Retail Store Stock Inventory Analytics

PROJECT REPORT

TEAM MEMBERS:

K.T.VENKATESHKARTHIK
MD AEJAZ AF
S.AJIT
N. KISHOREKUMAR

1.INTRODUCTION

1.1 Project Overview

Retail Store Stock Inventory analytics is basically used to maintain the inventory. The biggest FAQ's of a retailer are "How much amount of inventory should they carry", because this involves Capital and Operational costs. Lack of these costs will lead into the complex problem of making loss of costs, damaged brand which makes the Customer unhappy. Forecasting intermediate inventory and tracking is a complex problem to be solved because the stock rotation depends seasonally.

More than just switching from pen and paper to digital software is necessary to address all of the issues that might result from delayed, ineffective, incomplete, and error-filled retail inventory. Whether as an addition to your current enterprise resource planning (ERP) package or as part of a stand-alone, centralised procurement system, selecting an inventory management solution backed by both automation and data science improves every element of how retailers handle inventory.

1.2 Purpose

- Identifying Consumer Demands
- Management of Merchandise
- Convenience of timing.
- Automation of Real-time dashboard.
- Predication based on sales history of seasonal on-demands.
- Automatically determine the goods and service taxes like GSTs etc.
- Discount Prediction Sytem creation.
- Periodic generation of inventory reports
- Creation unique barcodes for the products toenhance the billingprocess.
- Instant invoice generation for the purchase

2. LITERATURE SURVEY

2.1 Existing problem

Inventory data management deals with large collection stock related data in the supply chain management environment. The frequency of data collection is very high in terms of stock volume. Content analysis management play savital role in managing the stock data in order to classify and cluster in terms managing the data. The process of data classification and clustering willkeep track on the stock in order to fulfill the customer need on demand.

2.2 References

1] R. Sheth, M. Vora, R. Sharma, M. Thaker and P. Bhavathankar, "A Proficient Process for Systematic Inventory Management," 2020 International Conference for Emerging Technology (INCET), 2020, pp. 1-5, doi:10.1109/INCET49848.2020.9154038.

[2] Macas, Cinthya & Aguirre, Jorge & Arcentales-Carrion, Rodrigo & Pena, Mario. (2021), “Inventory management for retail companies: A literature review and current trends”. 71-78. 10.1109/ICI2ST51859.2021.00018.

[3] J.C.F. Ehrenthal et.al, “Demand seasonality in retail inventory management”, European Journal of Operational Research, Volume 238, Issue 2,2014, Pages 527-539, ISSN 0377-2217, https://doi.org/10.1016/j.ejor.2014.03.030.

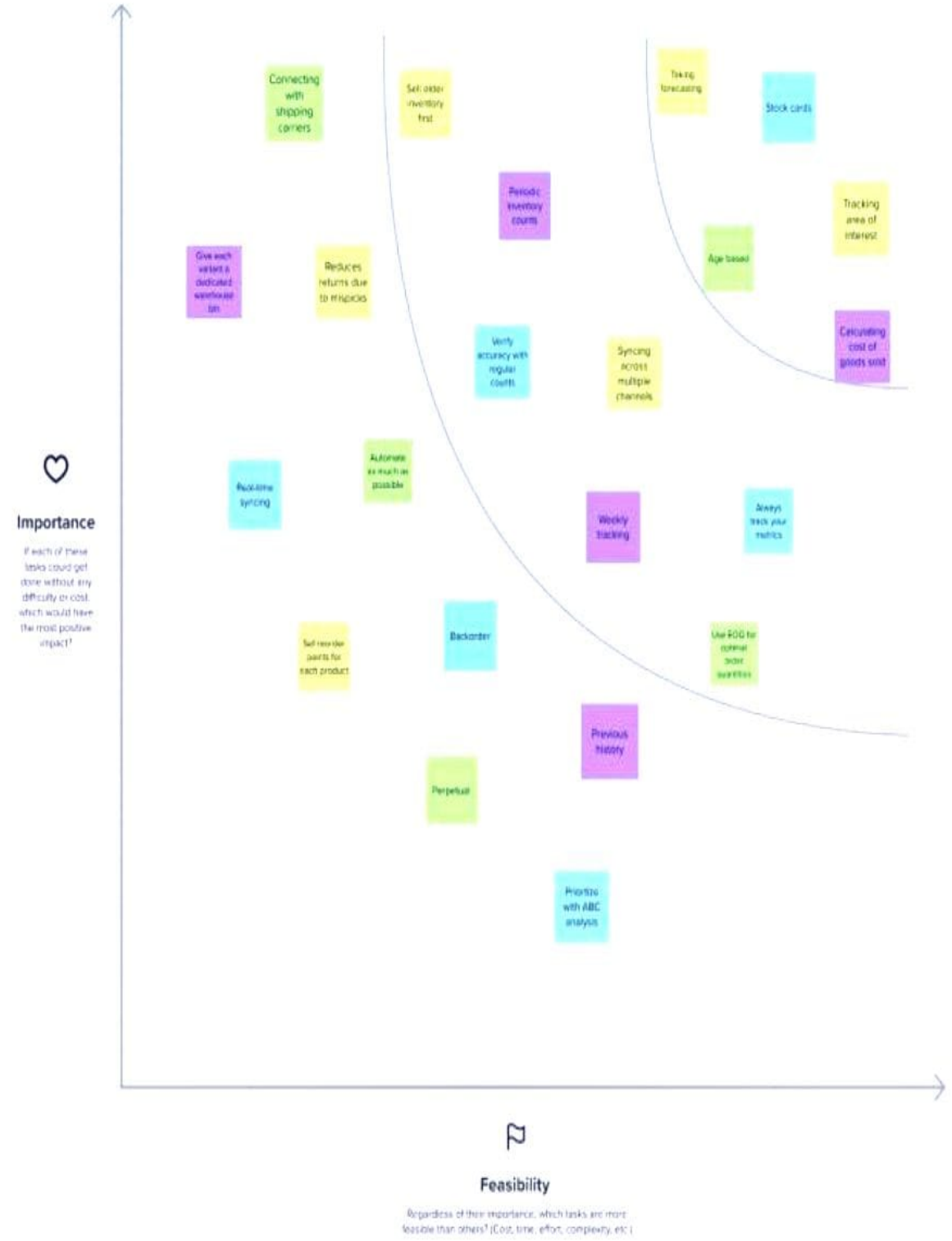
2.3 Problem Statement Definition

The problem faced by the retail store is they do not have any systematic system to record and keep their inventory data. It is difficult for the admin to record the inventory data quickly and safely because they only keep it in the logbook and not properly organized.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

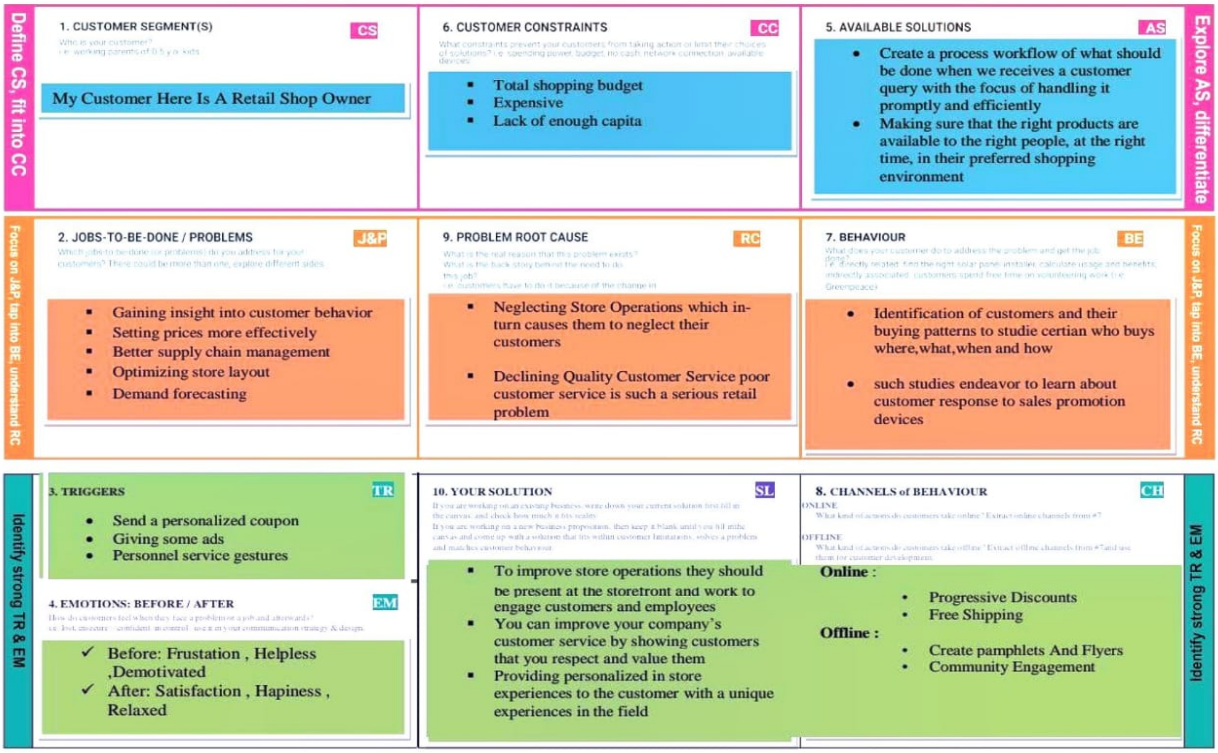
3.2 Ideation & Brainstorming


Connecting with shipping carriers
Sell older inventory first
Taking forecasting
Stock cards
Periodic inventory counts
Tracking area of interest
Age based
Give each variant a dedicated warehouse bin
Reduces returns due to impacts
Calculating cost of goods sold
Verify accuracy with regular counts
Syncing across multiple channels
Automate as much as possible
Real-time syncing
Weekly tracking
Always track your metrics
Backorder
Use ROP for optimal order quantities
Set reorder points for each product
Previous history
Perpetual
Prioritize with ABC analysis
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 Proposed Solution

S. No	Parameter	Description
1.	Problem Statement (Problem to be solved	The problem faced by the retail store is they do not have any systematic system to record and keep their inventory data. It is difficult for the admin to record the inventory data quickly and safely because they only keep it in the logbook and not properly organized
2.	Idea / Solution description	The goal is to utilize the given data set about the Retail Store Stock Inventory and store the data in the cloud .So the retail store can use this information to easily predict the inventory easily and quickly
3.	Novelty / Uniqueness	Complete a thorough analysis of our store; it leads to avoiding overstock and also analysis of the competitive relevant market. Gathering customer feedback and measuring our business results.
4.	Social Impact / Customer Satisfaction	When customers get the products they want faster with fewer mistakes or out-ofstocks, it increases customer loyalty
5.	Business Model (Revenue Model)	Ad based Revenue modelAwareness can be created for Optimize the use ofinventory, reduce handling cost, optimize cash flow

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	• Registration through Form • Registration through Gmail • Registration through LinkedIN
FR-2	User Confirmation	• Confirmation via Email • Confirmation via OTP

FR-3	User Login	• Login via form • Login via Email
FR-4	Stock Updating	• Adding/Removing Stocks • Generating barcodes for products
FR-5	Stock Management	• Predicting Out of stock and less brought stocks • Periodic generation of reports using Cognos analytics tool • Notification of the product regarding out of stock and expiry
FR-6	Billing Management	• Fast billing through barcode • Quick generation of invoice after tax calculation • Discounts based on credit points

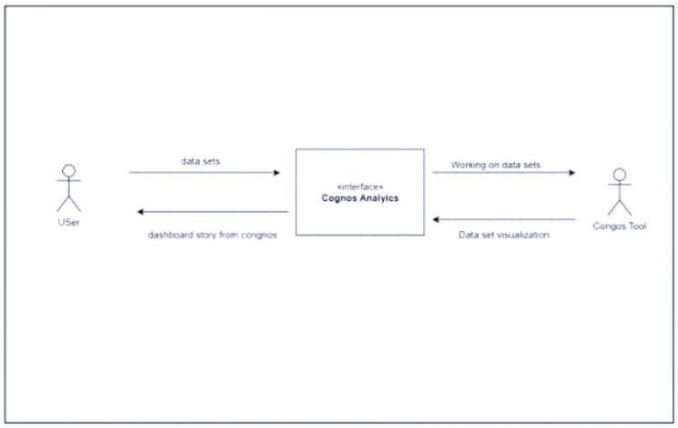
4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	• Prediction of stocks reduce out of stocks and cost • Compatible with mobile and desktop

NFR-2	Security	• Authentication of users to defend against other users • People without proper credentials cannot access the application
NFR-3	Reliability	• Provides Accurate Stock prediction • Reduce loss for retailer • Notification system for expiry and out of stock products
NFR-4	Performance	• Instant invoice generation using b arcode • Improved accuracy in sales prediction • Real time report generation • Working on large sets of data
NFR-5	Availability	• Accessible by all devices • Suitable for all kind of retail stores • Real time visibility into stock levels
NFR-6	Scalability	• Many Retailers can access the application without any issues • Allows adding of new features

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

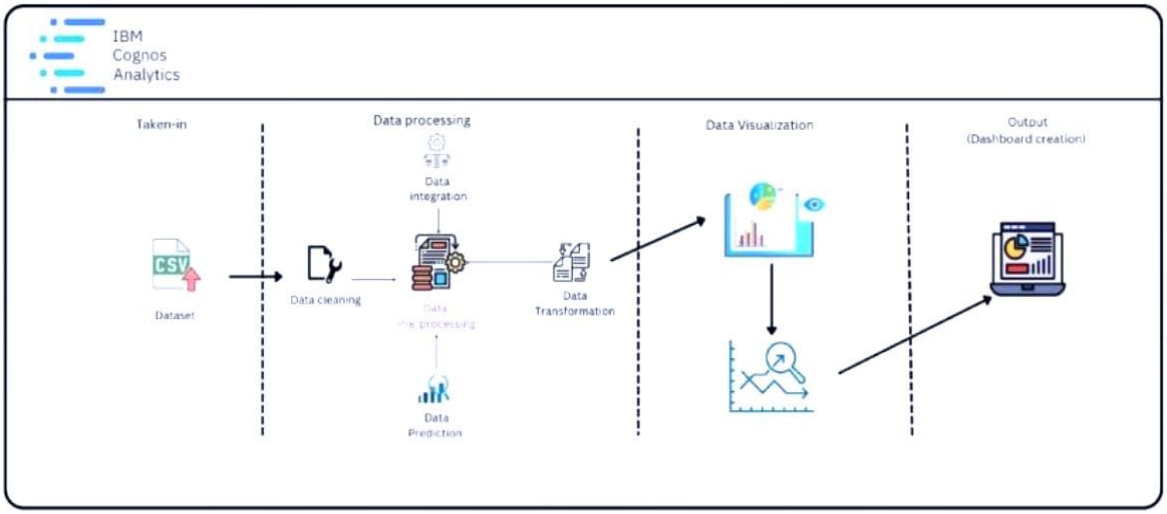


Table-1 : Components & Technologies:

s. no	component	Description	Technology
1.	User Interface	The user interacts with application using Web UI	SS, JavaScript
2.	Data Processing	The data from the dataset is pre-processed	IBM Cognos analytics
3.	Cloud Database	The clean dataset is stored on IBM Cloud	IBM Cloud
4.	Data visualization	The data is visualized into different forms	IBM Cognos Analytics, Python

Table-2: Application Characteristics:

s. no	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	IBM Cognos Analytics, Python
2.	Security Implementations	Request authentication using Encryptions	Encryptions
3.	Scalable Architecture	scalability of architecture 3 – tier	Web Server – HTML, CSS, Javascript Application Server – Python Database Server – IBM Cloud
4.	Availability	The application is available for cloud users	IBM Cloud Hosting
5.	Performance	The user can know how to maintain the inventory to increase profits	ML algorithms

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Relese
Customer (PC user)	Registration	USN-1	As a user, I can register for the Cognos by entering my email, password, and confirming my password	I can access my account / dashboard	High	sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	sprint-1
		USN-3	As a user, I can register for the application through IBM id	I can register & access the dashboard with IBM id Login	Low	sprint-2
		USN-4	As a user, I can register for the application through Gmail		media	sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	sprint-1

	Dashboard	USN-6	As a user, I can upload the dataset for visualization Dataset	Dataset gets uploaded in Cognos analytics	media	sprint2
		USN-7	User must pre-process the dataset before visualization		High	sprint-2
		USN-8	User can choose various types of visualizations from the available options		High	sprint-2
		USN-9	User can analyze the charts to make better business decisions		High	sprint-3
		USN-10	User can make use of the visualizations to make reports		media	sprint-3
		USN-11	user can send feedback regarding the platform	Cognos accepts the feedback given by the user	Low	sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

sprint	Functional Requirement(Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
sprint-1	Data Collection	USN-1	The dataset is collected and the understanding of dataset is done to present the analytics to the user	2	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N
sprint-1	Data Preparation	USN-2	As a user, I can view the accurate analytics of data by prepared data. The data preparation is done to restructure and clean the data	3	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N

sprint-2	Data Exploration	USN-3	As a user, I can view the visualized datato get thebetterunderstandingabout the sales stock, revenue and price.	8	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N
sprint-3	Dashboard Creation	USN-4	As a user, I can view the different visualization in the dashboard about the sales, stock, revenue and price	8	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N
sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
sprint-4	Report creation	USN-5	As a user, I can view the detailed report of the sales, stock, revenue and price. The user can get the report of the particular data.	8	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N
sprint-4	Story creation	USN-6	As a user, I can view the story to get the better understanding of the sales, stock, revenue and price. The user can make decisions based on the story.	8	High	K.T.venkateshkarthik Md Aejaz AF Ajit S KishoreKumar N

6.2 Sprint Delivery Schedule

sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
sprint-1	5	6Days	24 Oct 2022	29 Oct 2022	5	29 Oct 2022
sprint-2	8	6Days	31 Oct 2022	05 Nov 2022	8	05 Nov 2022
sprint-3	8	6Days	07 Nov 2022	12 Nov 2022	8	12 Nov 2022
sprint-4	6	6Days	14 Nov 2022	19 Nov 2022	16	19 Nov 2022

Velocity:

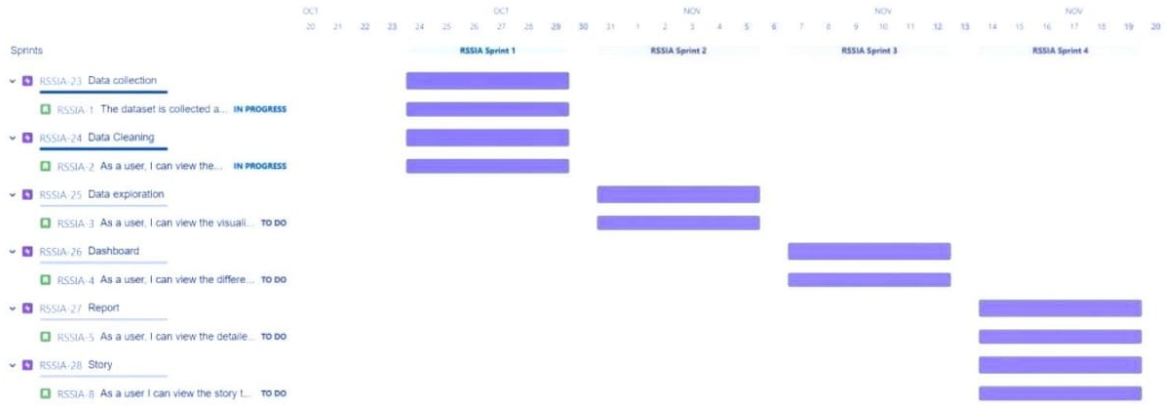
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) periteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{20} = 2$$

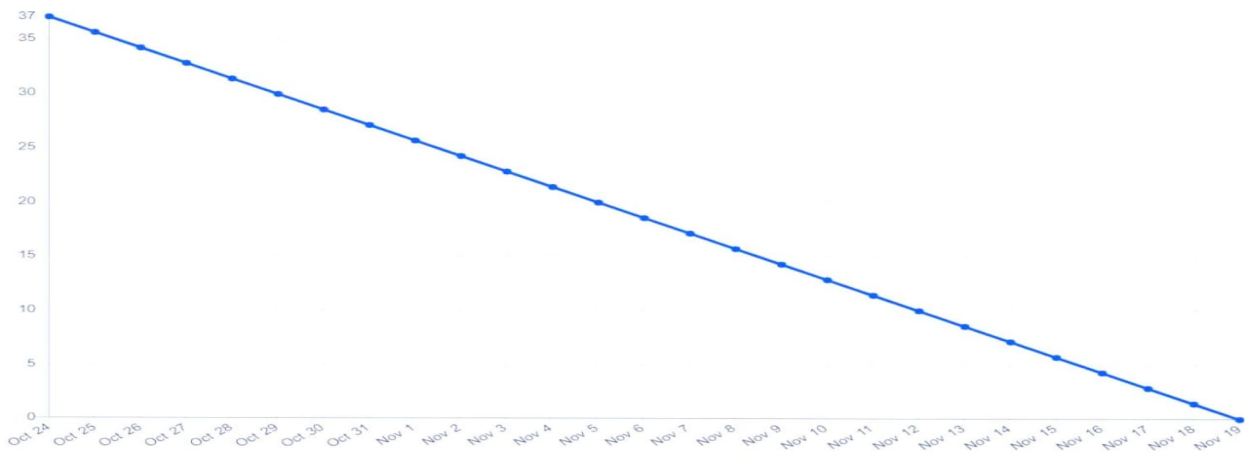
velocity 10

sprint	Story points	Duration	Average velocity
sprint-1	5	6	0.83
sprint-2	8	6	1.33
sprint-3	8	6	1.33
sprint-4	16	6	2.66
Total	37	24	1.54

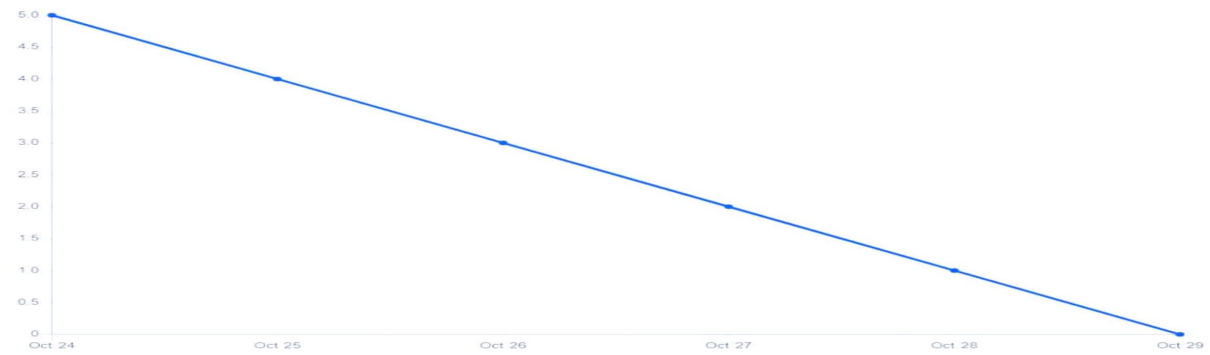
6.3 Reports from JIRA



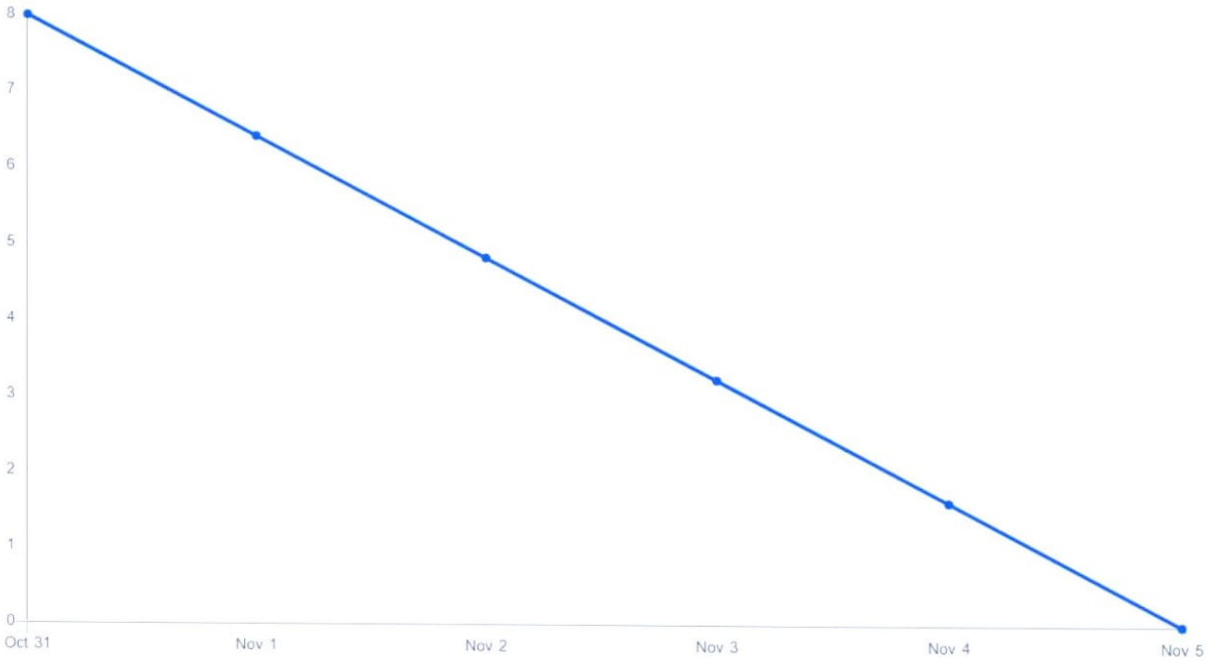
Burndown Chart:



Sprint-1



Sprint-2



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

```
def Login(request):
    if request.method == "POST":
        email = request.POST['email']


        user = authenticate(request, username = email, password = password)
        if user is not None:
            form = login(request,user)
        return redirect("/")
  else:
      return render(request,"Login.html",{'status':'Bad Cred if you are the new user
Register First'})
```

```
else:
  return render(request,"Login.html")
```

1 7.2 Feature2

```
def register(request):
  if request.method == "POST":
      name = request.POST['full-name']
        sname = request.POST['store-name']
      location  = request.POST['location']
            email = request.POST['your-email']
    password = request.POST['password']
         repassword = request.POST['comfirm-password']
   if password == repassword:
         user = User.objects.create_user(username = sname,password =
   password,first_name = sname,email = email)
   user.save()
        userinfo = extendedProfileInfo.objects.create(email = email,location =
   location,manager_name = name)
     userinfo.save()
  return redirect("/")
 else:
  return render(request,"registration.html"
```

2 7.3 Database Schema (if Applicable)

TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	DATA_TYPE	IS_NULLABLE
ibm	dataset	id	bigint	NO
ibm	dataset	mart_name	varchar	YES
ibm	dataset	name	varchar	YES
ibm	dataset	product_id	varchar	YES
ibm	dataset	Expriydate	varchar	YES

ibm	dataset	cp	varchar	YES
ibm	dataset	stock	int	YES
ibm	rsim_extende dprofilei nfo	id	bigint	NO

ibm	rsim_extended profilei nfo	manager_name	varchar	NO
ibm	rsim_regularsa les	id	bigint	NO
ibm	rsim_regularsa les	mart_name	varchar	NO
ibm	rsim_regularsa les	prod_name	varchar	NO
ibm	rsim_regularsa les	sales	int	NO
ibm	rsim_regularsa les	date	varchar	NO
ibm	sales_data	slno	int	NO
ibm	sales_data	mart_nae	varchar	NO
ibm	sales_data	year	int	NO
ibm	sales_data	month	int	NO

8. TESTING

8.1 Test Cases

Test case	feature Type	comp onent	steps to Execute	Expected Result	Actual Result	Status

Homepage _TC_001	Functional	Home Page	1.Enter URL and click go2.Click on MyAccount drop down button 3.Verify login/Singup popup displayed or	Login/Sign up popup should display	Login/Sign up popup should display	pass
LoginPage _TC_002	UI	Home page	1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link e.Last	Application should show below UI elements: a.email text box b.password text box c.Login button with orange colour d.New customer? Create account link e.Last password?	Working as expected	Fail
			password? Recovery password link	Recovery password link		
LoginPage _TC_003	functional	login page	1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter InValid username/ema il in Email text box 4.Enter valid password in password text box 5.Click on login button	Application should show 'Incorrect email or password ' validation message.	Working as expected	pass
LoginPage _TC_004	Functional	Dashboard page	Verified user is able to view the dashboard	Application should show dashboard and visualizati	Working as expected	pass

			and their shop details and visualization	on		
LoginPage _TC_OO5	Functional	Report page	Verified user is able to the periodic data report	Application generates report	Working as expected	pass
LoginPage_ O6	Functional	Update and Addition of produc	Verified user is able to the add and update stocks	Application will add and update the stocks	Working as expected	pass

8.2 User Acceptance Testing:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtota
By Design	10	4	2	3	19
Duplicate	2	0	3	0	5
External	3	3	0	1	7
Fixed	11	4	4	20	39
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	3	1	9
Totals	24	14	13	26	77

This report shows the number of test cases that have passed, failed, and untested

section	Total case	Not Tested	fail	pass
Print Engin	6	0	0	6
clientApplication	50	0	0	50
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	0	10
finalReport Output	4	0	0	4
VersionControl	2	0	0	2

9. RESULTS

9.1 Performance Metrics

A measure of mistakes between paired observations reflecting the same phenomena in

statistics is called Mean Absolute Error (MAE). Comparisons of expected against observed data, subsequent time against initial time, and one measuring technique against an alternate measurement technique are a few examples of Y vs X. The MAE is determined by dividing the total absolute errors by the sample size.

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

● The advantage of the retail inventory approach does not demand a physical inventory is a benefit.
● The retail inventory approach entails merely recording the retail prices of inventory items.
● The retail inventory method merely needs a company to keep track of the retail

DISADVANTAGES

● The retail inventory approach is only accurate if all pricing is consistent and changes occur at the same rate.
● In most circumstances, this is not feasible in retail due to the wide range of goods pricing. Depreciation, markdowns, product damage, and theft, for example, can all have an impact on the retail inventory price.
● As a result, any calculations generated using the retail inventory approach should only be used as a rough approximation.

11.CONCLUSION

Inventory management is a complex but necessary component of the supply chain. An efficient inventory management system aids in the reduction of stock-related costs such as warehousing, carrying, and ordering.

12. FUTURE SCOPE

● Prediction Accuracy Enhancement for better prediction.
● Creation of Credit Point System to manage discount points for every purchase.
● AI Chatbot support to resolve queries.
● Language Conversion to connect people worldwide.

13. APPENDIX

Source Code

```
from django.db import models
# Create your models here.
```

```
class Dataset(models.Model):
mart_name = models.CharField(max_length=9, blank=True, null=True)
 name = models.CharField(max_length=15, blank=True, null=True)
 product_id = models.CharField(max_length=10, blank=True, null=True)
expirydate = models.CharField(db_column='Expirydate', max_length=11, blank=True, null=True)
# Field name made lowercase.
cp = models.CharField(max_length=4, blank=True, null=True)
  sp = models.CharField(max_length=4, blank=True, null=True)
  stock = models.IntegerField(blank=True, null=True)
class Meta:
 db_table = 'dataset'
class extendedProfileInfo(models.Model):
 email = models.EmailField(unique=True)
location = models.CharField(max_length = 20)
  manager_name = models.CharField(max_length = 30)
class SalesData(models.Model):
 slno = models.AutoField(db_column='SLNO', primary_key=True)  # Field name made
lowercase.
 mart_name = models.CharField(max_length=20)
year = models.IntegerField()
 month = models.IntegerField()
  product = models.CharField(max_length=30)
 sales = models.CharField(max_length=6)
 product_id = models.CharField(db_column='product ID', max_length=10)  # Field
name made lowercase. Field renamed to remove unsuitable characters.
class Meta:
  managed = False
 db_table = 'sales_data'
class RegularSales(models.Model):
 mart_name = models.CharField(max_length=20)
 prod_name = models.CharField(max_length=40)
sales = models.IntegerField()
 date = models.CharField(max_length = 10)
prediction.py
import pandas as pd
import numpy as np
from datetime import date
import matplotlib.pyplot as plot
 from reportlab.lib.utils import ImageReader
from reportlab.pdfgen.canvas import Canvas
from reportlab.platypus import SimpleDocTemplate, Table, TableStyle
```

```
from reportlab.pdfbase.ttfonts import TTFont
 from reportlab.pdfbase.pdfmetrics import registerFont from reportlab.lib.pagesizes import A4
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics import io
from .models import SalesData,RegularSales
def Predict(prod_id,m_name):
df = pd.DataFrame(list(SalesData.objects.values('mart_name','month','sales','product_id')))
 df = df[(df.product_id == prod_id)]
df = df[(df.mart_name == m_name)]
todays_date = date.today()
 df.drop(['mart_name','product_id'], axis=1,inplace = True)
X = np.array(df['month']).reshape(-1, 1)
y = np.array(df['sales']).reshape(-1, 1)
 X_train, X_test, y_train,
y_test = train_test_split(X, y, test_size=0.09, random_state=0)
regr = LinearRegression()
regr.fit(X_train, y_train)
y_pred = regr.predict(X_test)
 y_test = y_test.astype(np.float)
 print('Mean Absolute Error (MAE):', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error (MSE):', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error (RMSE):',
np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
mape = np.mean(np.abs(np.subtract(y_test,y_pred) / np.abs(y_test)))
print('Mean Absolute Percentage Error (MAPE):', round(mape * 100, 2))
print('Accuracy:', round(100*(1 - mape), 2))
month = [todays_date.month - 1, todays_date.month, todays_date.month + 1,
todays_date.month + 2]
 m = np.array(month).reshape(-1,1)
 y_pred = regr.predict(m)
# print(y_pred)
 y_pred = list(y_pred)
dict = []
 pred_val = []
for i in range(len(m)):
 dict.append(month[i] % 12 if month[i] > 12 else month[i])
 pred_val.append(int(y_pred[i]))
return(round(100*(1 - mape), 2),dict,pred_val)
def visualize(m_name):
df = pd.DataFrame(list(RegularSales.objects.values('mart_name','prod_name','sales','date')))
```

```
 df = df[(df.mart_name == m_name)]
df.drop(['mart_name'], axis=1,inplace = True)
print(df.head())
plot.barh(df["prod_name"],df["sales"])
plot.savefig('myplot.png', dpi=1000)
def createPDF(name,mart_name):
canv = Canvas('text-on-image.pdf',pagesize=A4)
img = ImageReader('myplot.png')
registerFont(TTFont('arial','C:\\windows\\fonts\\arial.ttf'))
  x = 120
 y = 490
 w = 389
 h = 236
 canv.setFont("Courier", 24)
 canv.drawString(240, 800, "REPORT")
 canv.setFont("Courier", 16)
 canv.drawString(200, 750, "Name :")
 canv.drawString(300, 750, name)
 canv.drawString(200, 730, "MartName:")
canv.drawString(300, 730, mart_name)
canv.drawImage(img,x,y,w,h,anchor='sw',anchorAtXY=True,showBoundary=False)
 canv.setFont('arial',14)
 canv.setFillColor((1,0,0)) #change the text color
 canv.drawCentredString(x+w*0.5,y+h*0.5,")
 canv.drawString(240, 480, "Products VS Sales")
 width = 600
 height = 400
 val = RegularSales.objects.all().values()
 data = [['Slno','Product Name', 'Sales', 'Date']]
  j = 0
 for i in val:
data.append([j,i['prod_name'],i['sales'],i['date']])
 j = j + 1
x = 190
y = 250
f = Table(data)
f.wrapOn(canv, width, height)
f.drawOn(canv, x, y)
canv.save()
```

urls.py

```
from django.urls import path, include
from . import views
urlpatterns =[
path("",views.index,name="dr"),
path("login",views.Login,name="login"),
path("register",views.register,name="register"),
path("new_product",views.productInfo,name="productInfo"),
path("existingproduct",views.existingPro,name="existingPro"),
path("update",views.update,name="update"),
path("billing",views.billing,name="billing"),
 path("modDB",views.modDB,name="modDB"),
path("report",views.displayreport,name="dr"),
path("logout",views.log_out,name="lo"),]
```

views.py

```
 from django.shortcuts import render,redirect,HttpResponse
 from django.db import connection
 from barcode import Code39
 from datetime import date
 from django.http import JsonResponse,FileResponse
 from django.views.decorators.csrf import csrf_exempt
 from django.core.paginator import Paginator,EmptyPage, PageNotAnInteger
 from barcode.writer import ImageWriter
 from .models import Dataset,extendedProfileInfo,RegularSales
 from django.http import FileResponse
 from django.contrib.auth.decorators import login_required
 from django.contrib.auth import authenticate, login,logout
 from django.contrib.auth.models    import User,auth
 from . import prediction
 import io,requests,os
cache = {
"item_cnt":"1",
 }
def generate_barcode(string,pname):
my_code = Code39(string, writer=ImageWriter(),add_checksum = False)
my_code.save(pname)
f = open(pname+".png","rb")
return f.read()
def register(request):
if request.method == "POST":
```

```
name = request.POST['full-name']
sname = request.POST['store-name']
location  = request.POST['location']
email = request.POST['your-email']
password = request.POST['password']
repassword = request.POST['comfirm-password']
if password == repassword:
user = User.objects.create_user(username = sname,password = password,first_name =
sname,email = email)
 user.save()
  userinfo = extendedProfileInfo.objects.create(email = email,location = location,manager_name
= name)
 userinfo.save()
return redirect("/")
else:
 return render(request,"registration.html")
@login_required(login_url='login')
def productInfo(request):
if request.method == "POST":
 name = request.POST['name']
 expirydate = request.POST['edate']
 stock = request.POST['qty']
 cp = request.POST['cp']
 sp = request.POST['sp']
 cursor = connection.cursor()
 cursor.execute('"SELECT DISTINCT product_id,1 id FROM dataset"')
 row = cursor.fetchall()
 prodID = "PRD"+str(len(row) + 1)
 f = generate_barcode(prodID,name)
 buffer = io.BytesIO(f)
 buffer.seek(0)
 Dataset.objects.create(mart_name =
request.user.username,name=name,product_id=prodID,expirydate=expirydate,cp=cp,s
p=sp,stock = stock).save()
response = FileResponse(buffer, as_attachment=True, filename=name+'.png')
return response
else:
return render(request,"PI.html")
@login_required(login_url='login')
def existingPro(request):
dta = request.GET.get('keywords')
```

```
 val = []
 page = request.GET.get('page', 1)
 if dta != None:
 values = Dataset.objects.filter(name__iregex=dta,mart_name =request.user.username)
  integer = 1
 for value in values:
val.append({"id":integer,"name":value.name,"qty":value.stock,"sp":value.sp,"expdate":va
lue.expirydate,"pid":value.product_id})
  integer = integer + 1
 else:
values = Dataset.objects.filter(mart_name = request.user.username)
  integer = 1
for value in values:
val.append({"id":integer,"name":value.name,"qty":value.stock,"sp":value.sp,"expdate":va
lue.expirydate,"pid":value.product_id})
    integer = integer + 1
  paginator = Paginator(val, 5)
 try:
users = paginator.page(page)
  except PageNotAnInteger:
  users = paginator.page(1)
 except EmptyPage:
users = paginator.page(paginator.num_pages)
return render(request,"EP.html",{'val':users})
def Login(request):
if request.method == "POST":
email = request.POST['email']
   password = request.POST['password']
 user = authenticate(request, username = email, password = password)
      if user is not None:
  form = login(request,user)
   return redirect("/")
else:
return render(request,"Login.html",{'status':'Bad Cred if you are the new user Register First'})
 else:
return render(request,"Login.html")
def displayreport(request):
 if request.method == "POST":
 manager_name = extendedProfileInfo.objects.get(email = request.user.email).manager_name
prediction.createPDF(manager_name,request.user.username)
 return FileResponse(open('text-on-image.pdf', 'rb'), content_type='application/pdf')
```

```
   else:
     prediction.visualize(request.user.username)
      return render(request,"report-view.html")
 def update(request):
  prod_id = request.GET.get('prodid')
if request.method == "POST":
cp = request.POST['cp']
 sp = request.POST['sp']
 stock = int(request.POST['stock'])
  exp = request.POST['expiryDate']
  post = Dataset.objects.get(product_id=prod_id)
   post.sales = int(post.sales) + stock
   post.cp = cp
  post.sp = sp
  Post.expiryDate = exp
 post.save()
else:
accuracy,mon_values,pred_values =
prediction.Predict(prod_id,request.user.username)
 v = Dataset.objects.filter(product_id = prod_id,mart_name = request.user.username).values()[0]
return
 render(request,"update_records.html",{'cp':v['cp'],'sp':v['sp'],'stock':v['stock'],'acc':accura
cy,"months":mon_values,"pred":pred_values})
 @login_required(login_url='login')
@csrf_exempt
def billing(request):
res = list(Dataset.objects.filter(mart_name =
request.user.username).values_list('name',flat=True))
   if request.method == 'POST':
 val = request.POST['search_val']
  res = Dataset.objects.filter(mart_name = request.user.username,name = val).values()
  resp = []
  for i in res:
     resp.append({'id':cache["item_cnt"],'name':i["name"],'price':i["sp"]})
  cache["item_cnt"] =  int(cache["item_cnt"]) + 1
return JsonResponse({'name':resp})
 else:
cache["item_cnt"],cache['total'] = "1",0
  return render(request,"testbilling.html",{'name':res})
 def sendEmail(product,email,count):
 url = 'https://api.emailjs.com/api/v1.0/email/send'
```

```
  header = {
  'contentType': 'application/json'
 }
  myobj = {
    'service_id': 'service_r43q87l',
 'template_id': 'template_0dqkujg',
   'user_id': 'lYYw2FI0jGkE59hnh',
   'template_params':
{
  'product':product,
   'count':count,
 'mart_owner': email,
'to_name': extendedProfileInfo.objects.get(email = email).manager_name
     }
 }
 x = requests.post(url,headers=header, json = myobj)
   return(x.text)
def modDB(request):
 for i in request.GET:
 res = Dataset.objects.get(mart_name = request.user.username,name = i)
sendEmail(i,request.user.email,request.GET.get(i))
 RegularSales.objects.create(mart_name = request.user.username,prod_name = i,sales =
request.GET.get(i),date = str(date.today()))
    res.stock = int(res.stock) - int(request.GET.get(i))
  res.save()
 return redirect("billing")
def index(request):
  if request.user.is_authenticated:
    return render(request,"index.html",{'auth':"True"})
 else:
 return render(request,"index.html",{'auth':"False"})
 @login_required(login_url='login')
 def log_out(request):
   logout(request)
return redirect("/")
```

GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-3888-1658669795

Project Demao Link:

https://www.youtube.com/embed/d6AQAZVUaJk