's input and stores them in the appropriate table in the database, updating the number of items in each food category
Comments	The donor's information is stored in the database as part of the business needs of the Food Pantry.

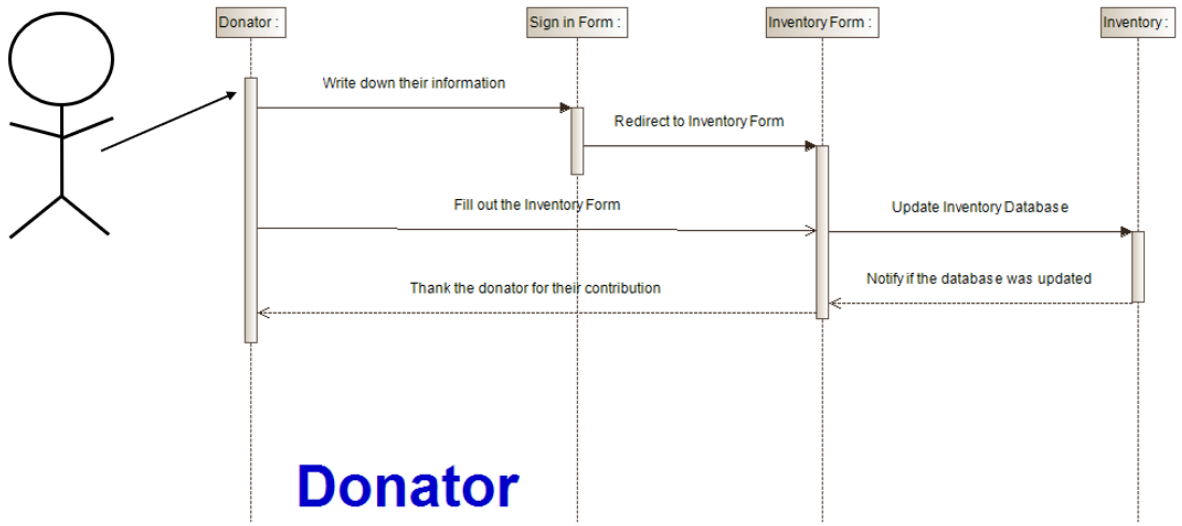
System	Food Pantry System

Use Case	Inventory Check
Actors	Food Pantry Employee
Data	The User checks the database of the inventory
Stimulus	The user will want to check how much food and items are available at the end of the day by checking the database of the inventory.
Response	The database will display the current amount of each category
Comments	Our database is hosted through a local host server

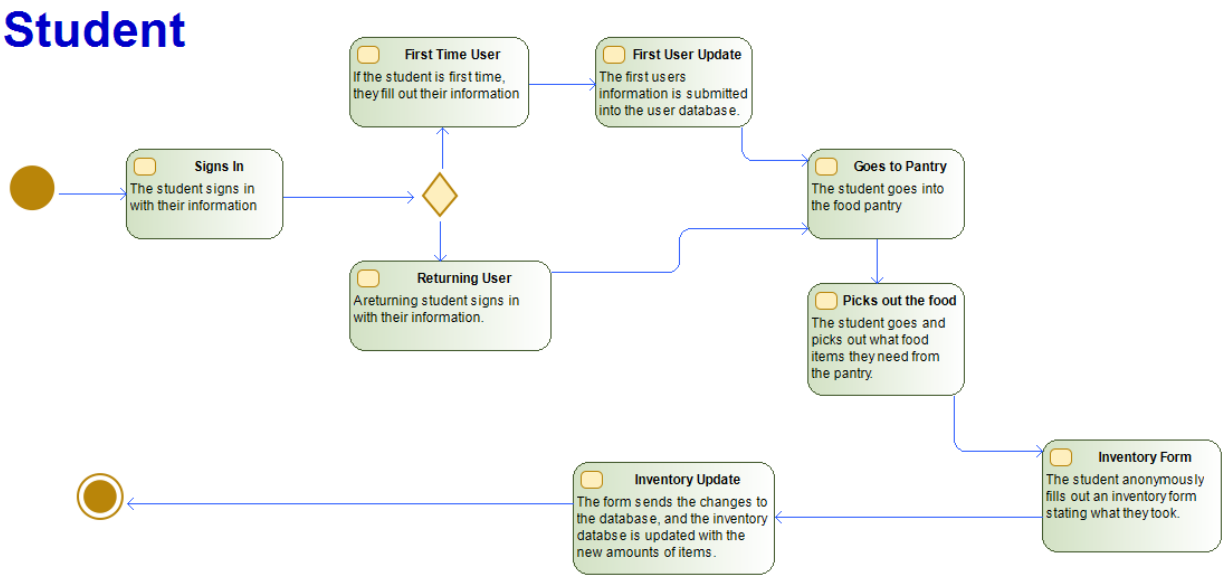
C. Architectural Design



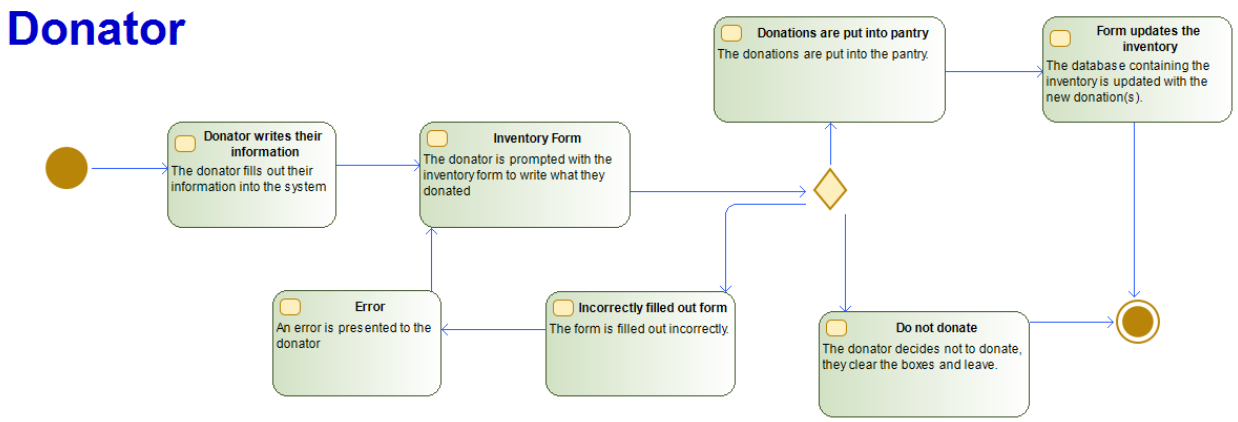
Student Sequence Diagram



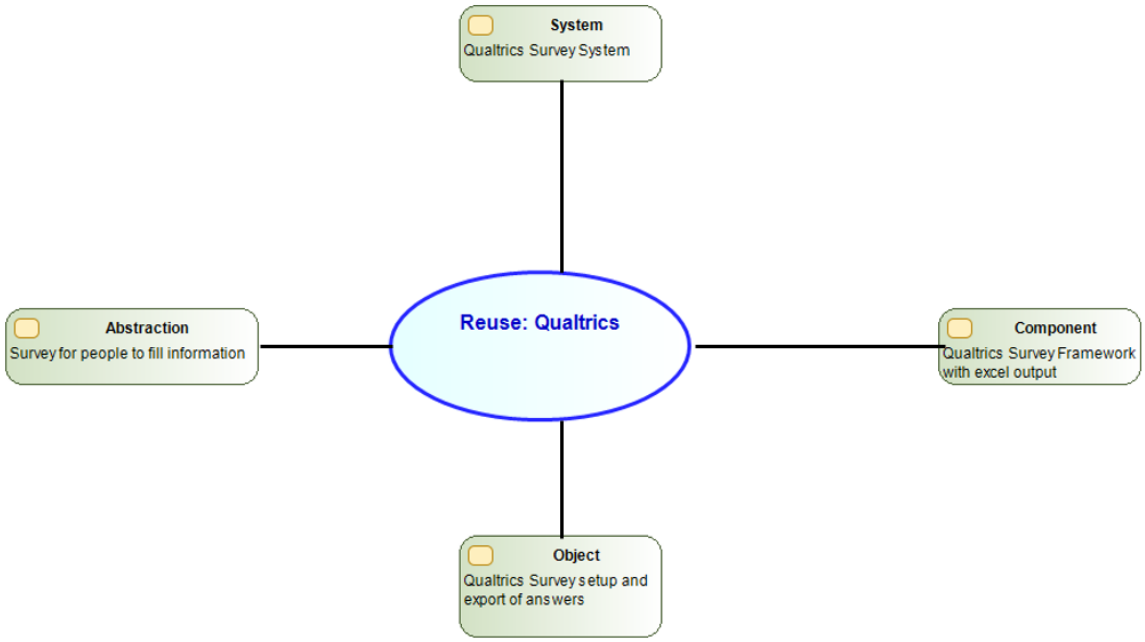
Donator Sequence Diagram



Student Activity Diagram



Donator Activity Diagram

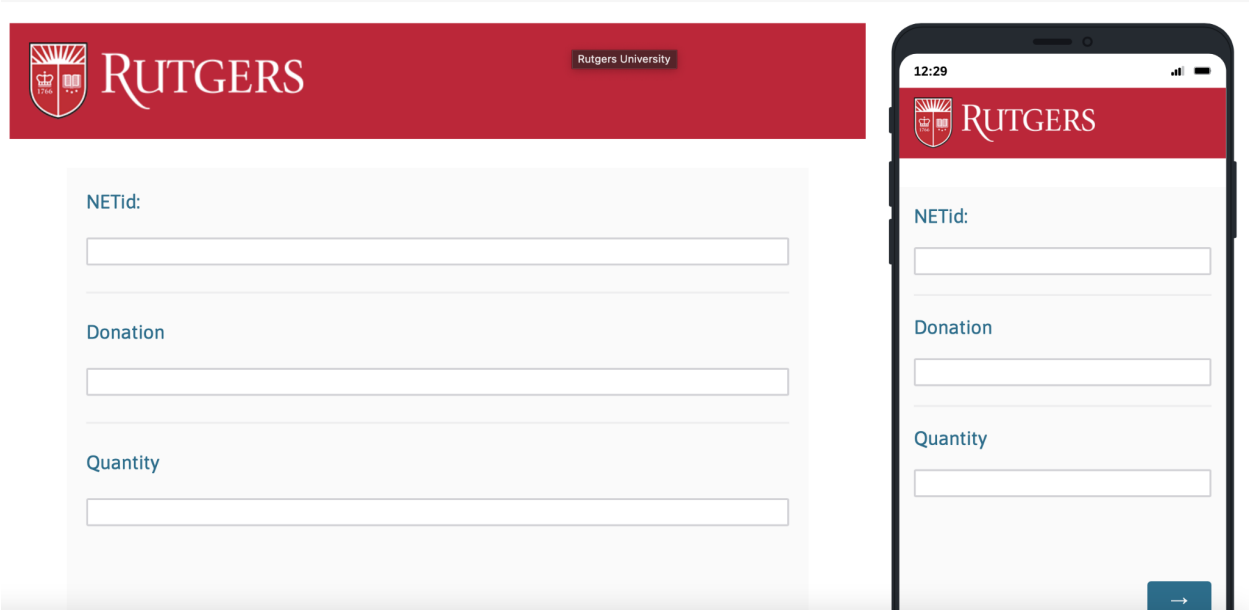


Qualtrics Software Reuse

D. Design and Implementation

The MVP for this sprint is a form which asks for certain information from the user relevant to the goal of software. It first asks for the user to enter their name and netid. Next, it will ask for the user to input the name of what they wish to donate. The final text field will be for the user to enter the quantity which they would like to donate. This was implemented via a qualtrics form. The output of this form is an excel spreadsheet containing the inputted information.

Due to constraints in time for this sprint, it doesn't cover how to delete items from an inventory, in the case that a student takes something from the pantry for instance. Below is an image of the screen in which a user will enter the aforementioned information.



E. Software Testing (all levels)

	Test 1	*Test 2*	*Test 3*	*Test 4*	*Test 5*	*Test 6*	*Test 7*
Name	Joe Shmo	Eric Mayer	Rick Jones	Rakim Long	$$$	Wilhelm Brown	Cherry Peters
netid	js3	em241	rdj73	rkl430	bjm12	(90)	cnp400
Donation	potatoes	applesauce	beans		cereal	pasta	fru32it
Quantity	3	-1	4	3	aa	1	
Expected Output	Successful submission	Unsuccessful	Successful submission	Unsuccessful	Unsuccessful	Unsuccessful	Unsuccessful
Actual Output	Successful submission	Unsuccessful	Successful submission	Unsuccessful	Unsuccessful	Unsuccessful	Unsuccessful

F. Evaluation

Verification

To verify a program is to confirm that it meets its specification. In order to do this for our Sprint 1 MVP, we ran the product against multiple test cases which tested the response of each field to a number of different inputs. We used inputs that should be allowed by the form as well as inputs that shouldn't. We took advantage of the fact that the form takes multiple inputs by using a mix of acceptable inputs and unacceptable inputs; we did this to verify that the form only successfully submits when all inputs are considered acceptable. Some of the test inputs we used are not expected to actually be used in the form's regular lifetime, but were used to verify that it meets specifications.

We tested the form with certain inputs for the netid field to ensure that a valid netid is being entered to verify a Rutgers student is accessing it. We also verified the donations field by ensuring that inputting unacceptable values would not allow the form to be submitted. Furthermore, we tested the donation quantity field with non-positive and non-numeric values to verify that the form would not allow such inputs.

Validation

Making sure that the program performs to a standard that is acceptable for the user and that the requirements of the program match what the user needs is validation, and this was done extensively for this MVP. We did this by testing the program with inputs we expect the form to typically receive from users and by validating that it performs as the user would expect. To validate the form, we checked to make sure that entering and using the form is easy and seamless. We also ensured that it was working the majority of the time, among other metrics.