

Listify Software Design Specification

Contributors

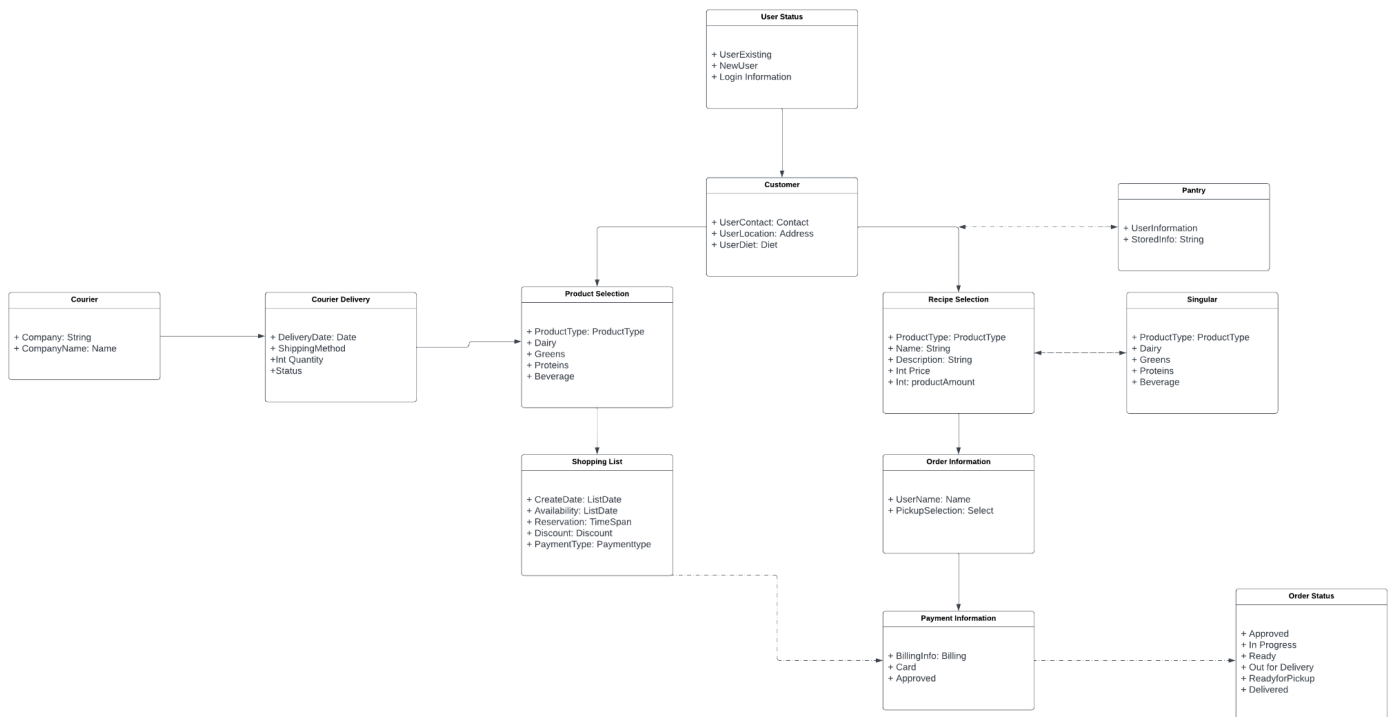
- Jaydon Eppinette
- Joshua Constine
- Agustin Munoz
- Marc Rodriguez

System Description

Listify is the best way to organize the mundane task of grocery shopping. Listify empowers our users by simplifying the process of planning their weekly meals. Using our extensive database of recipes, we quickly compile a list of the required ingredients. Users can also manually add items to the list. Additionally, Listify enables users to share their shopping list via a quick text message.

With Listify, grocery shopping becomes efficient and hassle-free.

UML Diagram



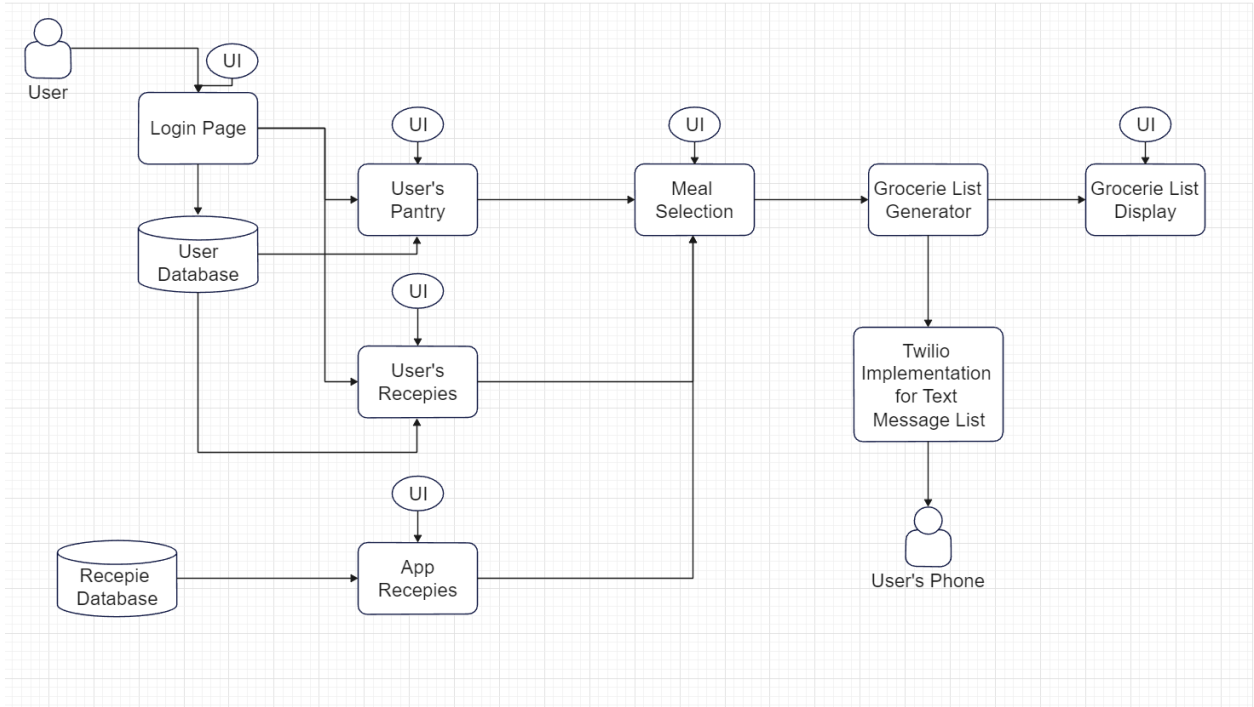
General Description of Classes

Classes	Descriptions	Attributes	Operations
User Status	<i>This class pertains to the general status of the user. Holds the information in relation to the registered account.</i>	<i>UserExisting and NewUser pertain to the status of a user in regards to their account. This is tied to Login Information.</i>	<i>The operations describe whether the user has existing information or needs to register new data.</i>
Customer	<i>Holds general customer information in relation to the previous class.</i>	<i>User: Contact, Location, and Diet are all key to user information and the specific needs of the customer. The diet portion is relative.</i>	<i>Operations include the input of information related to personal clientele data.</i>
Pantry	<i>Stores information in relation to the user's current items in storage.</i>	<i>UserInformation and StoredInfo pertain to current data related to previous client input.</i>	<i>The operation is performed in relation to currently held data from previous input.</i>
Recipe Selection	<i>Content within this class is in relation to options users are allotted to recipe information.</i>	<i>ProductType, Name, Description, and related integers are attributed to certain recipes and their cost.</i>	<i>The operations within this class hold data in relation to recipes available to the user and their cost.</i>
Singular	<i>Class holds information to single items that the user may add a la carte.</i>	<i>Attributed to single items and their categorization.</i>	<i>Operators are to hold single product information rather than bundled recipes.</i>
Order Information	<i>This class holds information in relation to general pick up.</i>	<i>UserName pertains to the name of the client and PickupSelection pertains to where the order is to be carried out from.</i>	<i>Limited to crucial information in terms of data relating to order status available to users.</i>
Payment Information	<i>Stores the payment information of customers and general clientele.</i>	<i>BillingInfo holds general billing information, Card holds type of</i>	<i>Data in relation to billing and status of payment.</i>

		<i>payment in relation to cards, and Approved signifies whether the transaction has successfully gone through.</i>	
Order Status	<i>Updated class in relation to all order information. Should be updated in relation to order status.</i>	<i>The attributes in this class hold a general step by step process in terms of the order's timing and status available to the client.</i>	<i>Data operators in relation to status of the order at hand.</i>
Product Selection	<i>A class containing general a la carte items sorted by type of product.</i>	<i>Similar to the attributes found in the Singular class given categorization and a la carte items.</i>	<i>Operators are in relation to types of singular products and their categorizations</i>
Shopping List	<i>Contains user's selected products.</i>	<i>CreateDate: Holds date to current standards Availability: Dictates the item's ability to be purchased. Reservation: Serves as a time of purchase. Discount: Applicable coupon or discount via promotion PaymentType: Form of payment</i>	<i>Operations are attributed to the availability and status of the user's intended products to purchase.</i>
Courrier	<i>Dependent on the courier delivering.</i>	<i>Company and CompanyName relate to specific couriers.</i>	<i>Operators in relation to certain or specific courier.</i>
Courier Delivery	<i>Holds courier delivery information</i>	<i>DeliveryDate: Date of delivery of items ShippingMethod pertains to method of shipment Integer of Quantity dictates amount of certain product Status is for</i>	<i>Operators are in tune with specific courier delivery information.</i>

		<i>estimation of delivery.</i>	
--	--	--------------------------------	--

Software Architecture Diagram

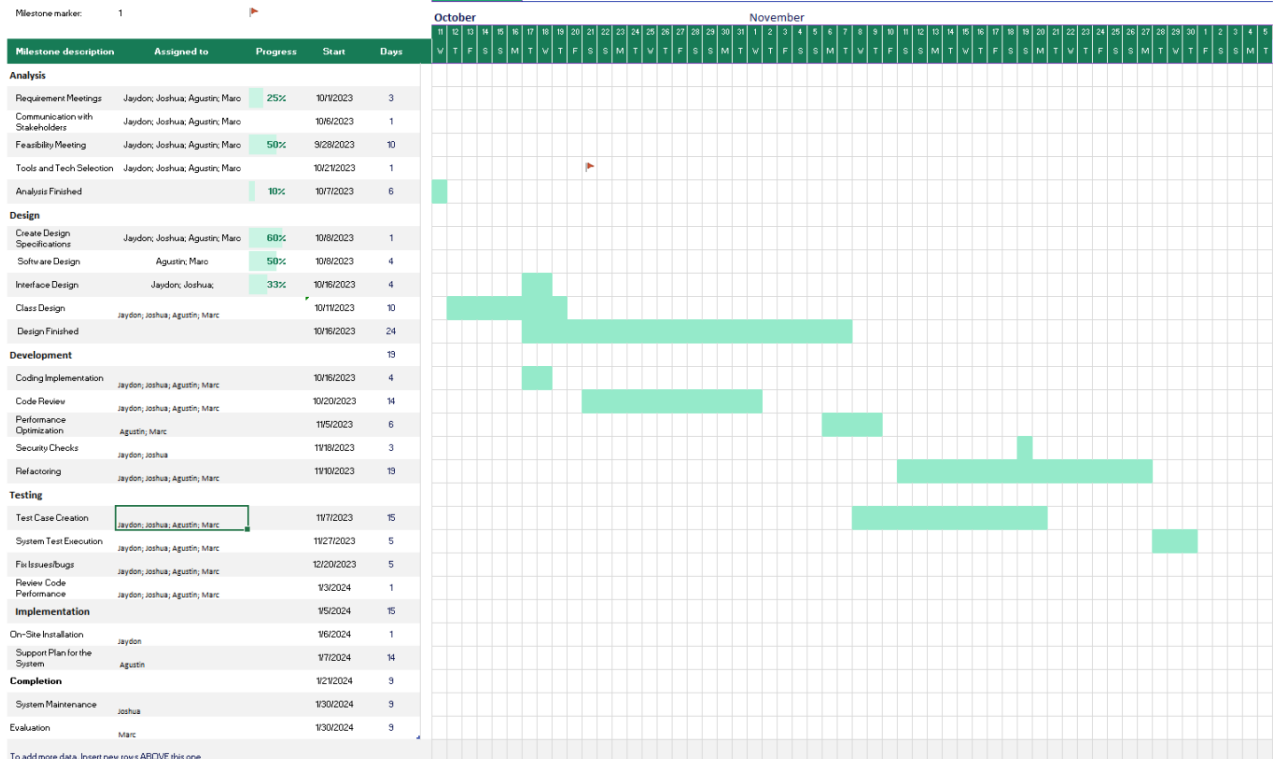


Jaydon Eppinette, Joshua Constine, Agustin Munoz, Marc Rodriguez

Project start date: 9/28/2023

Milestone marker: 1

Scrolling increment: 13



Test Plan

Equivalence Classes:

userStatus(Bool)

-Is the user registered? Yes, No

userName(String)

-1:16 Alphanumeric characters

userPassword(String)

-6:16 Alphanumeric characters

pantry(food)

-Any items of the food class

ingredients(string)

-1:16 Alphanumeric characters

recipe(arr[int][ingredients])

A list of ingredients and their amounts

ingredientStatus(bool)

-returns true or false depending on the ingredient being in the pantry

shoppingList(arr[food])

A list of ingredients

Test Cases:

Log in:

User Status:Registered

Username:JohnDoe

Password:Abc123

Output:Log in successful

User Status:Not Registered

Username:JohnDoe

Password:Abc123

Output:Account does not exist. Creating account

User Status:Registered

Username:JohnDoe

Password:Abc123asdf123456789!

Output:Password not accepted. Excessive characters. Must be between 6 and 16

Unit Tests

- The web server will have unit tests for each of the api methods.
- unit testing each of the api endpoints will ensure that our server is responding how we expect to individual requests.
- It will also be beneficial to unit test important application data. For example we can unit that all environment Vars are present.

Functional Tests

-Create a Recipe Test

This automation test will use a testing framework to test the UI for creating a new recipe, the test will select multiple ingredients, add a description, and test uploading a photo. Once the Recipe has been submitted, we will verify the data is saved, and accessible on the other pages

-Create Account and Sign in Function Test

This automation test will test to ensure our authentication solution is working as we

expect. This test will create a new user, and login with the new credentials. This test will also verify that the cookie is present as expected.