



---

School of Computing

## CS5224 Cloud Computing

2021 / 22 Sem 2

Final Report

*Save & Save*

Reduce Food Wastage by the Unsold But Not Unwanted

Jaivignesh Venugopal	A0154811L
Marcus Tan Wei	A0149980N
Niranjana Anand Unnithan	A0228601M
Oung Yet Ying	A0225562H

## Executive Summary

Save & Save is a SaaS-based solution to a pressing problem surrounding food waste, contributed primarily by local supermarkets. It solves the problem by presenting an avenue for supermarkets to sell soon-to-be-expiring products at a cheaper price to reduce the number of products that would have otherwise gone to waste. Furthermore, apart from solving the food waste problem, it also provides additional insights such as 'popular timings to list' and 'popular products due to discounts' to supermarkets - thus, enabling them to better manage their inventory. This solution is fully built using AWS services that automate the provision of necessary resources to tackle high traffic from its users.

The key objectives of Save & Save is to reduce edible food wastage. We will be targeting two major parties – the consumer and the seller.

- Promote and entice the unwanted food to potential consumers using Save & Save
- Provide sellers with analysis on items such as popularity, best period of time to sell certain products etc, to strategically improve sales performance

### Innovative Contributions

There are some ongoing efforts by supermarkets to reduce food wastage, such as FairPrice Online having clearance sale that sells off-season items and near-expiry items. (Towards Zero Waste, n.d.). However, without much publicity and research, these efforts are still unheard of by the majority of the population. By having Save & Save, data analytic tools will be used to help to recommend users food items that they can buy. A platform that provides a wider range of supermarket store selection also provides users with more choices. Save & Save also collects data to provide analysis on sales through Save & Save so that supermarkets can better manage their logistics. Therefore, we make creative contributions by advocating smart decision making in conjunction with saving money through discounts, in the effort to minimize food wastage.

## Business Case Identification

### Problem Formulation

Over 1 billion tonnes of food waste is produced every year, even as more than 800 million people worldwide go to bed hungry every night<sup>1</sup>. Supermarkets have been singled out as one of the major culprits behind food waste globally<sup>2</sup>. In 2019, a local Supermarket, Fairprice, reported to have generated 2940 tonnes of food waste, as compared to 3170 tonnes of food waste in

---

<sup>1</sup> <https://www.straitstimes.com/singapore/unsold-but-not-unwanted>

<sup>2</sup> <https://www.straitstimes.com/singapore/how-supermarkets-fight-food-waste-in-singapore>

2018<sup>3</sup>. Although measures have been taken by local supermarkets to reduce this wastage, the amount of food waste generated is still very significant and a cause for concern. Multiple solutions have been proposed over the years - donations to needy communities, discounted prices, etc. - however, there has not been a clear winner across the various supermarkets.

### Solution

*Save & Save* is a potential solution to this problem - a web platform for the various supermarket vendors to list and sell items that are expiring/to be cleared, at a discounted price. This platform would act as a bridge to channel the information to the interested consumers, encouraging people in close proximity to the supermarkets to reserve and buy these products. In addition, our application also aims to help Supermarkets to make a more informed inventory management and pricing of items to achieve better sales. This way, we hope to reduce waste while helping the community to make their penny worth their purchases.

There are mainly two parties at play here - the regular consumer and the store. To begin listing products, a store would first register with the platform. Thereafter, the store will be able to upload csv files of their products in a predefined format. These csv files would contain necessary information about the product listings - name, category, new cost per unit, original cost per unit, offer expiry date and image links (optional). With this, the platform would be able to put up listings for anyone to browse.

Additionally, the platform will also provide ways for consumers to reserve (temporarily buy) these products before making their way to the shops to collect these products. This way, it will be possible for the platform to monitor stocks for the different products (that were listed) and update the catalogue accordingly. This importantly paves the way for the platform to **further value-add** to the stores - by producing statistics on how quickly certain products get sold, price points that attract consumers and also timings that are favorable for listings to go up.

Thereafter, *Save & Save* will generate analytical reports based on transactions and item impression data. This report will provide insights for general inventory management, listing prices and discounts, as well as the listing time.

Comparison with Available (Similar) Services

Name	UglyFood (Store)	Supermarket Online Store
Introduction	Founded in 2017, aims to reduce food wastage by selling excess or ugly produce. Have an online store and physical store.	Aggregated all supermarkets that have online stores in Singapore, for eg. Fairprice, Sheng Siong, Giant etc

---

<sup>3</sup> <https://www.straitstimes.com/singapore/unsold-but-not-unwanted>

Store operating hours	Tue-Fri: 3pm – 5pm Sat: 1pm – 2:30pm Sun, Mon & PH: Closed	Daily (Generally 7am - 12am)
Services	Delivery, Pickup (only at 3 locations)	Delivery, Pickup
Analytics	None	Real-time predictive segmentation

Although these services do put in effort in reducing food wastage, we feel that these efforts are insufficient. UglyFood lacks advertisement and analytical tools to recommend products to users. Furthermore, the operating hours are short and difficult to fit in schedule. Supermarket Online stores have a clearance section where they put in effort to sell near expiry and unsold food, it is a very small portion of their listings and the main focus of these Supermarkets is to sell high quality products to boost the branding and attract customers. Therefore, Save & Save provides a platform with data analytics that recommend users products and it places emphasis on the notion of reducing wastage due to unsold but not unwanted food by retailers. Furthermore, it operates at the same hours as supermarkets with more pickup locations and provides an all-in-one platform for users to choose items from, therefore, much more convenient.

## Business Model

### Revenue Model

As a web-based service, Save & Save will be free for customers who are looking to buy products. This will boost the attraction of users who will potentially be buying these leftover or soon-to-be-expired products from us to use our application to view product listings. Charging users for using our service will not incentivise users to buy products using our application, and hence, will not help with the ultimate goal of reducing food wastage due to unsold items.

Our main revenue stream would be from a subscription based model where sellers will have to pay to list items on the platform. However, to lower the entry costs for SMEs and encourage platform adoption, we will be following a Freemium subscription model where sellers are only required to pay subscription fees when they exceed a certain number of sales.

We will also be charging a premium subscription service for sellers who want our customized data analytic report that include statistics on their sales for each product by category and also popular timings of purchase. This will help our stakeholders to plan and manage their inventory so as they can reduce stocking up on food items that do not sell as well and to stock up more items that are in higher demand albeit the lower quality.

After gaining popularity, with the website traffic generated by a large pool of users, customized advertisements such as food deals or food delivery services could be exhibited as user notifications or banners on Save & Save. Supermarkets can also bid for advertisements of their items on our platform. This will generate the second stream of income.

#### SaaS service vs On-premise IT resource

SaaS service was chosen instead of traditional web services using on-premise IT resources because it provides a lower entry and operation cost to set up the services. If on-premise IT resources were utilized, we would need to consider additional costs demanded by the following services - annual maintenance cost, vendor support, building rental and facility cost.

However, with adoption of SaaS service, pricing models are also more flexible and easier for a small team to scale up and upgrade. Overall, the total costs required would be lower and it eases development, which is suitable for a fast startup.

## Application Features

There are primarily 5 workflows in the application and these workflows/features are visually demonstrated in the video file that has been submitted together with this report. To aid in better understanding, explanations for the workflows have also been given below.

Feature	Explanation
Register / Login	A user will be able to register him/herself as both a store and customer. This gives the user entity to both sell and buy products.
Listing Items via CSV upload	A user - store - will be able to upload product listings via a CSV upload. However, the user has to follow the uniform pre-defined CSV format for the action to be successful.
Browsing Items	A user - store and customer - will be able to browse and view products that were put up for listing.
Purchase Workflow	A user - store and customer - will be able to add items to a cart from various stores. Subsequently, the user will be able to reserve these items by making a purchase.
Data Analytics	A user - store - will be able to view insights drawn from data.

#### *Application Features*

# Architecture & Implementation

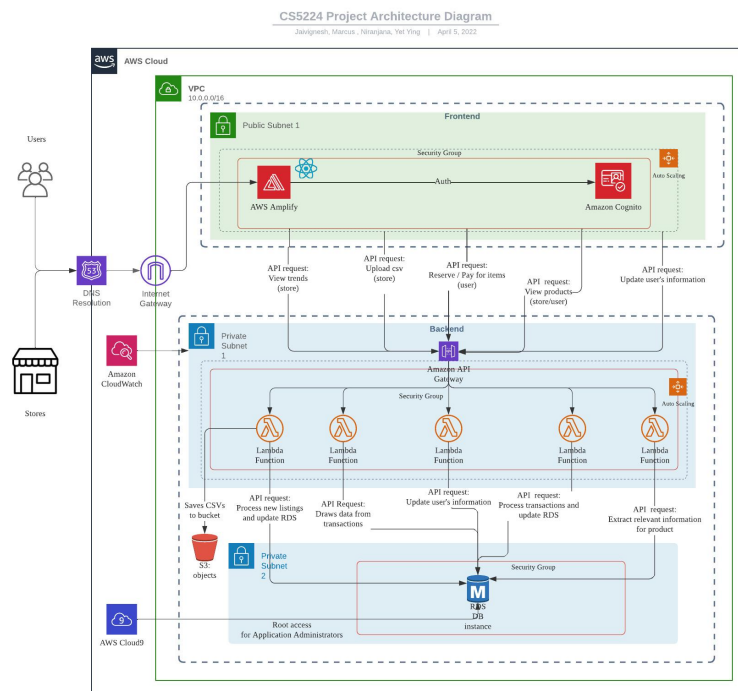
## Overall SaaS Design

The services that we used were primarily from AWS. Since our product is a web application, this requires a standard set of components - a Backend system, a Frontend system and a Database system. Further breakdown of the required components is listed below:

Components	SaaS Services
Frontend System	Amplify (build-in to configure CloudFormation, CloudFront, S3, AWS auth)
Backend System	AWS Lambdas
Database System	Amazon RDS (PostgreSQL)
Storage (for CSVs)	S3
Monitoring	AWS CloudWatch

*Components of Web Application*

The following architectural diagram encapsulates how the aforementioned services/components would come together.



*Architectural Diagram*

## Data Preparation

The driving force of this application would be inventory data and the respective discounts for the products in the supermarket. However, currently, discount information is not readily available as a dataset in Singapore. Therefore, a decision was made to scrape data from a major supermarket company here in Singapore - NTUC Fairprice - and augment it to suit the platform's needs.

This was done using a Python script and Selenium. Item data such as name, category, price and image links were extracted using the scraper. Using this technique, 28,591 products across various categories were collated in a master-items CSV file.

However, this was not enough. There were still four important attributes that were missing - discounted price, reason for discount, quantity and date availability. These were the attributes that had to be augmented. This was done by randomly assigning discount factors and then mapping a corresponding reason for the various discount ranges. Similarly, the quantity attribute and the date availability attribute were populated by selecting a value from 0 - 200 and by picking a date value in the next eight weeks respectively.

id	name	category	quantity	actual_price	discounted_price	discounted_reason	available_date	image_link
1315251	DODO Mini Chikuwa	Meat & Seafood	197	2.2	0.1	NORMAL DISCOUNT	3/4/2022	<a href="https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90039083_XL1_20201014.jpg">https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90039083_XL1_20201014.jpg</a>
1375145	Pan Royal Frozen Prawn Meat (nett weight 500g +/-) 1 KG	Meat & Seafood	116	11	0.73	NORMAL DISCOUNT	28/3/2022	<a href="https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90080659_XL1_20211009.jpg">https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90080659_XL1_20211009.jpg</a>
1334858	DODO Cooked Fish Ball (L)	Meat & Seafood	112	3.45	1.36	SEASONAL OFFER	1/4/2022	<a href="https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90053543_XL1_20201226.jpg">https://media.nedigital.sg/fairprice/tpol/media/images/product/XL/90053543_XL1_20201226.jpg</a>

*Sample Listing in a CSV file*

Thereafter, this list was split equally by 10, to augment listings from 10 different stores - Sheng Siong, Cold Storage, Giant, Lee & Lee, Little Farms, Marketplace, Mustafa, NTUC Fairprice, Redmart and Ryan Grocery. This way, there were now 10 users (stores) for the platform, with their respective listings in a CSV format to upload.

## Frontend System

Frontend UI is mainly developed by using ReactJS and Material UI component library. An open-source free template (CodedThemes, n.d.) is used as a quick start to build the skeleton of pages. After building up the user interfaces, AWS Amplify service is used to host and deploy the React application as the web application.

## **AWS Amplify**

AWS Amplify is a service provided by AWS to ease the configuration of AWS services for applications. This allows developers to focus on application development instead of configuring AWS resources for their application. This service is backed by Cloud Formation service. AWS Amplify helps the developers to define the required resources and use CloudFormation to run

the tasks like setting up the roles and policy, S3 buckets and gateway depending on the resources required for the application.

### ***S3 Bucket & CloudFront***

With the aid of AWS amplify, S3 bucket and CloudFront are configured to host the React application. S3 bucket acts as the static web page to serve the index.html, js and css bundles. After the client browsers obtain the files, rendering logic will be carried out in the browser. CloudFront is configured to serve the files faster in various regions.

### ***AWS Cognito***

The authentication service, AWS Cognito, is added in the frontend project by using the AWS Amplify too. It has provided the UI framework to allow developers to use or customize the login page based on the application requirement. In our project, only email and name are required for registration purposes. More configuration is possible during the initialization phase using the Amplify CLI. By using the framework, it saves the time to configure the api, database and authentication registration flow in the frontend.

## **Backend System**

### ***AWS Lambdas***

The backbone of any web application - the backend - would often be built using heavy-weight frameworks such as Rails or Django, where functionalities such as user authentication, content administration, and defense against many common security mistakes work right out of the box. However, it was decided that we would primarily use AWS Lambdas to deal with all backend logic, as lambdas were extremely lightweight and the nature of the web application did not demand many of the said features. This particular project simply needed Lambda functions to communicate with both the database system and the S3 storage system - which the functions flawlessly catered for.

As for scalability, Lambdas were an excellent choice as they were high-availability compute infrastructure that took care of capacity provisioning, automatic scaling, code monitoring and logging. This way, the team did not have to worry about using load balancers, setting up configurations for EC2 instances or even setting up logging systems. Lambdas took care of them all.

Importantly, since all communication between the frontend system and the backend system was provisioned through REST API calls, it was easy to invoke the relevant/corresponding Lambda



functions using the API Gateway. The following table describes the Lambda functions that were created and their functionalities.

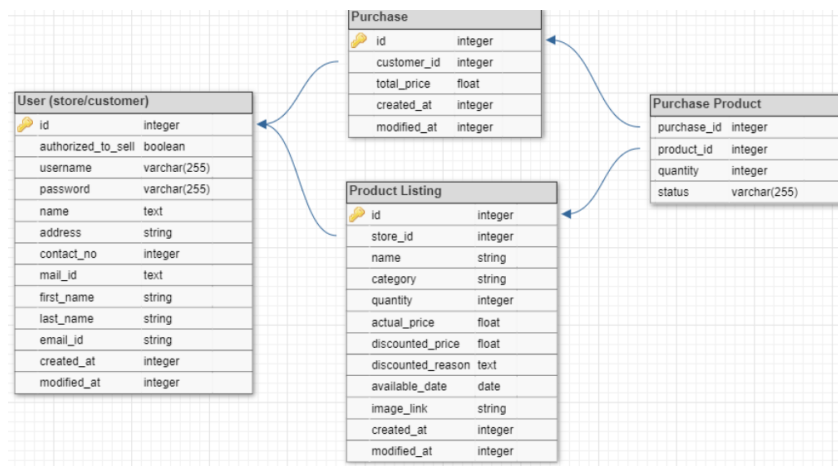
Lambda	Functionality
customer	Handles create, read, update and delete of platform user's (store/customer) information.
csv-handler	Handles creation of product listings via CSV files.
product	Handles create, read, update and delete of a single product listing.
products-query	Handles filtering of products based on keywords and offers expiry date.
purchase	Handles create, read of transactions between the platform and users.
purchase-product	Handles read of product listings and their quantities that were part of a transaction.
data-analytics-query	Handles queries such as 'popular listings', 'popular time to list' etc

*Lambda functions and their responsibilities*

### Amazon RDS

Due to the nature of the application, there was a need to have a database system. Information such as store/customer details, product details, and transaction details were necessary to be stored permanently to facilitate the various functions of the product. In this case, since information is well structured and relational, an SQL database was chosen instead of NoSQL to provide transactional data integrity and faster analytical queries. Specifically, the team decided to use PostgreSQL over other SQL database systems due to its robustness and fault-tolerance.

The entity relationship diagram below illustrates how the various tables are related.






*ERD Diagram*

The team now had two choices - to instantiate PostgreSQL instances in Amazon EC2s or to simply use Amazon's Relational Database Service (RDS) service. Since Amazon's RDS system primarily provided backup and recovery services, auto-scaling and management of security patches, it was undoubtedly the better choice for our application. Furthermore, it provided support for our preferred database system - PostgreSQL.

### **Amazon S3**

The application provides a way for stores to bulk upload their products via a simple file - CSV - upload. Internally, in the backend, each row of the file is added to the database. However, it is necessary to store the csv file for logging/monitoring/debugging purposes after the database update. Amazon S3 bucket is therefore used for this purpose. The lambda function that is designated to handle csv upload API call, also has read/write access to the S3 bucket. This lambda function specifically deals with processing and then storing the csv files in the S3 bucket.

<input type="checkbox"/>	Name ▲	Type ▼	Last modified ▼	Size ▼	Storage class ▼
<input type="checkbox"/>	 <a href="#">marcus_1648364835_2287947214.csv</a>	csv	March 27, 2022, 15:07:18 (UTC+08:00)	9.8 KB	Standard
<input type="checkbox"/>	 <a href="#">marcus_1648365303_2673284218.csv</a>	csv	March 27, 2022, 15:15:04 (UTC+08:00)	9.8 KB	Standard
<input type="checkbox"/>	 <a href="#">mustafa_1648655429_532434392.csv</a>	csv	March 30, 2022, 23:50:31 (UTC+08:00)	101.0 B	Standard

*CSV files in S3 storage*

### **Data-Analytics**

Through our application, we aim to provide our primary customers: supermarkets, with analytics services to learn customer behaviors so that they can make informed decisions regarding stocking, listing, etc. This, in turn, helps reduce wastage, which is the main goal of our application. Currently, we offer two main services, namely (i) Popular products of a supermarket and (ii) Best time to list the products

**Popular Products of a supermarket:** The popular products endpoint returns the most frequently reserved items by customers from a specific supermarket for a given period. Using this API, the supermarkets can retrieve the most popular products purchased from their stores on a weekly / monthly basis and stock these items more. The application maintains a database that stores product listings made by the supermarkets as well as information about reservations of products by customers. We do a join on both the tables to identify the products that were purchased from a specific supermarket. We filter for the given date range and order the items on the sum of the quantities purchased. It is presented in the form of a visualization chart to the user, which depicts the products as well as their quantities.

Best time to list products: The best time to list products endpoint returns the time range on each day of the week where the frequency of product reservation was the maximum. Using this API, the supermarkets can identify the time at which most reservations occur and plan their listings accordingly to enhance their views. This, in turn, results in better product purchases. We have divided each day into 12 segments - regular intervals of 2 hours. Given a date range, we extract all the purchases that happened within this duration and group them to the respective time segments. We extract the number of purchases that happened in each of the time ranges and average it over the number of days in the date range. Finally, we display this information in the form of a trending chart. Using this, the supermarket owners can get an overall idea of the customer buying patterns and manage their listings accordingly.

### Logging System

It is necessary to have a logging system for any application, to understand bugs or crashes that occur during production mode. For this project, AWS Cloudwatch was used to monitor the performance of all the Lambda functions. Print statements were littered around in all the lambda functions to better facilitate comprehension of logs. This aided in debugging during the development of the application and also when we stress-tested the application in production mode.

## Economic Factors

Firstly, when users access the Save & Save website, there will be revenue generated from the website traffic and advertisements on our website service as one source of economic benefit.

Secondly, our application promotes sales of unsold but unwanted food, which helps to add to the supply of food. Based on the Law of Demand and Supply, having this boost of supply will help to generally lower prices of food, which will help lower income families in Singapore as food becomes more affordable.

Thirdly, the impact of increased sales in general (additional sales made due to selling of unsold food items) will benefit retailers. Extra revenue generated also means that more resources can be allocated to other areas, such as research and development, to bring about more benefits.

Key considerations of our application would be *Cloud Service Provisioning*, where we as cloud consumers want to impose a cloud service usage threshold. This is because our target audiences are Singaporeans, and during non-seasonal periods (such as festive season), we would limit the cloud service usage period to be the same as the operational period of supermarkets. This is because during the night time, there will be significantly lower demand and we would want to

tabulate inventory changes at the end of the day to update supermarkets so they can manage their stock count. During the festive seasons, there may be a dramatic fluctuation in requests to the website and we will also need to deal with the increase in demand.

### Pricing Models

Pricing model is curated on the assumption of about 100,000 active users a day.

Service	Properties	Cost (Month)
AWS Amplify	<u>Assume 100,000 requests per day</u> 4 builds per month (4 releases per month) 4 Mb data stored per month 12,000 GB data served per month	\$0.01/build minute \$0.023/GB stored \$0.15/GB served <b>1800.04 USD</b>
AWS API Gateway	<u>Assume 1,000,000 requests per day, 10ms/request, with basic</u> Total number of requests per month: 30,000,000	30 USD
AWS Lambda	<u>Assume 1,000,000 requests per day, 10ms/request, with basic</u> Total number of requests per month: 30,000,000 Total memory for Lambda session: 128mb memory	6.63 USD
AWS RDS	100% Utilization Per Month db.t2.medium Standard Single Availability Zone Storage SSD - General Purpose 1000 Gb	168.44 USD
AWS S3	<u>Assume 100 stores do a CSV upload (10mb) everyday for 30 days.</u> Retention duration of CSV files: 30 days Total No of Put Requests: 3000 Total Storage Space Needed: 30Gb	0.71 USD
Total		2005.82 USD

### Tradeoff Between Cost and SLA

Contrary to other on-premise services, we reduce the cost of IT resources by building our website and running the application. We guarantee our service quality using Amazon Web Services, where effort and money spent on initial setup and machine supply is a lot lower. Using AWS Amplify also made developing the frontend application much easier. AWS Lambda also reduced the need to manage the infrastructure and server. Currently, we are using AWS Lambda because we feel that we are able to reduce cost of execution and development by providing reasonable response time, albeit not real-time. If we manage to attract much more users, we will be planning to migrate to running our application on multiple EC2 instances to ensure

reliability and scalability. For that, we will be raising the premium fees for Supermarkets in order to sustain higher infrastructure costs of running replicas.

## Conclusion

*Save & Save* is a solution to the food waste problem, that is aggravated especially by supermarkets. This application was made with key considerations in the domain of cloud computing. Although effort has been put to ensure that the application is able to handle traffic and security issues, it is important to undergo a round of beta-testing to observe how it behaves with live data and traffic.

## References

- CodedThemes. (n.d.). *Berry Free React Material Admin Template*. GitHub. Retrieved March 5, 2022, from <https://github.com/codedthemes/berry-free-react-admin-template/tree/main/src>
- TowardsZeroWaste. (n.d.) *Food Waste*. <https://www.towardszerowaste.gov.sg/foodwaste/>