Planning

Team Name: Lechẽ

Team Members:

- O Moo Gay: Team Leader, Business Analyst, Project Coordinator
- David Thompson: Security Engineer
- Caden Bonnell: UX-UI
- David Reveles Hernandez: Database Architect/Network Architect
- Chris Gaona: Software Developer/Web Developer

O Moo will assign tasks to group members and everyone will get some sort of tasks to work on every week. Everyone will submit something to O Moo two days before the assignment is due. He will read them over and put the submitted documents together. There is a discord set up for the team to communicate. If they have any problems, they will report to the team and the team will work to resolve it.

Project/System Request

The sponsor of the project is O Moo's Aunt Nay Blu Moe. Nay Blu Moe runs a small Asian grocery shop called "Asian Store". Her products are mostly for the Thai and Karen people. She has new Thai/Karen products coming in every week and she needs a website where she can upload pictures of the product. Her fellow Thai and Karen people should be able to view what products she has every week. Nay Blu Moe needs a simple admin page where she can upload pictures of her products. She wants to be able to list prices and show how many of the products she has; an up-to-date inventory however, this is not a shopping website, it is a place

to just view what is available in her grocery shop. The customers that will be using this website will mostly be older Asian ladies who are not very familiar with technology. Nay Blu Moe's customers need a simple user interface where their customers will be able to navigate through the website without confusion. Widgets like buttons, search boxes, and drop-down menus need to be clear.

Requirements/ Value

The website must have a responsive web design for it to look good on different devices. There must be an image for each product, along with its name, price, and the quantity of the product. The store name must be present in the app, as well as the store phone number. There needs to be a search bar to do a quick look up. This app would be a good addition to the shop operation. The customers will not have to make a random drive to see what is available. This is especially important if the customers live far away since this place is the most accessible to Thai/Karen products in town. This app should help Nay Blu Moe keep track of her product every week. This is not an inventory app, but will simply show Nay Blu Moe which products he is selling. Nay Blu Moe should be able to see how many of her products are selling. There also needs to be a notification system to notify Nay Blu Moe to upload pictures of her products every week.

Constraints/ Level Of Effort

When it comes to constraints, the app should not be a shopping app or an inventory app. It is just a viewing app. The app should be simple because the people that will be using this app are older people who are not very good with technology. The app will be developed as a web

app which should not cost much money to develop. The creator of this app should be able to find free software tools to be able to make this app. There is a risk when using free software tools because it might not be that secure and fast. However, if Nay Blu Moe wants more security and more speed, she could have the developers look for something that is more secure and that will run faster which will cost more money.

Feasibility Analysis

This application will be a website where the interface is simple due to the age of customers and experience customers have with technology. Nay Blu Moe can have the developer use the free package that is offered to make the app or upgrade to a premium software to make the app. This app might increase Nay Blu Moe's revenue but it is not clear. There are intangible benefits if this web app is added to the store. It shows that Nay Blu Moe cares about her customers by letting them know what she has without having to make trips. But then there are intangible costs. If Nay Blu Moe gets lazy and does not update the app weekly, the customers might get angry and not use the app. For organizational feasibility, if the app is designed well and is not confusing, and if the owner updates the app weekly, the app would be good for the store. The customer would make use of the app.

Project planning

We will use the waterfall process method to make this app. We have our requirements set as user stories, and we may need to add more requirements later on as the project

progresses. Once the website is presented to the customer there is a good chance the customer might want to add something or want us to make changes as we go. Even though this is not agile we will still do some sort of iteration to check our work. This will be possible with the creation of user stories that will correspond with what the admin and users want on the app. Then make wireframes for each user story, then workflow, then code the program. After that we will run the app on pythonanwhere.com. The first phase to get everyone on github. Make sure everyone knows how to push codes, create pull requests. Then make sure everyone has django and visual studio code installed. Then we can start coding. Here, team members will be given tasks. Templates will be created. One of the members will be responsible for making the templates look good. This person will work with css and bootstrap. The functionality will also be implemented. After creating the functionality, the team will move on to hooking up all the functionality to the templates. Then we show the product to the owner. If changes need to be made, that will get done.

Analysis

Business Requirements

Functional Requirements:

1. Admin Management

    1.1 The system shall allow the creation of an admin user.

    1.2 The system shall allow the admin user login and logout.

2. Item Management

    2.1 The system shall allow the creation of products.

    2.2 The system shall display the products that admin users create.

    2.3 The system shall allow the deletion and update of products.

    2.4 The system shall send notifications to the admin users to update the products.

3. User Item View Management

    3.1 The system shall allow the user to search/filter items available.

    3.2 The system shall display available items that the user searched for.

Nonfunctional Requirements:

1. Operational

    1.1 The system should be web-based.

    1.2 The system should run on any web browser on any device.

2. Performance

    2.1 The system should update the website when new products are added/updated/deleted in less than 1 minute.

3. Security

    3.1 The admins user account should be secured.

    3.2 The admin user should only be allowed to make changes with items.

High Level Overview of System

- Supply an accessible service for small businesses to use and benefit from
- Make the shop owning experience more efficient to operate
- Provide a learning experience for developing as students

Use Cases / User Stories

1. As a customer I want to be able to view the different products available
2. As a customer I want to be able to see the price of the product
3. As a customer I want to be able to view a picture of the product
4. As a customer I want to know how many products are there
5. As a customer I want to be able to filter through products
6. As an owner I would like to be able to add products to my webpage
    a. As an owner I want to be able to add prices
    b. As an owner I want to be able to add pictures
    c. As an owner I want to able to add the quantities of products I have
    d. As an owner I want to be able to delete prices, pictures, items
7. As a customer I want to be able to search through the products
8. As a customer I want the app to be satisfying to use

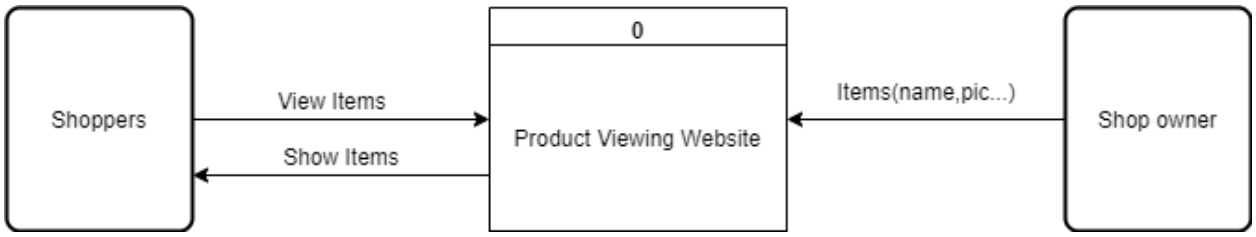
Interview Questions

1. How much time are you willing to spend updating your inventory per week?
    a. One hour is the amount of time the owner wants to spend on updating the website, but it's alright if it takes more time
2. Should this primarily be a mobile or web app?
    a. It should be a website, but it would be good if a mobile app can be implemented.
3. What pages would you want to prioritize?

    a. The page the customer will interact with
4. What theme would you like?
    a. Dark and light mode
5. What do your customers consist of?
    a. There will be older and younger people. There are not many worries for younger people but the designer should keep in mind that there will be quite a bit of older people that aren't good with technology will be using the website
6. How much traffic are you expecting?
    a. It shouldn't be overwhelming. There are probably 200 to 300 customers every week.
7. What is your budget?
    a. I want to spend zero dollars if i can
8. How many products would you add roughly?
    a. It should be around 100 items. And the items will be updated every week.
9. What type of results would you like to prioritize coming from the users searching items? (name, price, etc)
    a. There should be images, name's, and price's of the items.
10. Do you want an about page?
    a. yes , it should also have the store contact info
11. Do you want the app to be in Karen or Thai?
    a. Not sure/ if possible, Karen would be prefered
12. Do you want a contact page?
    a. Yes, that would be very helpful

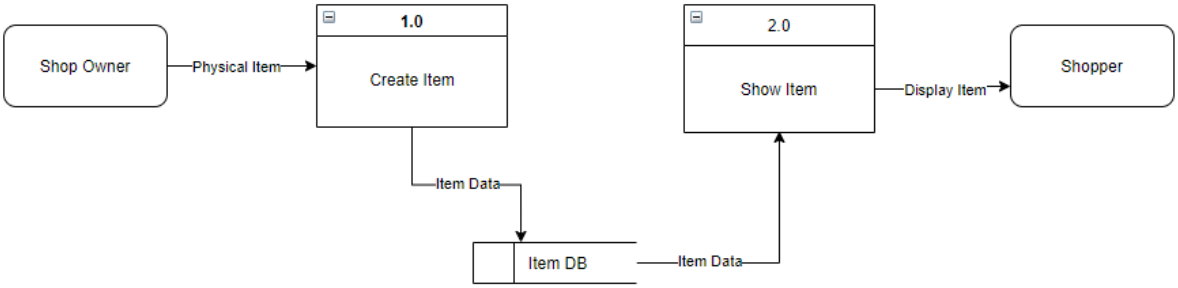
Process Model

DFD(Context diagram)



This is the Context diagram. It has three different parts. The first part is the shop owner. The shop owner will be able to upload the items that arrived. The name, price, image, and number of quantities should be uploaded as well. Then that gets sent to the website. This website will format the data uploaded by the owner and then it will be displayed to the shoppers.

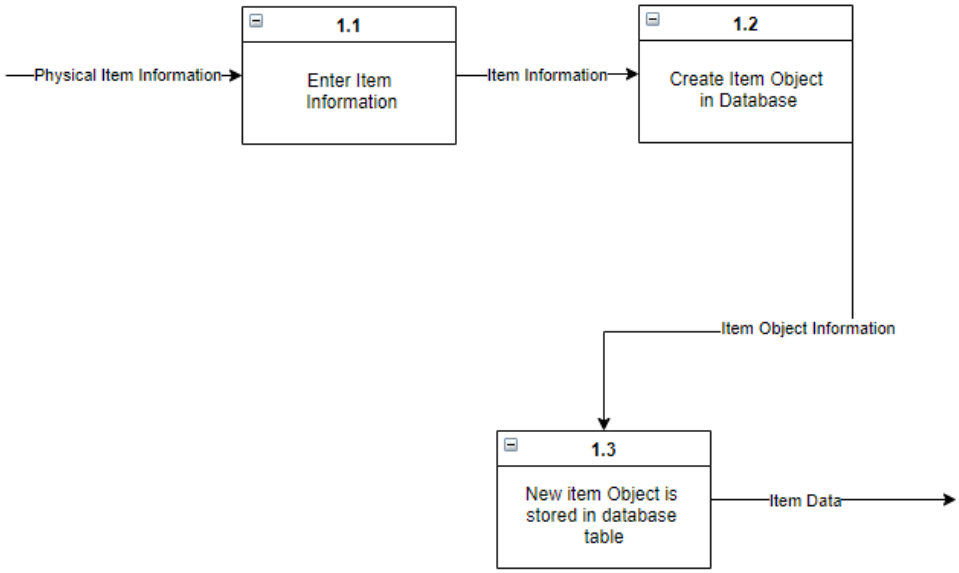
DFD(Level 0)



This is the level zero of the data flow diagram. It shows the main entity and the main process that occurs. There will be a shop owner that will upload items with all its attributes. Then that gets sent to the website. The website has two main processes, which are the create item and show item. The websites will create the items and send that to the database. Then the website will display the item. The shopper should be able to view the products.

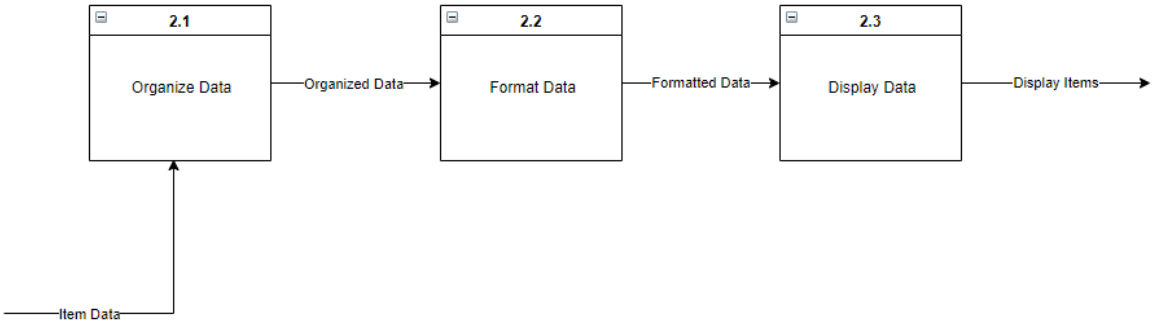
DFD(Level 1)

1.0



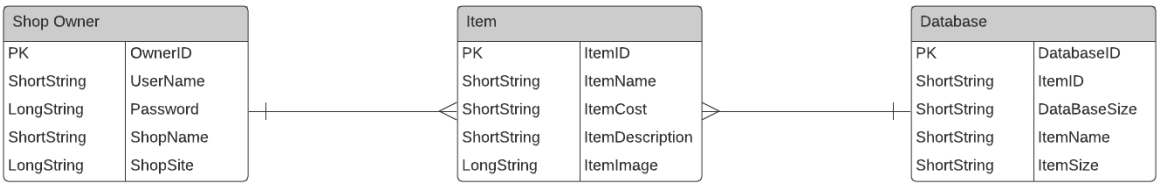
In this portion of the DFD level 1 diagram, we see how data is transformed from a physical item, to data that can be stored into the web page's database. We start from the beginning where the owner enters the item information of what is in stock. The data entered is created into an object in the database which can then be used to display onto the webpage. Once the data object is created, it can be stored and updated within the database as well as be used for displaying onto future webpages in development.

2.0



This Level 1 DFD is breaking down the ‘Show Item’ (2.0) process from our Level 0 DFD. The data starts as raw item data from the database. The raw item data is then organized into the structure it will be displayed on the user’s screen (2.1). Next this organized data is formatted so it looks nice (2.2), and finally the data is displayed to the user in the organized, formatted way (2.3).

Data Model Diagram(E/R Diagram)



For the entity relationship diagram, the main concepts that come across for translating the website data have to deal with the Shop Owner, the Items that the owner decides to add and a Database to host all of the information to display from backend to frontend for the end user. The owner will submit information in order to host their ecommerce shop such as a UserName, Password, ShopName, and ShopSite link as well as a primary key for the OwnerID which would be given by the system. The Shop Owner will have a 1:N relationship with items due to the fact that an owner can create many items but each item can only have 1 owner. The Item information that will be needed in order to form that instance consists of ItemName, ItemCost, ItemDescription and an ItemImage link that can map a photo for the Database to grab and display. The Item entity will also have a generated primary key for ItemID. Finally, the Database entity would also have a 1:N relationship with the Items due to 1 database hosting all of the items the Shop Owner generates. The Database information will consist of its DatabaseID as the primary key, foreign keys of ItemID and ItemName, DatabaseSize, and ItemSize. These entities should give a basic starting point to expand the overall project and let any user create their own personal ecommerce business.

Design

For our project, we will be designing and building a website for a local grocery store owner which will display the items that are currently available for sale. The software development process we will be using is the Waterfall model paired with the Trello service to keep track of the progress we have been making. The website will be built using python's framework Django, Bootstrap, and HTML and will be hosted on the PythonAnywhere servers.

The website will be a simple website that will consist of having the shop owner upload items into the system's database by first signing in to the Django administration database. Once the shop owner is signed in, they will be able to create items on the database tables, which tables consist of Shop Owners and Item, and from there the system will be able to display the item onto the system's UI. The system will organize and format the data that was entered into the database by the shop owner and then display the data correctly for the customers to view.

The goal for building this website is to reduce the number of trips a customer takes to the shop by letting them know exactly what is available in the shop in real time.

Members of Team Leche will be given one or two tasks every week. And every week The team members will push some kind of code to the GitHub that was created for this project until the whole project is done. Every team member should know how to use Git. The ones that do not know how to use Git will be taught the first week of class. Then the team will move onto making the project and assigning tasks. There will be a meeting every week on monday to make sure the team member knows what their tasks are and how they must go about getting that done. Any question or confusion that the team members have will be addressed during these meetings. These meetings could also happen with the professor to let the professor know where the team is on the project.

O Moo will be in charge of assigning tasks and making sure everyone has something to do each week. He will be responsible for keeping track of how the project is doing and keep track of the

amount of work left. There will also be some kind of burndown chart to help keep track of the due dates and how much work is left.

Caden will be responsible for making the website look good. So he will be working with CSS and Bootstrap.

Chris will make the templates that the websites will need. So he will be responsible for creating html pages. He will also create the views that are needed and hook up the urls. The CSS and Bootstrap should be updated every week along with the templates created.

David Reveles Hernandez will be responsible for creating the database and the fields that are needed for this project. Django comes with a SQLite database and SQLite will be used for storing data.

David Thompson will be responsible for deploying the Django project on python anywhere server. David will also test the website and make sure all the functionality is working how they are supposed to. He will be responsible for making sure the website is acceptable to the customer and the owners.

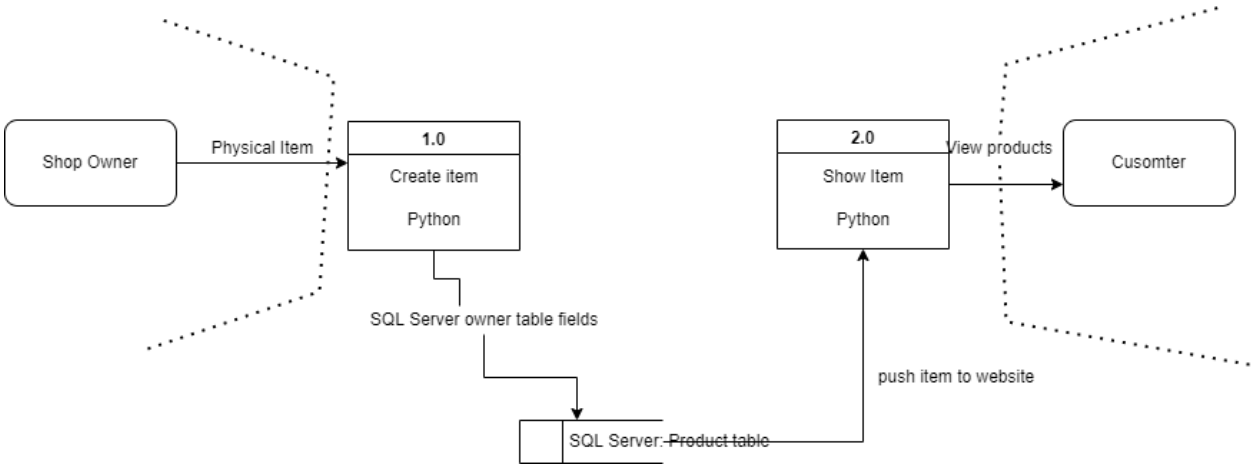
List of Technologies:

- Django web framework
- Python programming language
- JavaScript programming language
- HTML markup language
- CSS style sheet language
- Bootstrap front-end framework
- Git version control software
- PythonAnywhere PaaS web hosting service
- SQLite relational database management system

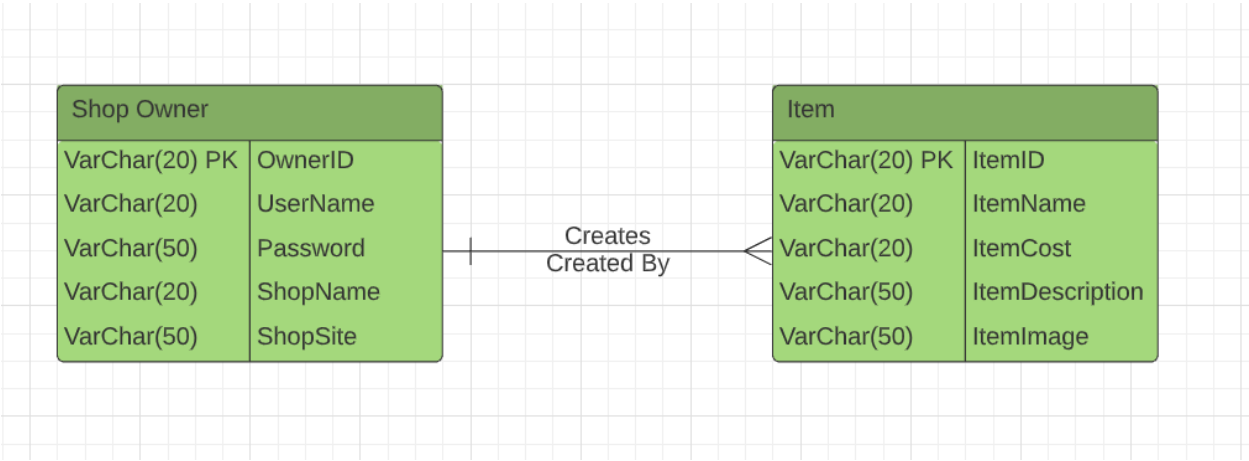
For our website we will be using the Django web framework to create it. The programming languages that will be mainly used are Python, JavaScript, and HTML. Bootstrap will be used for our front-end framework, which uses CSS and JavaScript. SQLite will be used for the database since Django has it configured by default and is easy to use. For hosting our website we will be using PythonAnywhere and using it will have us not worry about having computer hardware to create a server. The team will be using GitHub for version control of the project.

Physical DFD

Physical Process Model - Local grocery store



Physical ERD



Implementation

Team Leche Github for the project

cgaona1/La-Leche: Systems analysis and design group project. (github.com)

Members will be assigned some sort of coding task every week and this is where the code will be pushed.

Work Assignment

O Moo will be assigning tasks to other members. Trello will be used to help keep an eye on the tasks that have been assigned to everyone. Members will update their status on the Friday of every week. There will also be some sort of a burndown chart to keep track of the time we have left and the amount of work we have done. A burndown chart will also give us a clear picture of if we are going to finish the project on time or if we are behind our set release date.

Testing

Every team member will do some sort of unit testing. Every method they create, they must test to make sure it works correctly. David Reveles will be in charge of the integration test. He will make sure the app will be able to run on the python anywhere server. Then there will be the acceptance test. Acceptance will be with the shop owner and O Moo. O Moo will make sure everything that the shop owner wants is present. He will make sure the owner can use the website as intended. The owner should be able to sign in, upload items and show it to the customers. The owner should also be able to see what the customer sees.

Documentation

Caden Bonell will be in charge of documentation. The type of documentation that will be done will be a tutorial. Tutorial documentation will be most helpful to the customer and will be in video format. Caden will make videos detailing how the customer will be able to use the website. There will be no more than five videos showing the customer how to use the website and each video will be no longer than six minutes.