

# **Amazon Internship: Meridian Migration**

Project report submitted in fulfillment of the requirement for the degree of

**Bachelor of Technology**

in

**Computer Science and Engineering**

By

**Naina Garg 181424**

Under the supervision of

**Dr. Himanshu Jindal**

to



Department of Computer Science & Engineering

**Jaypee University of Information Technology Wagnaghat,**

**Solan, Himachal Pradesh**

# Certificate



## INTERNSHIP LETTER

Naina Garg  
House no. 507, Gagan Vihar  
Shamli – 247776  
UP  
IN

Dear Naina,

On behalf of **Amazon Development Centre (India) Private Limited**, a company incorporated under the laws of India, having its registered office at # 26/1, Brigade Gateway, World Trade Centre, 10th Floor, Dr. Rajkumar Road, Malleshwaram (W) Bangalore - 560 055. Karnataka India (hereinafter the “Company” or “Amazon India”), we are very pleased to issue this Internship Letter for the position of an **Intern** at **Bangalore**, India.

Your internship with the Company will be subject to your acceptance of this Internship Letter and the terms and conditions set forth hereinbelow on or before 10 business days in the manner provided for by the Company.

Upon your acceptance of this Internship Letter, the same shall form a valid and binding agreement between Amazon India and you, and you shall be bound by the terms and conditions stipulated herein below.

### **1. Date of Commencement**

Your internship with Amazon India will commence on **07-Feb-2022** and shall end as per the provisions contained in Section 12 herein below. The said duration of internship shall hereinafter be referred to as the “Term”.

### **2. Duties**

- 2.1 You will be engaged in the position of **Software Dev Engineer Intern**. Your manager will advise you about your duties and responsibilities after your joining with us. You will be expected to perform your duties to the best of your ability at all times as per the

1

REGISTERED OFFICE : # 26/1, Brigade Gateway, World Trade Centre, 10th  
Floor, Dr. Rajkumar Road, Malleshwaram (W) Bangalore - 560 055. Karnataka  
India

Tel. : + 91 - 80 - 6787 3000, Fax : +91 - 80 - 3007 1031 / 33 CIN :  
U72200KA2004FTC034233

## **Candidate's Declaration**

I hereby declare that the work presented in this report entitled “**Meridian Migration**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat, is an authentic record of my work carried out over a period from February 2022 to May 2022 under the supervision of **Dr. Himanshu Jindal** (Assistant Professor(SG), Computer Science and Engineering and Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

**Naina Garg**  
**181424**

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

**Dr. Himanshu Jindal**  
**Assistant Professor (SG)**  
Computer Science & Engineering and Information Technology  
Dated: 27 May 2022

## **Acknowledgments**

I would like to express my sincere gratitude to my project guide “**Dr. Himanshu Jindal**” for allowing me to work on this topic. It would never be possible for me to take this project to this level without his innovative ideas and his relentless support and encouragement.

I also thank our family, friends, and all the faculties who gave their unfiltered suggestions and feedback and helped me face all the challenges and hurdles which came along during the project.

**Naina Garg**  
**181424**

# Table of Contents

<b>Certificate</b>	<b>2</b>
<b>Candidate's Declaration</b>	<b>3</b>
<b>Acknowledgments</b>	<b>4</b>
<b>Table of Contents</b>	<b>5</b>
<b>Abstract</b>	<b>6</b>
<b>Chapter 1: Introduction</b>	<b>7</b>
1.1 Introduction	7
1.2 Problem Statement	8
1.3 Objectives	8
1.4 Methodology	10
1.5 Organization	18
<b>Chapter 2: Literature Review</b>	<b>19</b>
2.1 Literature Survey	19
<b>Chapter 3: System Development</b>	<b>22</b>
3.1 Overview	22
3.2 Assumptions	22
3.3 Requirements	22
3.4 Architecture	25
3.5 Validations	33
3.6 Design	37
3.7 TOOLS	42
<b>Chapter 4: Performance Analysis</b>	<b>43</b>
<b>Chapter 5: Conclusions</b>	<b>48</b>
5.1 Conclusions	48
5.2 Future Scope	49
<b>REFERENCES</b>	<b>50</b>

## **Abstract**

The associate tools are a set of web applications that are used by 1P, 2P, and 3P associates at Amazon Physical Stores. There are currently over a hundred unique tools that are owned by dozens of teams distributed across multiple organizations within Amazon. The current iteration of associate tools are embedded within IhmSeaward, which is a base tool that loads different associate tools within an iframe. IhmSeaward is served through IhmPewter, which is a ruby-on-rails server. IhmSeaward and the current generation of associate tools are making use of the AngularJS framework, which is planned to reach its end-of-life. We have found that using AngularJS also reduces team productivity as there is less experience with the framework across the teams that need to build tools and it lacks many features of modern frameworks like ReactJS, Angular 2 or VueJS. The Meridian framework is the recommended framework for the IHM organization. Since Meridian framework full support is only available in ReactJS, we decided to use ReactJS framework for Associate Tools Platform to support Meridian migration.

KisanPoReceiveTool is an associate tool loaded as part of IhmPewterWebsite. There are two main PO operations in Vendor POReceive flow: Get PODetails and Receive Quantity against a PO. PO Search API should require both POId and Vendor Code. Associate is required to Scan/manually enter Vendor Code & POId to view all the ASINs to be inbounded for the PO.

# Chapter 1: Introduction

## 1.1 Introduction

**Amazon** has completed the integration of its grocery stores Fresh and Pantry into a single unified store called Amazon Fresh. The new store, available across 300+ cities in India, continues to offer customers unbeatable savings, a wide selection of products, and fast and convenient delivery options in one single online destination.

Driven by a commitment to be an ‘everything’ and ‘everyday’ store for customers. A customer obsessed company and continue to listen to our customers to offer them enhanced shopping experiences, a wide range of selection, and the best value and convenience. Customers will continue to enjoy super value savings, a wide selection of products, and convenient delivery options. They also get an upgraded shopping experience, with a dedicated app-in-app for grocery, and convenient features like personalized widgets and reminders to ensure that frequently shopped items aren't forgotten during checkout.

We are laser-focused on providing customers the best online shopping experience, coupled with quick and safe delivery. We know fully well that customers will shop with us only until they find a better experience elsewhere, so we strive hard to meet their standards. This launch has allowed us to simplify the shopping experience for groceries via our dedicated AmazonFresh app-in-app experience, and sets us up to deliver many new features and enhancements in the coming months. Apart from offering great savings, Amazon Fresh will also reduce barriers to grocery shopping online.

Clicking on the Amazon Fresh icon on the homepage takes you to the dedicated grocery shopping store, where you'll find features that help you build your weekly/monthly basket in a few minutes. Shopping for groceries online has become more rewarding, fast, safe and convenient with Amazon Fresh.

## 1.2 Problem Statement

We have found that using AngularJS also reduces team productivity as there is less experience with the framework across the teams that need to build tools and it lacks many features of modern frameworks like ReactJS, Angular 2, or VueJS. The Meridian framework is the recommended framework for the IHM organization. Since Meridian framework full support is only available in ReactJS, we decided to use ReactJS framework for Associate Tools Platform to support Meridian migration.

### Goal:

1. Migration for the existing APIs supported by IhmSeaward.
2. Migration of the utilities, services, and other features supported by IhmCrookhook.
3. Migration plan for existing tools: This document will just recommend migration steps for the tools, but tool owners will own the migration.
4. Deprecation path of IhmSeaward & IhmCrookhook.
5. Deprecation of existing unused services/libraries

## 1.3 Objectives

KisanPoReceiveTool is an associate tool loaded as part of IhmPewterWebsite. There are two main PO operations in Vendor POReceive flow: Get PODetails and Receive Quantity against a PO.

1. PO Search API should require both POId and Vendor Code. Associate is required to Scan/manually enter Vendor Code & POId to view all the ASINs to be inbounded for the PO.
2. Need to show Zones(Ambient/Tropical) for each ASIN during Inbound and categorize the ASINs based on zones.



3. Need to show the total weight received for weight based ASINs and total units received for unit-based ASINs when PO details are retrieved during Inbound.
  4. We need to show the entered Weight/Unit to the container map when an Associate enters the weight for a scanned container during Inbound.
  5. When a user enters the lotContainerCode during POReceive, there should be multiple validations done on it before adding items in it.
  6. We need to fetch disassembled relationships for a weight based ASIN during PO Inbound. We need this data to be sent in associate audit and for showing details of weight based ASIN during Inbound.
  7. Inbounding of weight based ASINs should be done in weights only and not in units. Users should not be allowed to enter the number of units for a weight based ASIN as it is being done currently.
  8. Weight based ASIN received in weight conversion to units for vendor payments for phase 1 based on conversion factor : 1 unit = 1000 grams:
  9. If weight is greater than 500 grams the units are rounded up and if weight is less than 500 grams the units are rounded down.
- (e.g. if total quantity being received is 100.6 KG vendor payments will be done on 101 units and qty received is 100.1 Kg vendor payments will be done on 100 units)

## 1.4 Methodology

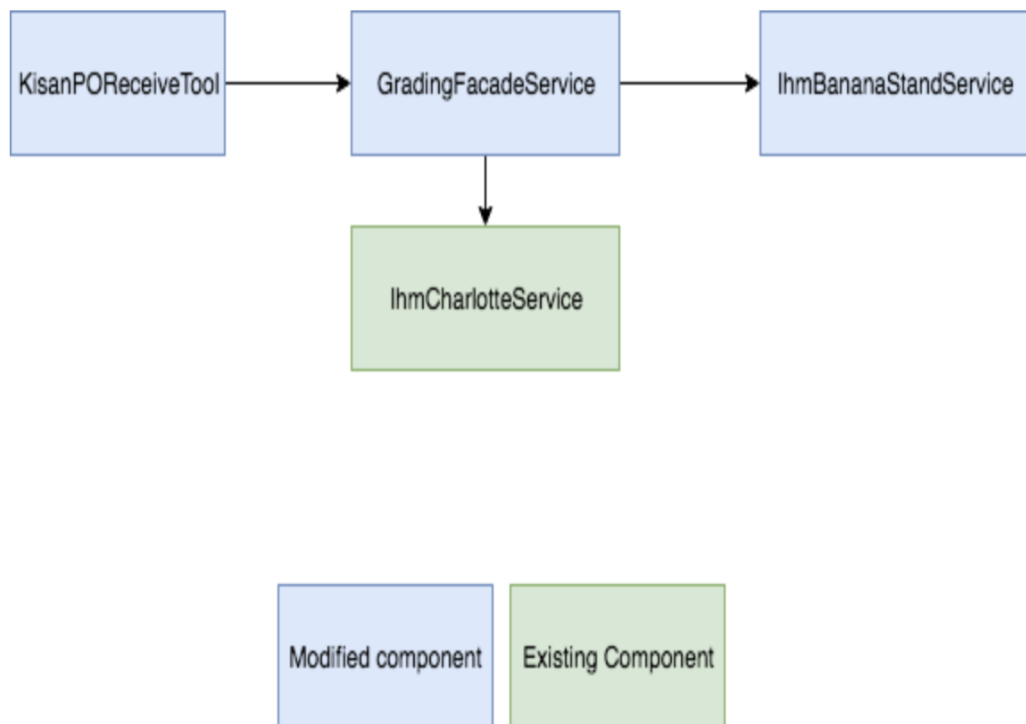
### 1.4.1 Perform vendor code Validation & fetch ASIN attributes

This is the current architecture and sequence diagram for GetPODetails operation in POReceive flow.

---

#### Architecture Diagram

---



Architecture diagram

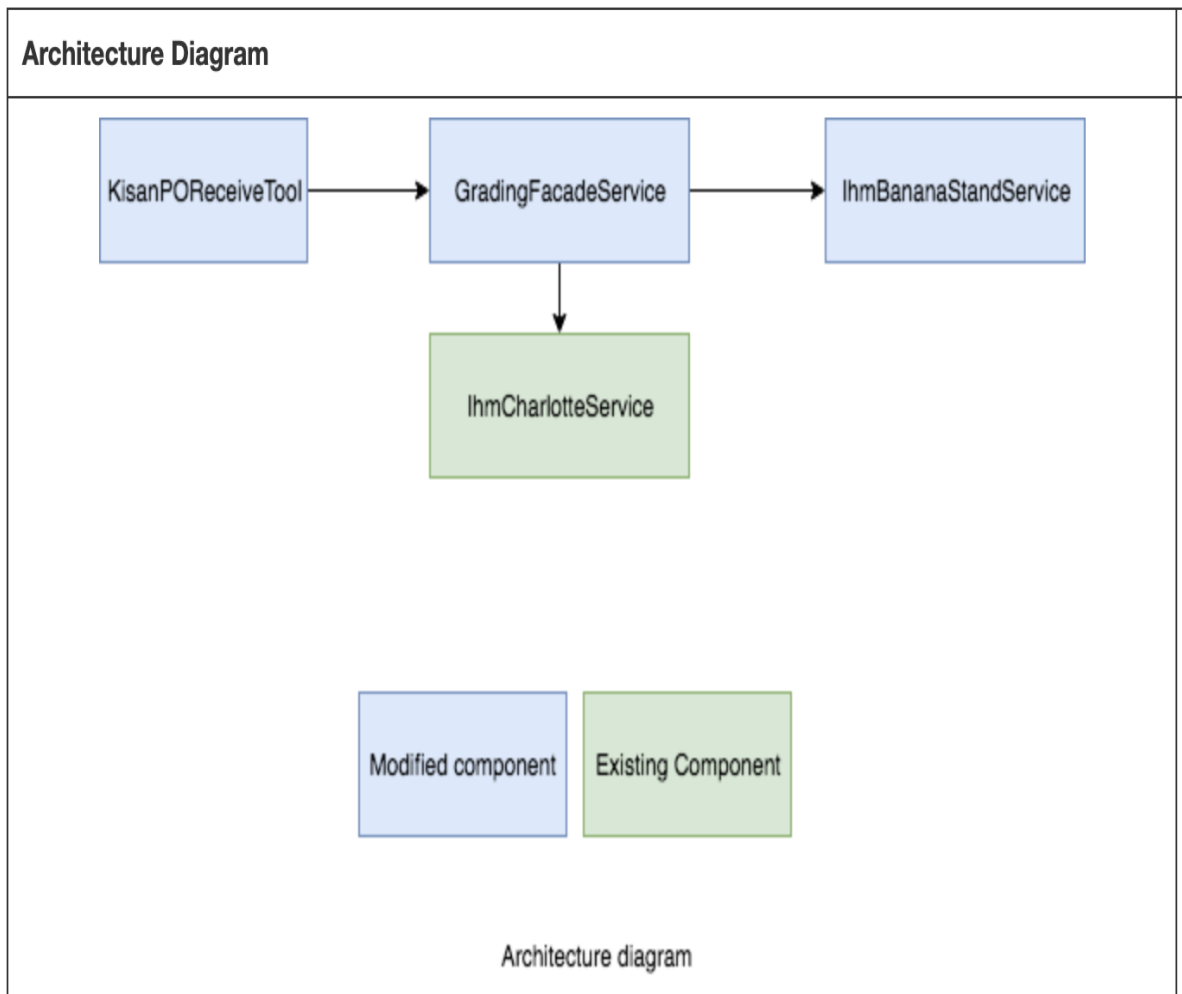
---

### Current Sequence Diagram

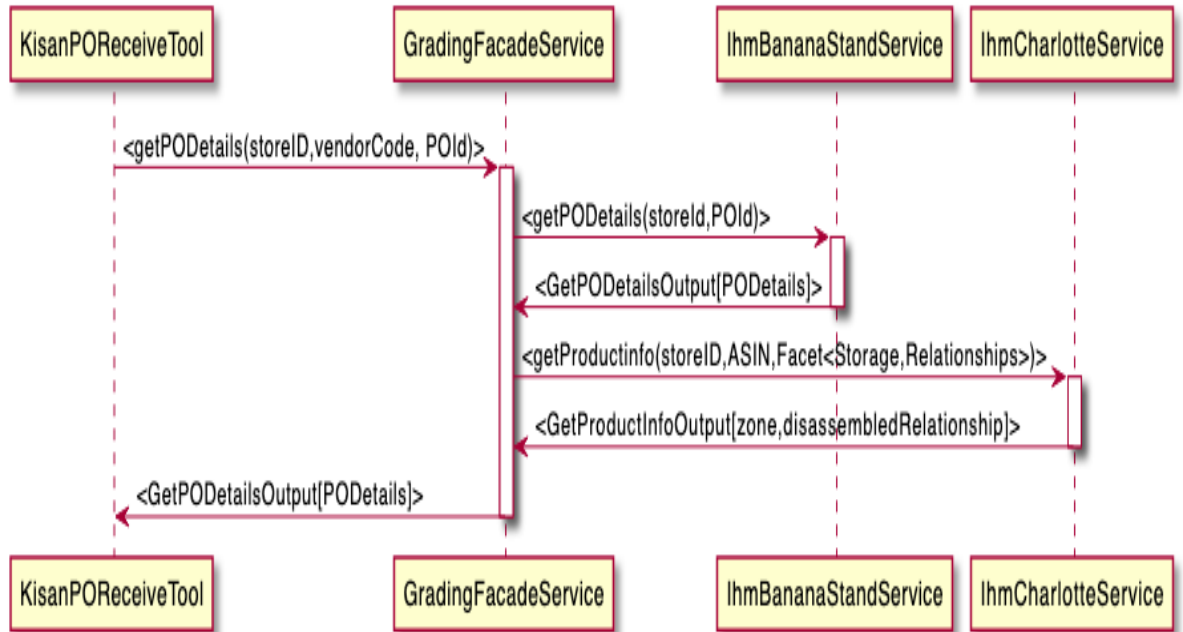


Following are the approaches we will use to fulfill our requirements for performing validation on Vendor Code and fetching ASIN attributes from catalog in the new Inbound Experience flow:

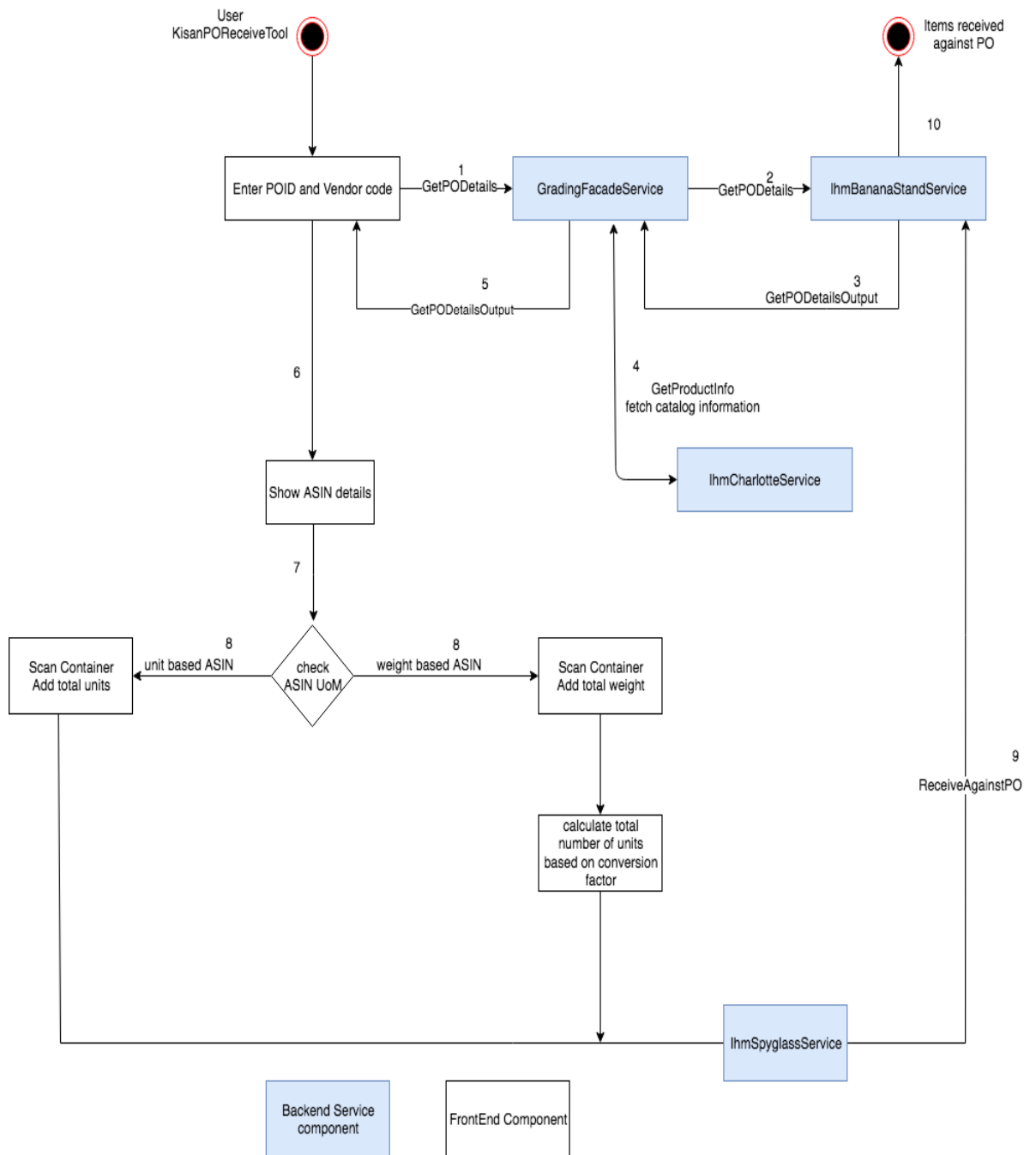
### 1.4.1.1 Approach 1: Using GradingFacadeService (Preferred)



### Sequence diagram



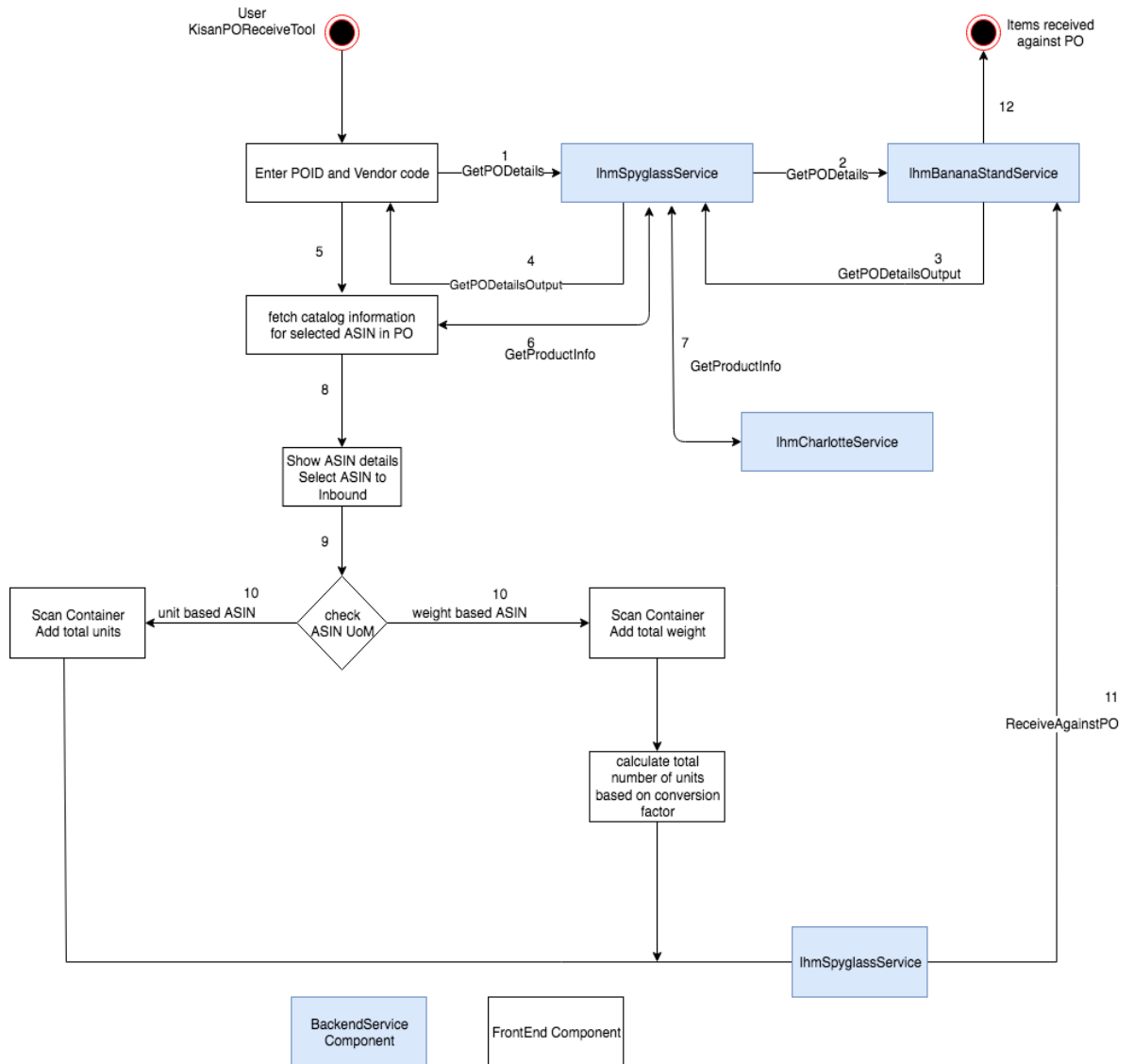
**Flow chart with API details:**



### 1.4.1.2 Approach 2: Using IhmSpyglassService

In this approach, we are retaining the current architecture of GetPODetails PO operation.

**Flow chart with API details**

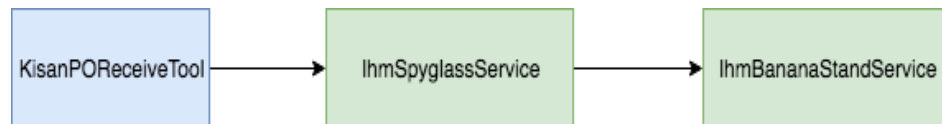


## 1.4.2 Receive Against PO

In the new flow for Receive quantity against PO, we will retain the current architecture for POReceive flow. If the ASIN is unit-based we will simply pass the number of units in the ReceiveFlow as it happens in the current flow. If the ASIN is weight-based we will calculate the number of units we will make using the conversion factor, and use the below logic to calculate the number of units per container.

To identify if an ASIN to be received is weight-based or Unit based, we will make use of DisassembledTransformationRelationship defined for that ASIN in the catalog. This relationship is defined only for weight based ASINs.

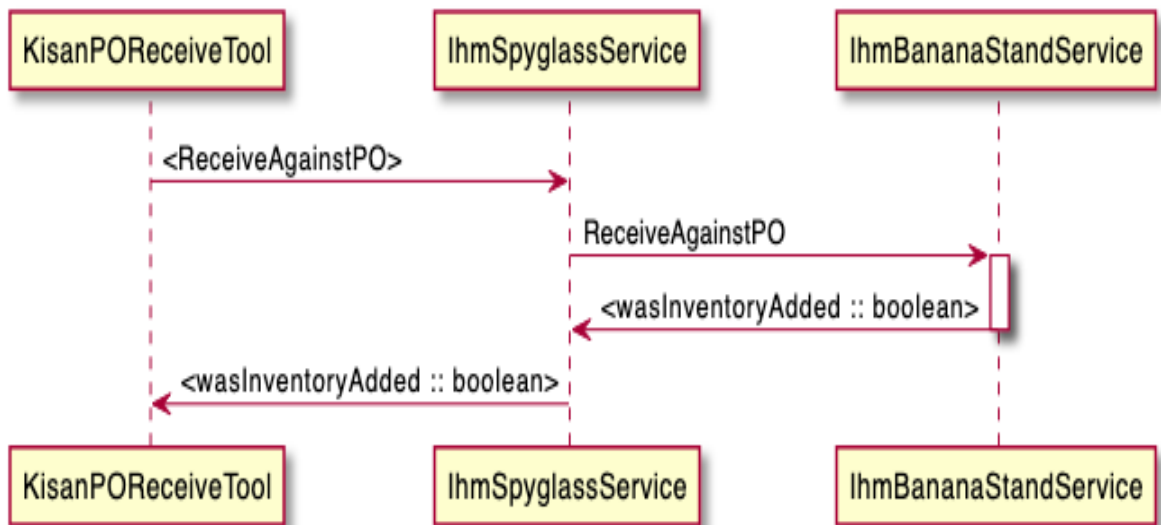
### Current Architecture Diagram:



Architecture diagram



**Current Sequence Diagram:**



## **1.5 Organization**

Amazon is the top IT company in India, and a major IT company in the US. Amazon employees around 3 lakh employees and recruits around 20 thousand fresh people every year from India. Amazon also hires from different countries across the globe. Amazon offers various role in the company like develop, Designer, Tester and Manager in the company, but, before becoming the associate every person should complete the intern period and after the intern period there is one year of probation period in the company for the associate to join the company. The internship period varies and depends on the roles, which the intern gets, like someone who got developer profile, for them internship period will be of around 4-5 months and for the quality insurance, it might vary from 5-6 month. The domain allocation is random in the Amazon for the interns, but sometimes it depends on the assimilation test also. The person who got higher marks in the assimilation test will have higher chances to get a better profile or domain and it also depends on the first come first serve basis.

### **Mission, vision, values, and objectives**

Mission – Amazon mission is to train every fresh person who got selected in to the Amazon. Amazon provides internships to every person who gets selected in Amazon. Every year Amazon train college fresh out student in bulk number before giving them the associate role. This recruit happens from all colleges over the india.

Amazon spends much time,effort and money in training the intern before giving them the actual work and before them to work in the real environment.

## Chapter 2: Literature Review

### 2.1 Literature Survey

- Ramos, Miguel, Marco Tulio Valente, and Ricardo Terra. "AngularJS performance: A survey study." *IEEE Software* 35, no. 2 (2017): 72-79.
- Chansuwath, Wutthichai, and Twittie Senivongse. "A model-driven development of web applications using AngularJS framework." In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pp. 1-6. IEEE, 2016.
- Gupta, Suhit, Gail Kaiser, David Neistadt, and Peter Grimm. "DOM-based content extraction of HTML documents." In *Proceedings of the 12th international conference on World Wide Web*, pp. 207-214. 2003.
- Kopecký, Jacek, Karthik Gomadam, and Tomas Vitvar. "hrests: An html microformat for describing restful web services." In *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 1, pp. 619-625. IEEE, 2008.
- Cohen, William W., Matthew Hurst, and Lee S. Jensen. "A flexible learning system for wrapping tables and lists in HTML documents." In *Proceedings of the 11th international conference on World Wide Web*, pp. 232-241. 2002.
- Lie, Håkon Wium, and Janne Saarela. "Multipurpose Web publishing using HTML, XML, and CSS." *Communications of the ACM* 42, no. 10 (1999): 95-101.
- Park, Thomas H., Brian Dorn, and Andrea Forte. "An analysis of HTML and CSS syntax errors in a web development course." *ACM Transactions on Computing Education (TOCE)* 15, no. 1 (2015): 1-21.
- Tilkov, Stefan, and Steve Vinoski. "Node. js: Using JavaScript to build high-performance network programs." *IEEE Internet Computing* 14, no. 6 (2010): 80-83.

- Lingyu, Meng, Christenson Lauren, and Dong Zhijie. "Strategic development of fresh e-commerce with respect to new retail." In 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), pp. 373-378. IEEE, 2019.
- Kang, Chunghan, Junghoon Moon, Taekyung Kim, and Youngchan Choe. "Why consumers go to online grocery: Comparing vegetables with grains." In 2016 49th Hawaii International Conference on System Sciences (HICSS), pp. 3604-3613. IEEE, 2016.
- Mantha, Aditya, Yokila Arora, Shubham Gupta, Praveenkumar Kanumala, Zhiwei Liu, Stephen Guo, and Kannan Achan. "A large-scale deep architecture for personalized grocery basket recommendations." In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3807-3811. IEEE, 2020.
- White, Susan. "Amazon and Whole Foods: adventures in grocery shopping." The CASE Journal (2020).
- Meslin, Halley Rose. "Food Access in the Age of Online Grocery: An Evaluation of Current Retail Trends and Their Potential to Alleviate Food Deserts in the US." IU Journal of Undergraduate Research 4, no. 1 (2018): 58-62.
- Lingyu, Meng, Christenson Lauren, and Dong Zhijie. "Strategic development of fresh e-commerce with respect to new retail." In 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), pp. 373-378. IEEE, 2019.
- Javeed, Arshad. "Performance optimization techniques for ReactJS." In 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-5. IEEE, 2019.
- Soundarya, K., M. Abirami, Kumaran R. Senthil, D. Prabakaran, B. Srimathi, and G. Nagarajan. "Webapp service for booking handyman using mongodb, express JS, react JS, node JS." In 2021 3rd International Conference on Signal Processing and Communication (ICPSC), pp. 180-183. IEEE, 2021.
- Ivanov, Artem, AUFAR ZAKIEV, Tatyana Tsoy, and Kuo-Hsien Hsia. "Online monitoring and visualization with ros and reactjs." In 2021 International Siberian Conference on Control and Communications (SIBCON), pp. 1-4. IEEE, 2021.

- Desai, Vaishnavi, Isha Ghiria, Twinkle Bagdi, and Sanjay Pawar. "KRISHI BAZAAR: An E-Commerce Application For Direct Farmer-to-Consumer Trading." In 2021 IEEE Bombay Section Signature Conference (IBSSC), pp. 1-5. IEEE, 2021.
- Hahn, Jungpil, Robert J. Kauffman, and Jinsoo Park. "Designing for ROI: toward a value-driven discipline for e-commerce systems design." In Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pp. 2663-2672. IEEE, 2002.
- Grover, Varun, and James TC Teng. "E-commerce and the information market." *Communications of the ACM* 44, no. 4 (2001): 79-86.
- Khan, Abdul Wahid, and Siffat Ullah Khan. "Critical success factors for offshore software outsourcing contract management from vendors' perspective: an exploratory study using a systematic literature review." *IET software* 7, no. 6 (2013): 327-338.
- Kaushalya, Thilanka, and Indika Perera. "Framework to Migrate AngularJS Based Legacy Web Application to React Component Architecture." In 2021 Moratuwa Engineering Research Conference (MERCon), pp. 693-698. IEEE, 2021.

## **Chapter 3: System Development**

### **3.1 Overview**

In October 2019 Amazon India launched its first Processing center to have sourcing capabilities from farms and have back end processing solutions to be able to buy in un-packaged, ungraded form and create in-house ready to sell ASIN trans-shipped to now spoke. At the time of launch Inbound capabilities were leveraged from an existing tool available as a part of IHM Kitchen inbound flow which was suboptimal. In this document we will outline the approaches that will facilitate the changes required for new vendor inbound experience in the Processing Centers (PC) and Collection Centers (CC).

### **3.2 Assumptions**

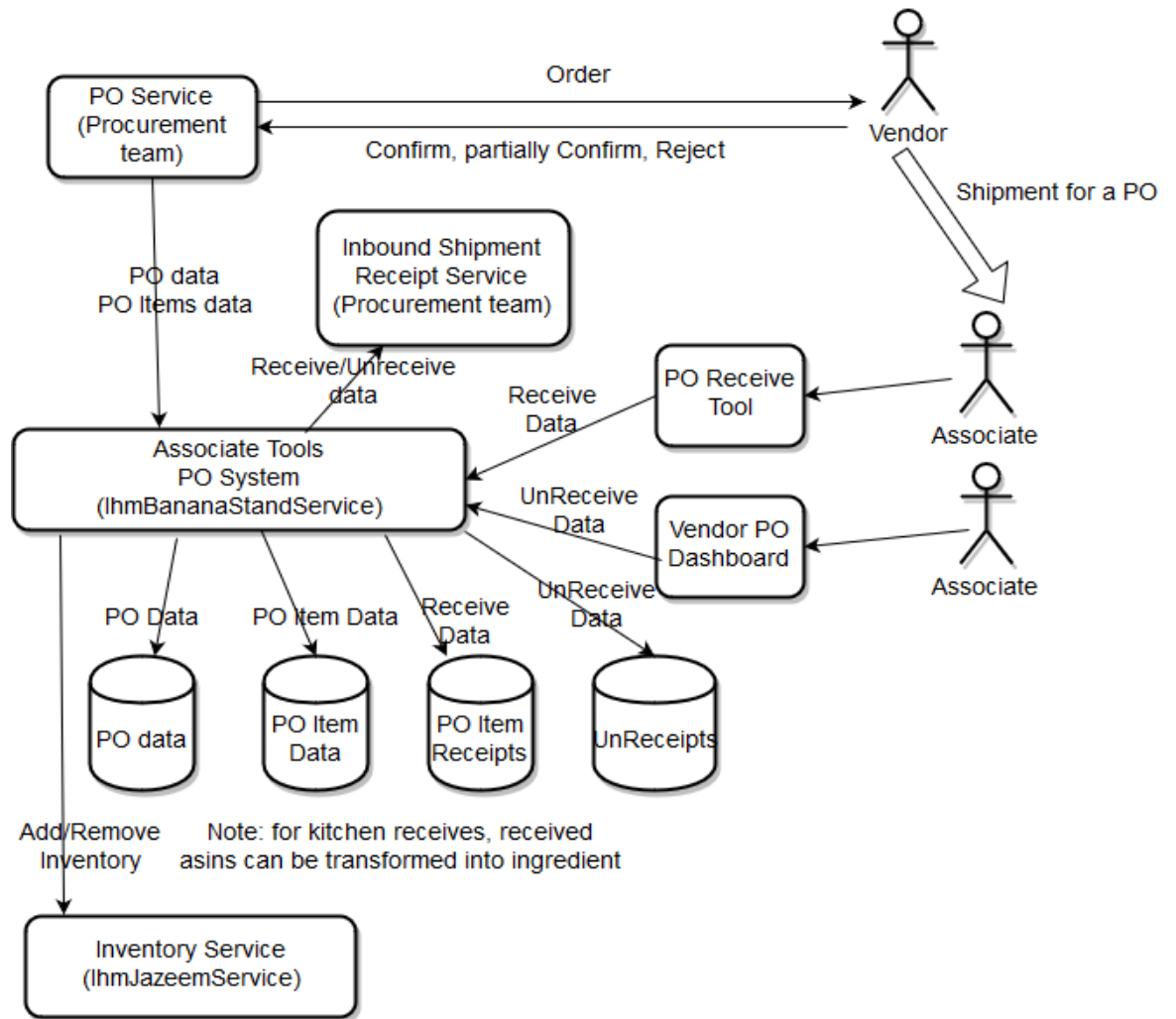
1. The conversion factor defined for phase 1 is 1 unit : 1000 g.
2. As a process inbound for 1 ASIN will have to be done together and will not be done in parts.

### **3.3 Requirements**

1. PO Search API should require both POId and Vendor Code. Associate is required to Scan/manually enter Vendor Code & POId to view all the ASINs to be inbounded for the PO.
2. Need to show Zones(Ambient/Tropical) for each ASIN during Inbound and categorize the ASINs based on zones.
3. Need to show the total weight received for weight based ASINs and total units received for unit-based ASINs when PO details are retrieved during Inbound.
4. We need to show the entered Weight/Unit to the container map when an Associate enters the weight for a scanned container during Inbound.

5. When a user enters the lotContainerCode during POReceive, there should be multiple validations done on it before adding items in it.
6. We need to fetch disassembled relationships for a weight based ASIN during PO Inbound. We need this data to be sent in associate audit and for showing details of weight based ASIN during Inbound.
7. Inbounding of weight based ASINs should be done in weights only and not in units. Users should not be allowed to enter the number of units for a weight based ASIN as it is being done currently.
8. Weight based ASIN received in weight conversion to units for vendor payments for phase 1 based on conversion factor : 1 unit = 1000 grams:  
If weight is greater than 500 grams the units are rounded up and if weight is less than 500 grams the units are rounded down.  
(e.g. if total quantity being received is 100.6 KG vendor payments will be done on 101 units and qty received is 100.1 Kg vendor payments will be done on 100 units)

## Process and Data Flow





## 3.4 Architecture

### 3.4.1 Fetching PO Details during Inbound

#### Approach 1 : Using GradingFacadeService (Preferred)

We will create a new API in GFS, GetPODetails which will act as a facade for fetching information from IhmBananaStand.

Changes in GradingFacadeService :

We will create a new API in GFS, GetPODetails which will act as a facade for fetching information from IhmBananaStand.

#### Sample Input and Output:

GetPODetailsInput

```
{  "storeId": "KISAN-LPC-DEVO",
  "poId": "QA-TEST-74775971-c",
  "vendorCode": "CODE"
}
```

## GetPODetailsOutput

```
{  "po":
  {
    "poId": "QA-TEST-74775971-c",
    "poItems": [
      {
        "fnsku": "BT6S5PC9HQ",
        "quantityConfirmed": 10,
        "quantityReceived": 3,
        "totalWeightReceived": {
          "__type": "com.amazon.ihm.units.coral#DecimalMeasure",
          "unit": "g",
          "value": 3000
        },
        "receiveDetails": [
          {
            "originalReceiveQuantity": 3,
            "receiveQuantityAfterAdjustments": 3,
            "receiveWeightAfterAdjustments": {
              "__type": "com.amazon.ihm.units.coral#DecimalMeasure",
              "unit": "g",
              "value": 3000
            }
          }
        ],
        "disassembleFactor": {
          "__type": "com.amazon.ihm.units.coral#DecimalMeasure",
          "unit": "g",
          "value": 1000
        },
        "disassembleFnsku": "BT6S5PC9ZM",
        "zone": "AMBIENT"
      },
    ],
  },
}
```

```

{
  "fnSku": "BT6S5PBSSF",
  "quantityConfirmed": 10,
  "quantityReceived": 3,
  "totalWeightReceived": {
    "_type": "com.amazon.ihm.units.coral#DecimalMeasure",
    "unit": "g",
    "value": 3200
  },
  "receiveDetails": [
    {
      "originalReceiveQuantity": 1,
      "receiveQuantityAfterAdjustments": 1,
      "receiveWeightAfterAdjustments": {
        "_type": "com.amazon.ihm.units.coral#DecimalMeasure",
        "unit": "g",
        "value": 1250
      }
    },
    {
      "originalReceiveQuantity": 2,
      "receiveQuantityAfterAdjustments": 2,
      "receiveWeightAfterAdjustments": {
        "_type": "com.amazon.ihm.units.coral#DecimalMeasure",
        "unit": "g",
        "value": 1950
      }
    }
  ],
  "disassembleFactor": {
    "_type": "com.amazon.ihm.units.coral#DecimalMeasure",
    "unit": "g",
    "value": 1000
  },
  "disassembleFnsku": "BT6P5TMSLO",
  "zone": "TROPICAL"
}
],
"vendorCode": "CODE"
}
}

```

**Pros:**

1. This change will affect all KISAN systems, so creation of APIs in GFS end will be a good solution if Spyglass deprecates.
2. This change also requires less changes done at the UI end, as we will populate all attributes at GFS's end.

**Cons:**

1. Spyglass is already acting as a facade for calling IhmBananaStand, creating a new API in GFS will add on the load of existing GFS APIs.

**Approach 1 : Using IhmSpyglassService**

In this approach we are retaining the current architecture of GetPODetails PO operation. In this approach we will make changes in the IhmSpyglassService, IhmCrookHook package to support our requirements.

**Changes in IhmSpyglassService :**

We need to add support for StorageFacet and RelationshipFacet in the GetProductInfo API for IhmSpyglass. This change is backward compatible for all IhmSpyglass dependent services. The StorageFacet returns the Zone information set for an ASIN inside catalog data. The RelationshipFacet returns the disassembled factor and disassembled ASIN for the weight based ASIN.

These facet additions require change in IhmRoverCatalogHelper, IhmRoverTypes, IhmSpyglassServiceModel packages to support the facet response in IhmSpyglassService.

## **Changes in IhmCrookHook :**

In the UI end, the KisanPOReceiveTool uses productInfo.js defined inside IhmCrookHook package to get the response from IhmSpyglass GetProductInfo API.

We need to make changes in this package to support the retrieval of StorageFacet and RelationshipFacet from IhmSpyglassService

## **Pros:**

We will achieve all our requirements by making changes in existing POReceive Flow. We do not need to create additional APIs.

## **Cons:**

1. Performing vendor code validation in UI end is a security risk, we should keep the validations at backend only.
2. This change will affect all KISAN systems, so creation of APIs in GFS end will be a good solution if Spyglass deprecates. However since this change will also affect our current tools like Grading, this change can be mitigated if and when Spyglass is deprecated.

## **Fetch total weight Received for an ASIN**

As per the requirements for the new Inbound experience flow we need to show the value of total weight received, for an ASIN when Inbounding against that ASIN in the new POReceiveFlow. These details are fetched from IhmBananaStandService GetPODetails

API. Currently we show the quantity received, and quantity confirmed in each for an ASIN. We will convert quantity received, and quantity confirmed to weight received and weight confirmed using the conversion factor in UI for the weight based ASINs. We do not need any change in IhmBananaStandService as per current assumptions.

### **3.4.2 Receive Against PO**

In the new flow for Receive quantity against PO we will retain the current architecture for POReceive flow. If the ASIN is unit based we will simply pass the number of units in the ReceiveFlow as it happens in current flow. If the ASIN is weight based we will calculate the number of units we will make using the conversion factor, and use the below logic to calculate the number of units per container.

To identify if an ASIN to be received is weight based or Unit based, we will make use of DisassembledTransformationRelationship defined for that ASIN in the catalog.

This relationship is defined only for weight based ASINs.

## **Logic for storing number of units per container**

We will calculate the number of units based on the aggregation of weights over all containers received in one cycle, while maintaining that the variance of units per container is not more than 1.

Example

Let's say we have 3 containers : C1 - 9.6kg , C2-11.6kg , C3-5.2 kg

Total weight : 26.4 kg which will amount to 26 units, so the number of units assigned to each container will be:

C1-10, C2-11, C3-5 total : 26 units

Let's say we have 4 containers : C1 - 9.6kg , C2-11.6kg , C3-5.2 kg, C4- 6.8kg

Total weight : 33.2 kg which will amount to 33 units, so the number of units assigned to each container will be:

C1-10, C2-12, C3-5, C4-6 total : 33 units

## **Using IhmSpyglassService**

We will use GFS GetPODetails response to fetch the disassembled relationship from IhmCharlotte.

To calculate the number of units for a weight based ASIN, we will define the conversion factor in KisanPOReceiveTool and calculate the number of units at UI end, then receive the quantity against that PO using Spyglass ReceiveAgainstPO API.

### **Pros**

We will achieve our requirements using the existing flow of ReceiveAgainstPO.

The unit per container calculation already happens in the current flow in UI.

### **Note**

We would want this conversion factor to come from catalog, and not get stored in UI; this is still a work in progress from Product.



## 3.5 Validations

### 3.5.1 Container validations

For container validation there are 3 requirements -:

Input container should be in Valid format

ContainerId should be valid containerId generated by Feluda

ContainerId shouldn't have been used before

All the 3 requirements are fulfilled by GFS "GetLotContainerDetails" API. When the above three requirements are not met for a container it throws Exception

If the containerCode has been generated by Feluda, then the "assigned" flag identifies whether it has been used or not.

```
<operation name="GetLotContainerDetails">
  <input target="GetLotContainerDetailsRequest" />
  <output target="GetLotContainerDetailsResponse" />
  <error target="DependencyException" />
  <error target="InternalException" />
  <error target="InvalidRequestException" />
  <error target="com.amazon.kisan.coral.exceptions#InvalidLotContainerCodeException" />
  <error target="com.amazon.kisan.coral.exceptions#InvalidLotContainerCodeFormatException" />
</operation>

<structure name="GetLotContainerDetailsRequest">
  <member name="storeId" target="StoreId" />
  <member name="lotContainerCode" target="LotContainerCode" />
</structure>
```

```
<structure name="GetLotContainerDetailsResponse">
  <member name="assigned" target="Boolean" />
  <member name="asin" target="Asin" />
  <member name="fcSku" target="String" />
  <member name="lotContainerItemSource" target="String" />
</structure>
```

If no fcSku is assigned to a given containerId then "assigned" value is false which the UI will use to identify whether containerId is valid and not associated with any FcSku. Whenever a given containerId is entered on the UI, it will invoke this backend API to check whether containerId is valid or not before associating any weight with it.

### 3.5.2 Validations on UI front

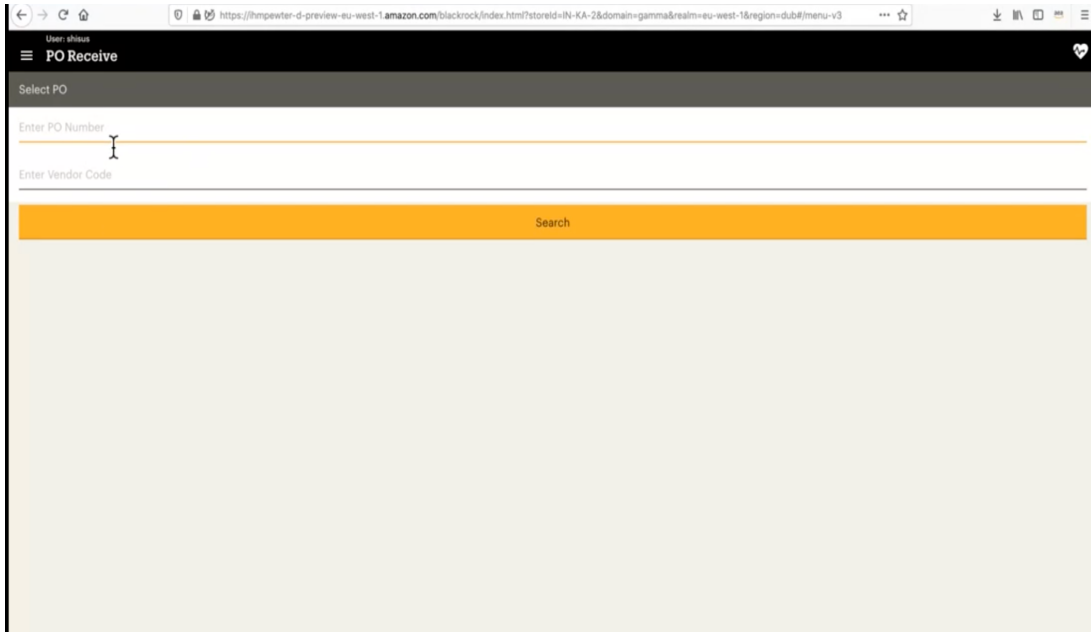
1. ASIN Title information using the ASIN.
2. Rejects events handling will be published as part of UI events. This data will be retrieved from the DW.
3. Deletion of containers, from the UI will be equivalent to reverse of Receive API flow from the backend. This will require deletion of entries from "PurchaseOrderItemReceipts" DDB table. Beside removal of entries from "PurchaseOrderItemReceipts" DDB, it will also require changes in BananaStand to invoke Jazeem for removal of entries. Currently this functionality is not supported by BananaStand and this will require lot of changes in the BananaStand Service for this. Given little value add it offers, we can take care of this feature later on given that it is a large change and require lot of changes in BananaStand service.
4. Warning to be thrown if the quantity being added in the container is beyond XX units or XX gms in a container id. Threshold validation to be there in case receiving quantity is  $XX\% > PO$  quantity. Currently this XX is defined to 20% as confirmed with the Product team.
5. Guardrail Check on value being entered for Reject Quantity.
6. Allow associates to capture reason codes for rejected quantities.
7. Containers validations:
8. Should not allow Decimals in quantity screen
9. Should not allow negative or 0 value on the screen
10. Receive Confirm Button to get activated only if some container has been added for receiving or rejected quantity has been added.

11. Duplicate entry of container not allowed.
12. Weight based ASIN received in weight conversion to units for vendor payments for phase 1 the conversion factor is 1 unit = 1000 grams:
13. If weight is greater than 500 grams the units are rounded up and if weight is less than 500 grams the units are rounded down. (e.g. if total quantity being received is 100.6 KG vendor payments will be done on 101 units and qty received is 100.1 Kg vendor payments will be done on 100 units)

## 3.6 Design

### Old Angular UI vs New Meridian UI:

#### SelectPOController Page: Old Angular UI

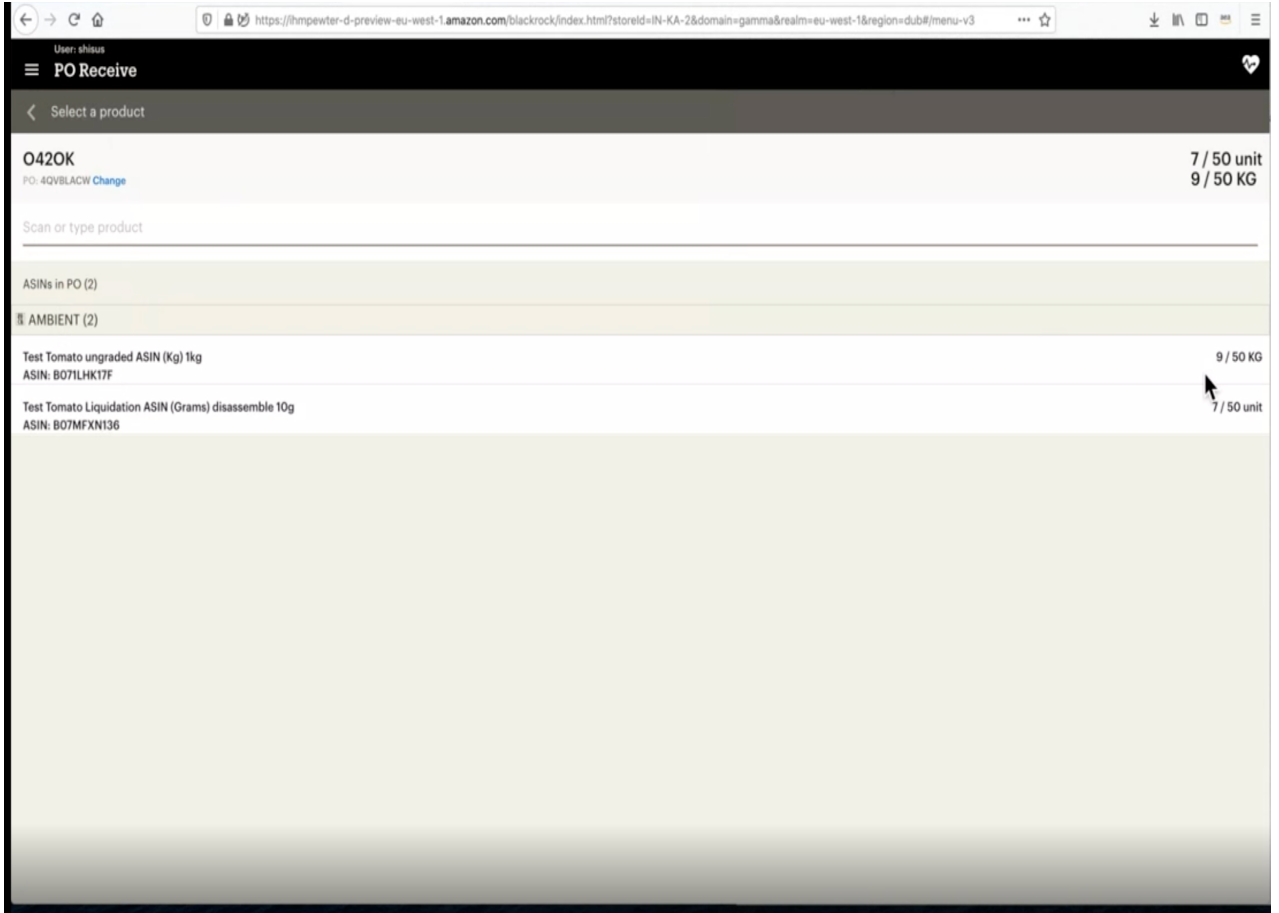


#### New UI



# ScanItemController Page:

Old Angular UI



# New Meridian UI

< Select a Product

**093HID8** 2/19 Units

PO: [IM\\_A\\_PO Change](#) 15/65 KG

Scan or Type Product

ASINs in PO (5)

- ▼ FROZEN (1)
  - Chocolate milk 4/10 KG
  - ITEM\_ID\_3
- › CHILLED (1)
- › AMBIENT (2)
- › UNKNOWN (1)

# ReceivePO Page


## Old Angular UI

User: shisus

PO Receive

Add Container and Quantity

Vendor: O42OK PO: 4QVBLACW

 Test Tomato ungraded ASIN (Kg) 1kg 9/50 KG  
ASIN: B071LHK17F

CN.000.F8D

220

Add

Total Containers: 0 Total Weight: 0 gms


Container ID	Quantity
Enter Rejects 0 gms	
Confirm Receive	
Cancel and go back	



# New Meridian UI

**Add Container and Quantity**

Vendor: NAINAPO: IM\_A\_PO

**ITEM TITLE**  
ASIN: ITEM\_ID\_3

**4/10 KG**

CONTAINER 1

1000

**Add**

Total Containers: 1 Total Weight : 1000 gms

Container ID	Quantity	
CONTAINER 1	1000	<input type="button" value="X"/>

Enter Rejects0 gms

**Confirm Receive**

Cancel and go back

### 3.7 TOOLS

- **AngularJS:** AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJS's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.
- **CSS:** CSS stands for cascading Style Sheets which describes how HTML elements are to be displayed on the screen, pages or other media. It is very important as it helps with the layout control.
- **Visual Studio Code:** Visual studio code is a source code editor developed by Microsoft for windows, linux, and macOS

## Chapter 4: Performance Analysis

When you think about the JavaScript Ecosystem, you'll almost certainly think of Angular and React, as they're two of the most popular front-end development frameworks. But how can you pick between angular vs react? Should you base your selection on the project's specifications first? Or, in the early stages, consider its popularity and ramp-up time?

What's the difference between Angular and React? For years, there has been a violent rivalry between these two prominent front-end frameworks. So, which option is the best?

Every time front-end programming is required, the Angular vs React debate occurs. The answer relies on a variety of circumstances, and even front-end developers have debated the topic for years.

Angular is a Google-developed and maintained web framework that was first released in 2010 under the name AngularJS. It quickly became one of the most popular web frameworks at the time. This was owing to capabilities like two-way data binding and dependency injection, as well as the fact that it was supported by a tech giant.

React, a JavaScript library used by Facebook, was open-sourced in 2013. This framework popularized a web development concept known as component-based architecture, which has a number of benefits, including:

Components that are modular and coherent, making them highly reusable and contributing to a faster development time.

It's used in mobile development since it allows developers to reuse the logical section of an app while simply changing the view.

Easy maintenance and improvement due to self-contained components.

#### Key Features of Angular

- Built-in support for AJAX, HTTP, and Observables are just a few of Angular's highlights. There is widespread support in the community.
- In line with current technologies
- Typescript is time-saving.
- Coding that is more clear and concise
- Error-handling support has been improved.
- Angular CLI allows for seamless updates.
- Validation and forms
- Local CSS / shadow DOM
- Separation of User Interface and Business Logic

#### Key Features of React

- React's key features include the ability to use third-party libraries.
- Time-Saving
- Composability and Simplicity
- Facebook is fully behind you.
- Improved user experience and lightning-fast speed.
- More rapid development
- One-directional data binding provides code stability.
- Components in React

Meridian is built to accelerate your design and development processes. We surveyed our customers and found that teams save Amazon substantial time and money when they use Meridian for their projects and products.

Average time savings of 32%. Source: Meridian Q2 2021 survey.

Median annualized headcount savings of 3 FTEs — 1 design and 2 dev. Source: July 2021 survey of 17 two-pizza teams.

In addition to UI components, Meridian also offers additional tools to help the product development process, increase your teams' velocity, accessibility, and so much more. Use the Compass Sketch plugin for designers aids in the design-development process. Meridian provides illustration and data visualization patterns and components to help your team create a rich experience for your customers. Finally, use the Accessibility Annotation Guide and tools to raise the bar of accessibility for all of your customers.

## **Existing Flow vs New Flow**

The key callouts in the new flow are :

Identification of Unit based/Weight based ASIN in the flow itself.

Calculation of number of units based on the amount of weight entered for a weight based ASIN.

Ability to measure inbound rejects during inbound.

There are two main PO operations in Vendor POReceive flow: Get PODetails and Receive Quantity against a PO.

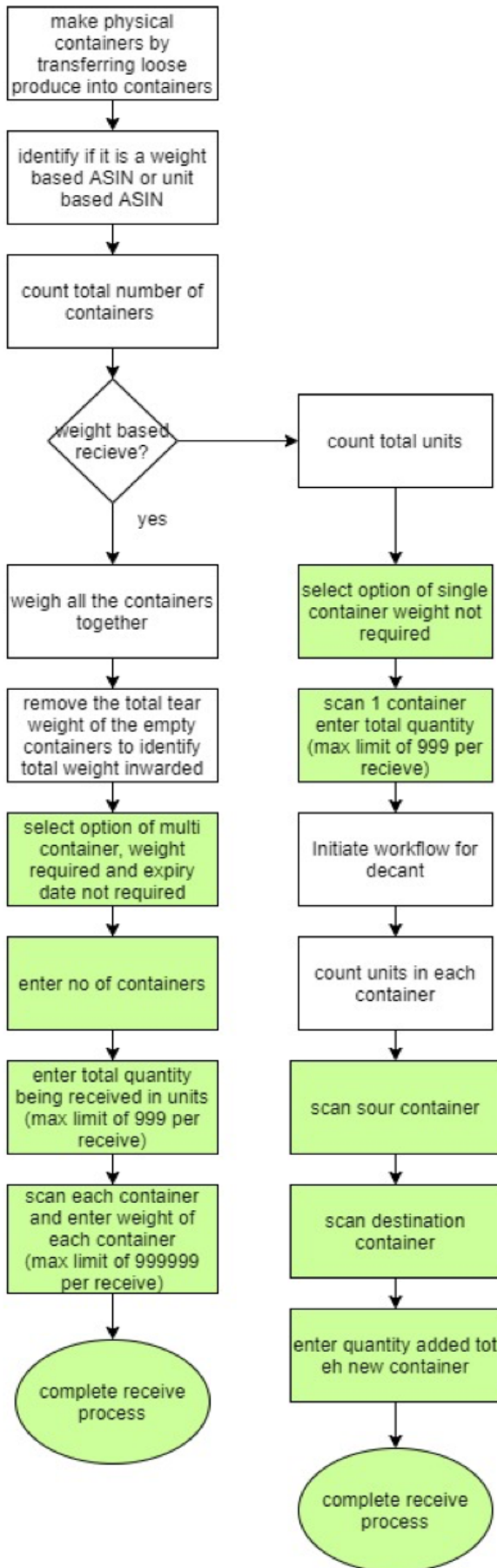
The existing flow of the KisanPOReceiveTool makes use of IhmSpyglassService to call IhmBananaStand APIs GetPODetails and ReceiveAgainstPO to perform these PO operations.

The GetPOdetails operation needs changes in IhmBananaStandService to show the total weight received for an ASIN, so far during POInbound.

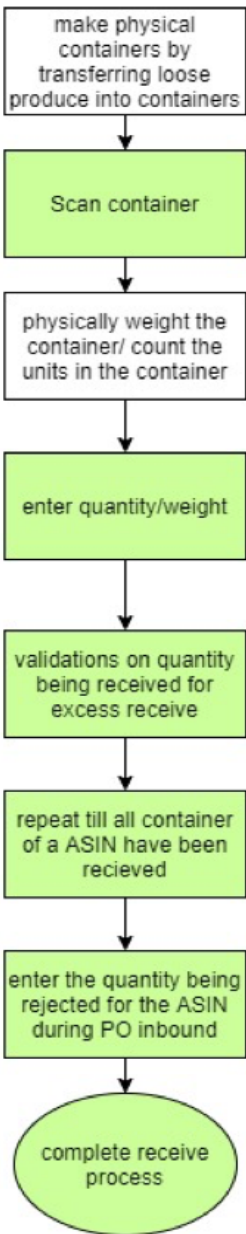
The receive flow for a PO Inbound will not require any backend changes in the new experience.

The old experience allowed us to enter the number of units received for a weight based ASIN, in the new experience we will not allow this entry of units for the weight based ASIN.

Existing Flow



New Flow



## **Chapter 5: Conclusions**

### **5.1 Conclusions**

I am still on the way doing my internship with the Amazon and I have learned so much from this internship offered by the internship, rally helped me in shaping my personality and equipping me with the knowledge of this technologies.

My Final internship project still remains with Amazon and I will give my best in doing the internship project.I like to thank in advance to the coaches, SDM, mentor and trainer of Amazon who guided me through the whole journey of my internship in Amazon and solved all my doubts during the internship. The Coaches, SDM, Mentor and trainer were all of good nature and at every moment helped me when I was doing wrong and shaped me during my whole internship.

Specially my mentor gave his more effort during the internship and passed our all query to the higher authority in the company whether it was related to the reattempt of the assessment, technical issue faced in the assessment or providing extra time to complete the work. I would highly recommend my juniors to prepare well for the offer in Amazon and get the internship opportunity from Amazon because Amazon is a top fortune company in the information technology field.

I would like to thank my TNP officer Mr. Pankaj Kumar and Faculty member Dr. Nafis U khan sir for their support and hard work during the whole placement process because I know how complex the management of the placement drive is.



## 5.2 Future Scope

The Associate Tools are a collection of web-based applications used by associates(1P, 2P and 3P users) in physical stores. There are currently over 100 unique tools that are owned by dozens of teams distributed across multiple orgs in Amazon.

These tools are currently almost exclusively built using AngularJS, which is reaching its end-of-life support timeline on EOY 2021. We found that using AngularJS also reduces team productivity as there is less experience with the framework across the teams that need to build tools and it lacks many features of modern frameworks like React, Angular 2 or Vue. Meridian Framework is a recommended framework to be used in IHM Org[ref]. Since Meridian framework full support is only available in ReactJS, we decided to use ReactJS framework for Associate Tools Platform to support Meridian migration.

Currently these tools are tightly coupled with the framework used in IhmCrookhook which is why the tools migration to latest technologies is difficult. This document will try to address this problem by de-coupling dependency of Associate Tools on IhmCrookhook framework. This document will also propose a plan for React migration for existing tools and IhmSeaward(owned by the IHM APT team).

## REFERENCES

- Amazon Hand book
- Internship experience
- Assessment
- Amazon internship curriculum
- Amazon wiki
- <https://broadcast.amazon.com/videos/372652>
- <https://meridian.a2z.com/components/?platform=react-web>
- <https://docs.angularjs.org/guide>
- <https://ihmassociateplatform.corp.amazon.com/docs/associate-tools/#making-tool-changes>
- <https://w.amazon.com/bin/view/NinjaDevSync/>
- <https://builderhub.corp.amazon.com/docs/black-caiman/user-guide/>
- <https://builderhub.corp.amazon.com/docs/dev-setup/laptop-macos.html>
- <https://meridian.a2z.com/>
- <https://reactjs.org/>
- [https://w.amazon.com/bin/view/Lily/Development/Metrics/React\\_Migration\\_Project/React\\_Best\\_Practices](https://w.amazon.com/bin/view/Lily/Development/Metrics/React_Migration_Project/React_Best_Practices)