**Project Report Format**

### **Topic :** AI-Based Localization And Classification Of Skin Disease With Erythema

**Team Members:**

1. **VIJAYALAKSHMI.K (Team Leader)**
2. **AKILA.K.B**
3. **NANDHINI.A.R**
4. **RAMYA .T.I**

**1. Introduction**

**1.1 Project Overview**

Skin diseases are more common than other diseases. Skin diseases may be caused by fungal infection, bacteria, allergy, or viruses, etc. The advancement of lasers and Photonics based medical technology has made it possible to diagnose the skin diseases much more quickly and accurately. But the cost of such diagnosis is still limited and very expensive. So, image processing techniques help to build automated screening system for dermatology at an initial stage. The extraction of features plays a key role in helping to classify skin diseases. Computer vision has a role in the detection of skin diseases in a variety of techniques. Due to deserts and hot weather, skin diseases are common in Saudi Arabia. This work contributes in the research of skin disease detection. We proposed an image processing-based method to detect skin diseases. This method takes the digital image of disease effect skin area, then use image analysis to identify the type of disease. Our proposed approach is simple, fast and does not require expensive equipment other than a camera and a computer. Then resize the of the image to extract features using pretrained convolutional neural network. After that classified feature using Multiclass SVM. Finally, the results are shown to the user, including the type of disease, spread, and severity. The system successfully detects 3 different types of skin diseases with an accuracy rate of 100%.

**1.2 Purpose**

To overcome the above problem we are building a model which is used for the prevention and early detection of skin cancer, psoriasis. Basically, skin disease diagnosis depends on the different characteristics like colour, shape, texture etc. Here the person can capture the images of skin and then the image will be sent the trained model. The model analyses the image and detect whether the person is having skin disease or not.

**2.Literature Survey**

**2.1 Existing problem**

Skin diseases may be caused by fungal infection, bacteria, allergy, or viruses, etc. The advancement of lasers and Photonics based medical technology has made it possible to diagnose the skin diseases much more quickly and accurately. But the cost of such diagnosis is still limited and very expensive.

**2.2 References**

[1] Arifin, S., Kibria, G., Firoze, A., Amini, A., & Yan, H. (2012) “Dermatological Disease Diagnosis Using Color-Skin Images.” Xian:International Conference on Machine Learning and Cybernetics.

[2] Yasir, R., Rahman, A., & Ahmed, N. (2014) “Dermatological Disease Detection using Image Processing and Artificial Neural Network.“Dhaka: International Conference on Electrical and Computer Engineering.

[3] Santy, A., & Joseph, R. (2015) “Segmentation Methods for Computer Aided Melanoma Detection.” Global Conference on Communication Technologies.

[4] Zeljkovic, V., Druzgalski, C., Bojic-Minic, S., Tameze, C., & Mayorga, P. (2015) “ Supplemental Melanoma Diagnosis for Darker Skin Complexion Gradients.” Pan American Health Care Exchanges

[5] Suganya R. (2016) “An Automated Computer Aided Diagnosis of Skin Lesions Detection and Classification for Dermoscopy Images.”International Conference on Recent Trends in Information Technology.

[6] Alam, N., Munia, T., Tavakolian, K., Vasefi, V., MacKinnon, N., & Fazel-Rezai, R. (2016) “Automatic Detection and Severity Measurement of Eczema Using Image Processing.” IEEE.

[7] Kumar, V., Kumar, S., & Saboo, V. (2016) “Dermatological Disease Detection Using Image Processing and Machine Learning.” IEEE.

[8] Krizhevsky, A., ILYA, S., & Geoffrey, E. (2012) “ImageNet Classification with Deep Convolutional Neural Networks.” Advances in Neural Information Processing Systems.

[9] Cristianini, N., Shawe, J., “Support Vector Machines”, 2000.

[10] SOMMERVILLE, I., “Software Engineering”. 9th .2011.

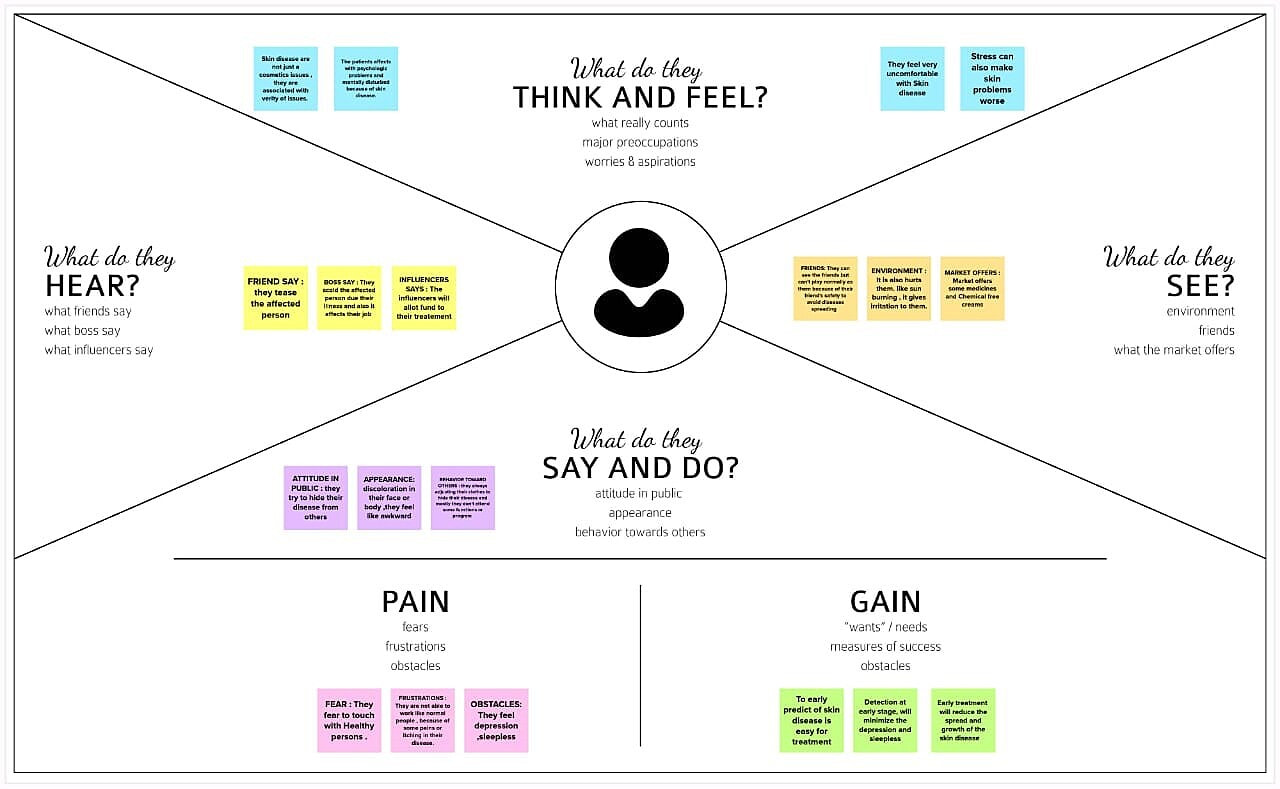
**2.3 Problem Statement Definition**

We have clearly used image processing methods to accomplish the mission of pre-processing. The classification tasks can be made much easier and more productive by defining this pattern.

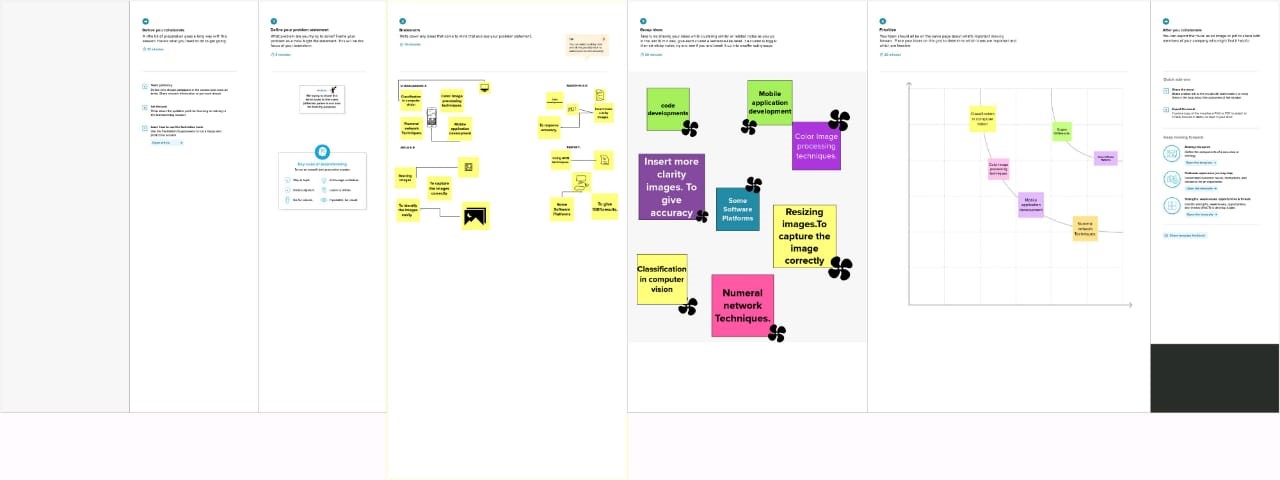
We present a fully automated classification system in this article for diagnosing skin disorders by using image mining techniques. Our model is designed to compromise pixel, object and pattern levels in three steps, respectively. We have used multiple image mining techniques likeaugmentation, feature extraction, classification in order to classify skin disorder images effectively.

**3. Ideation and Proposed Solution**

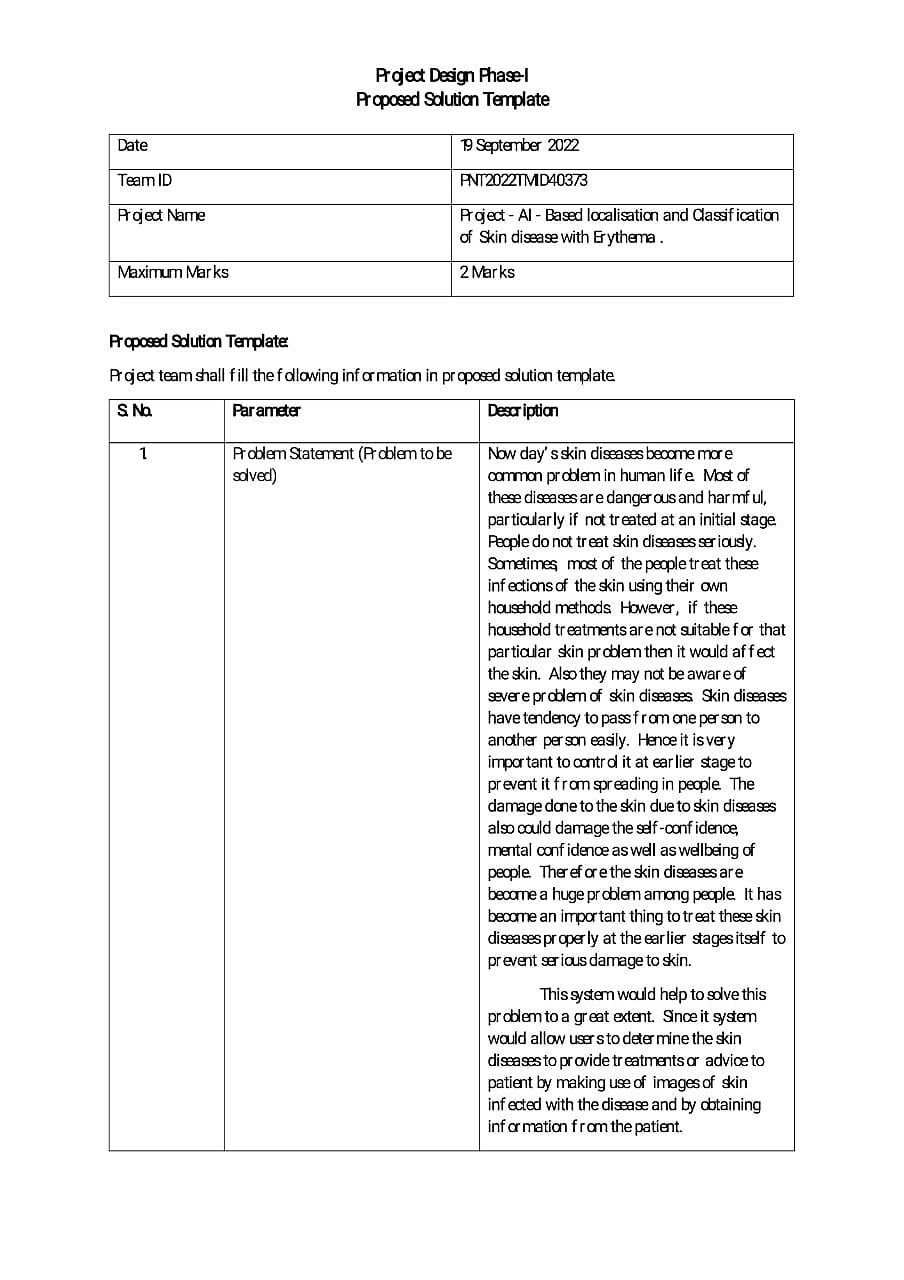
**3.1 Empathy Map Canvas**



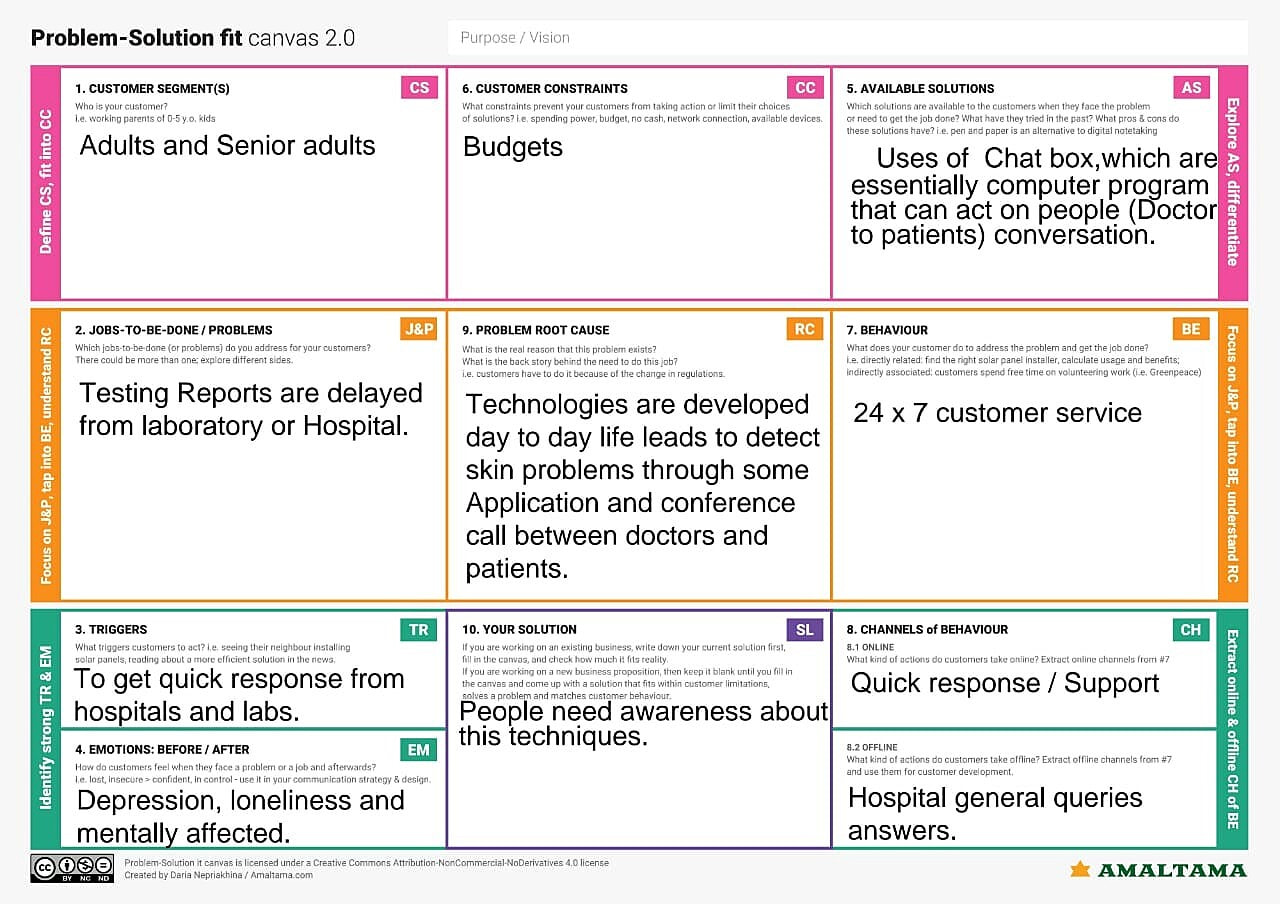
**3.2 Ideation and Brainstorming**



**3.3 Proposed Solution**



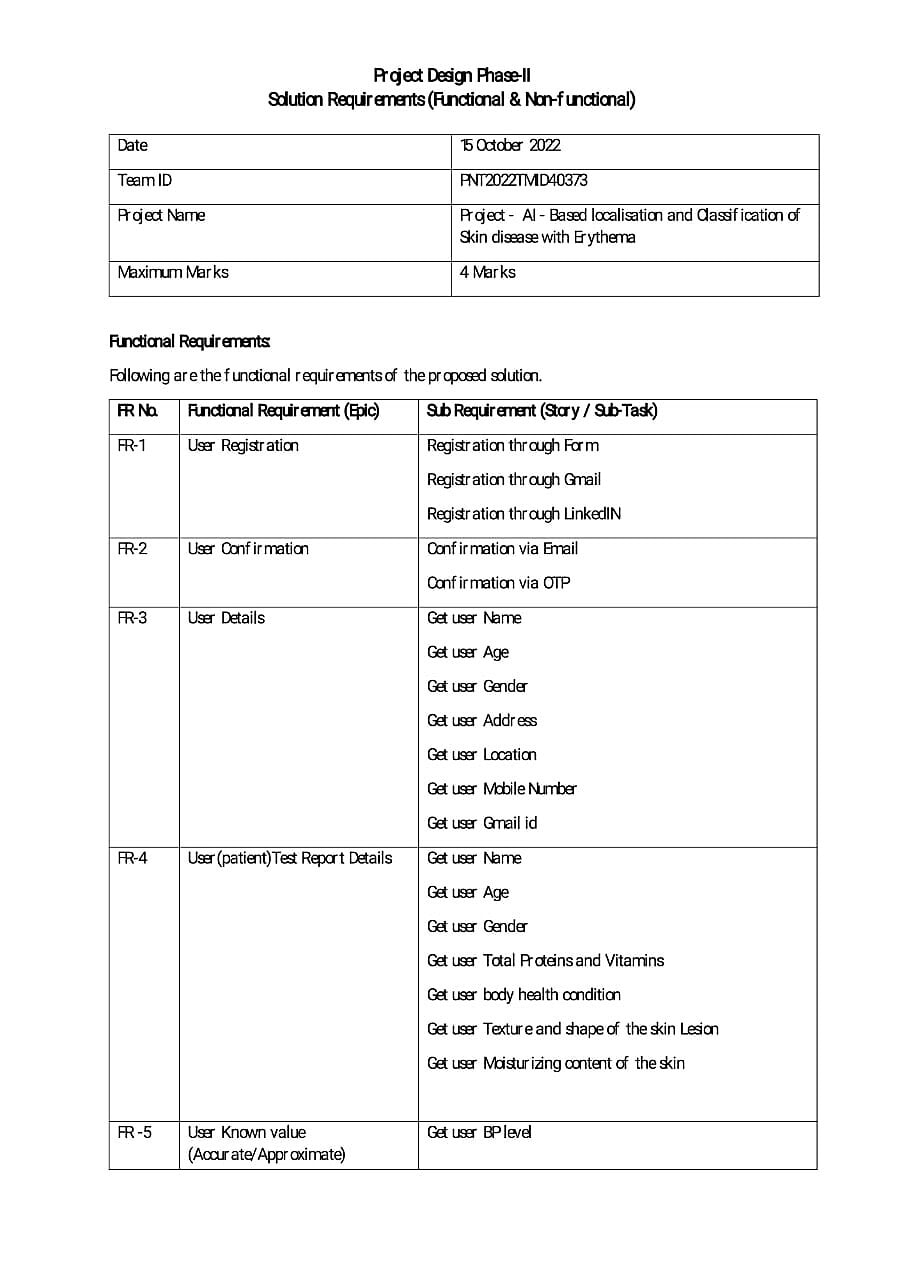
**3.4 Problem Solution fit**

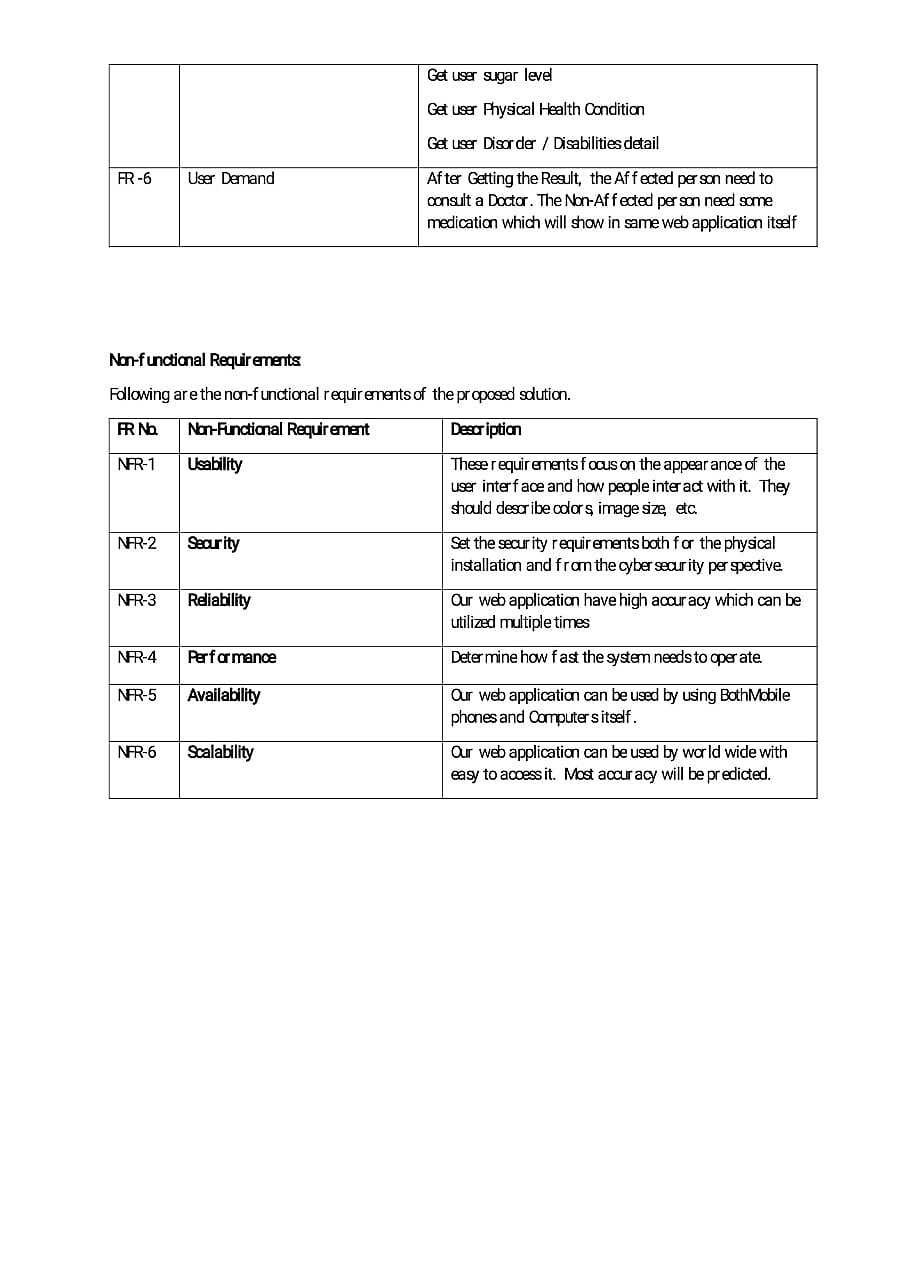


**4. Requirement Analysis**

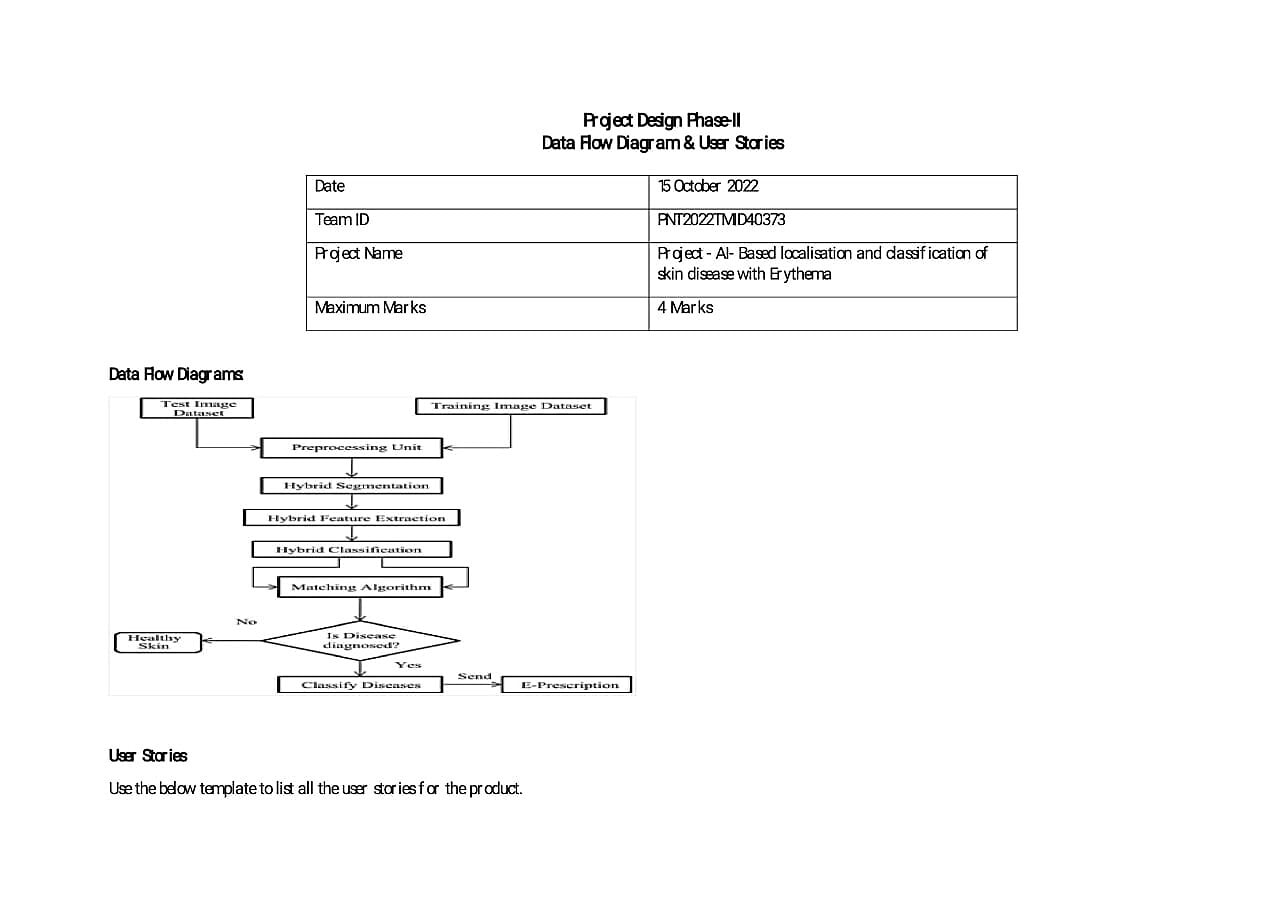
**4.1 Functional requirement**

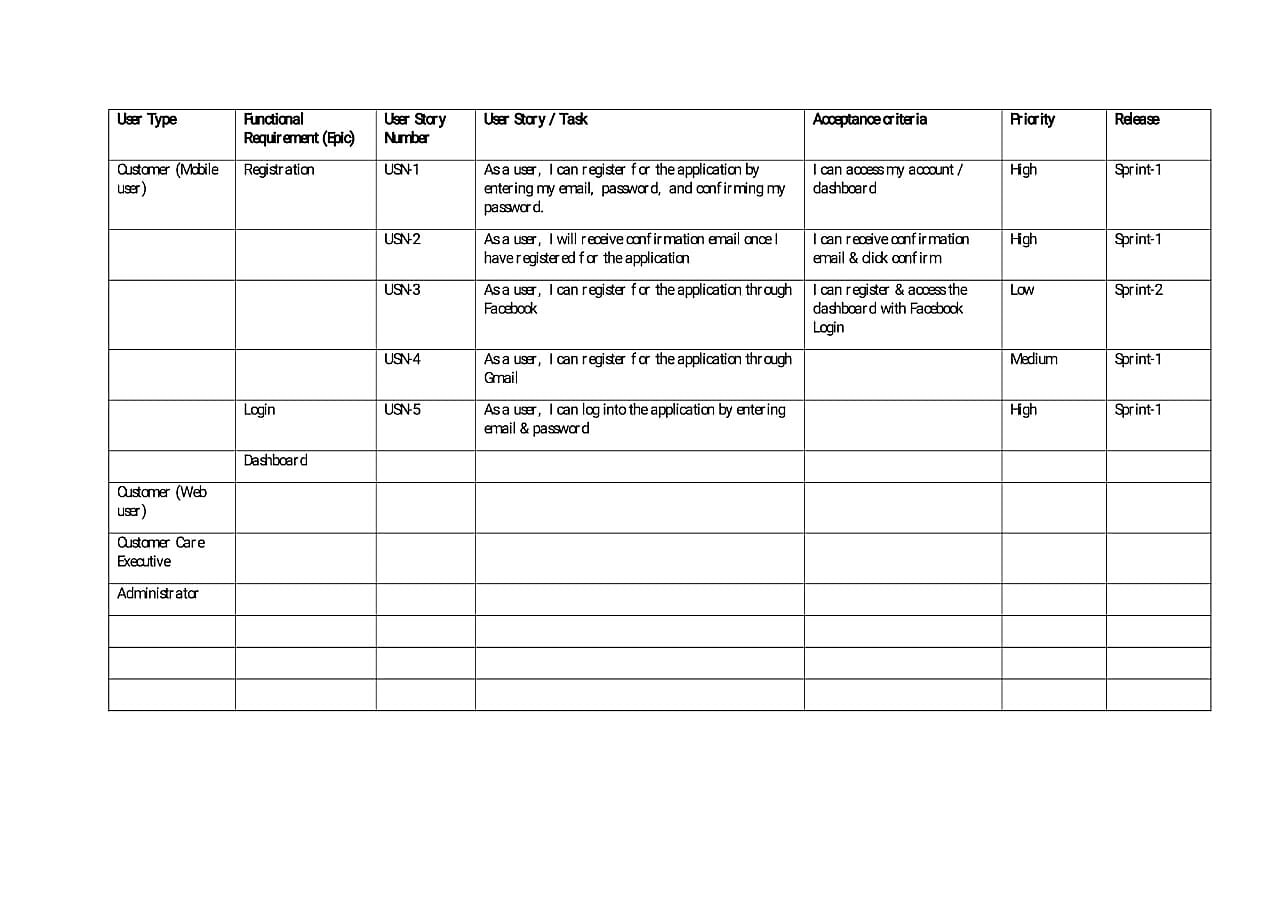
**4.2 Non-Functional requirements**



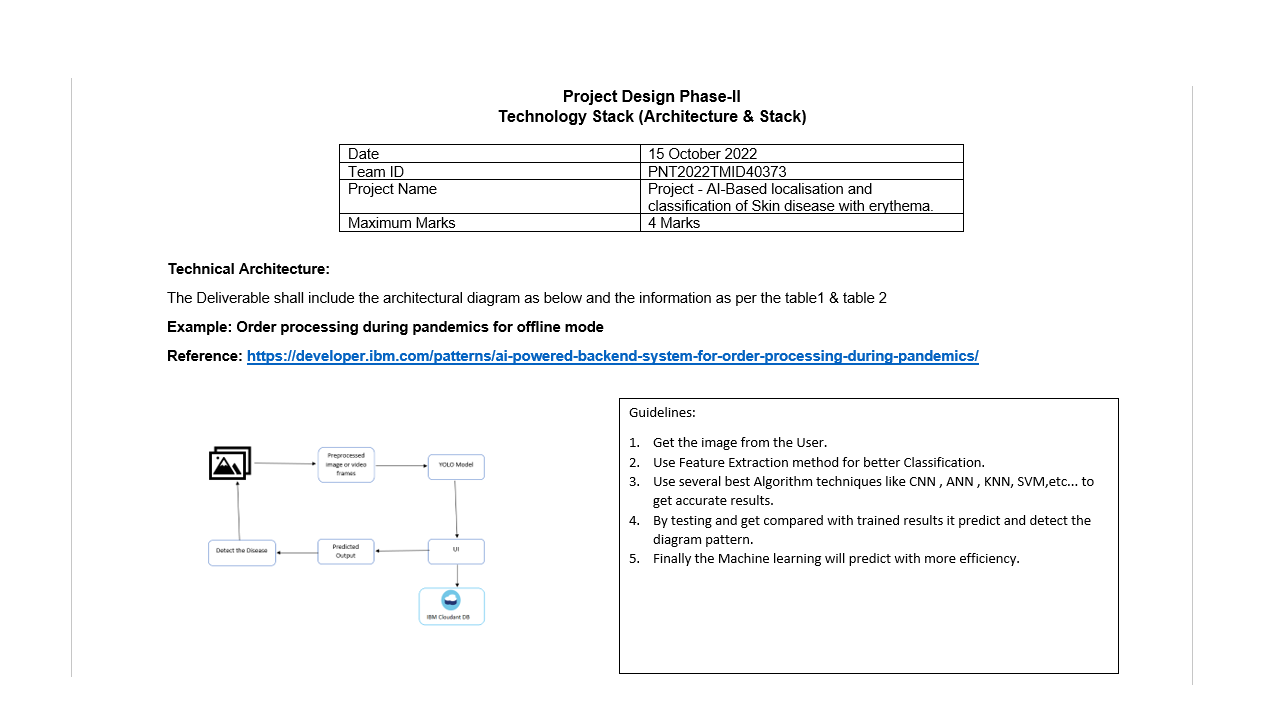
**5.Project Design**

**5.1 Data Flow Diagrams**



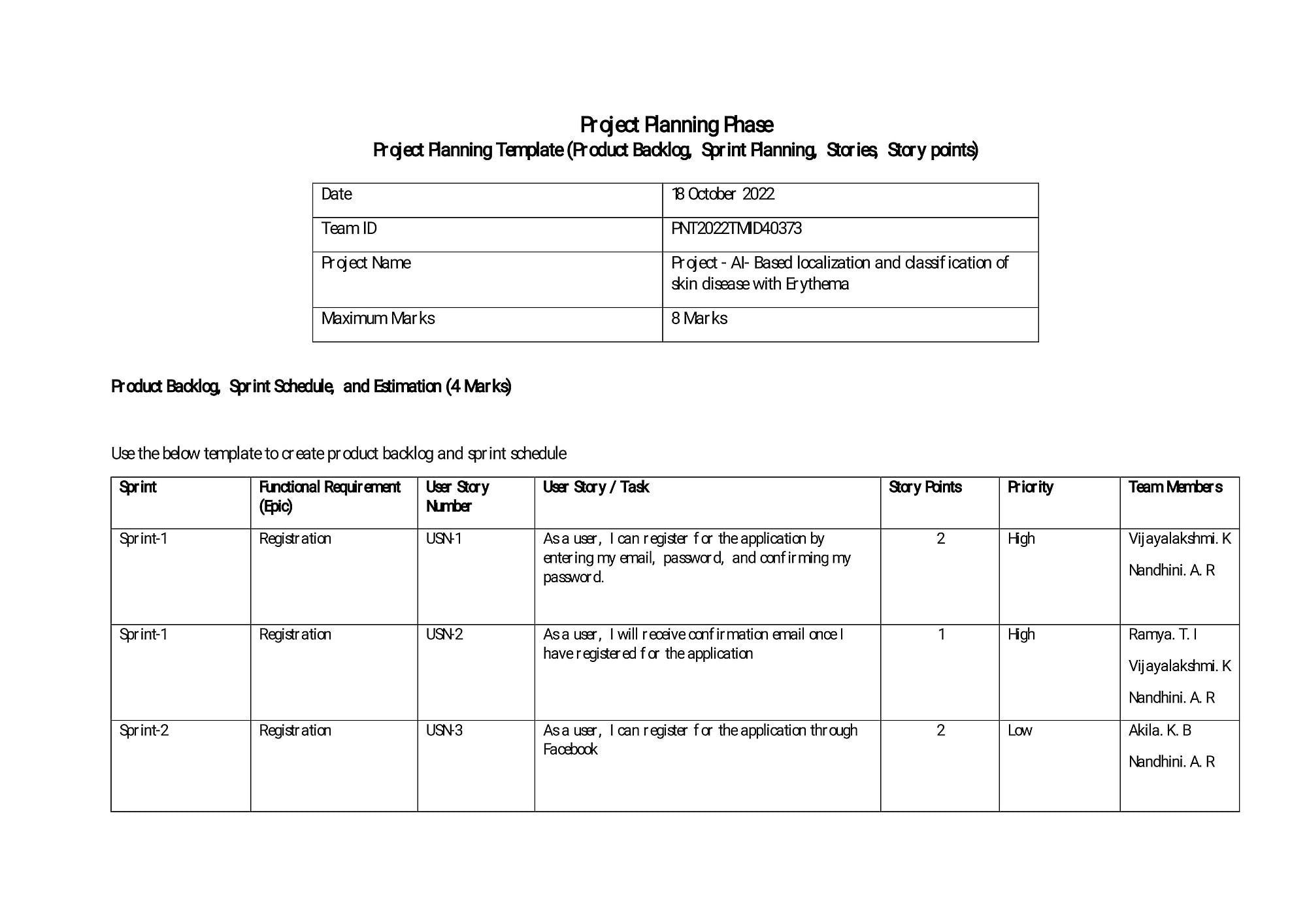


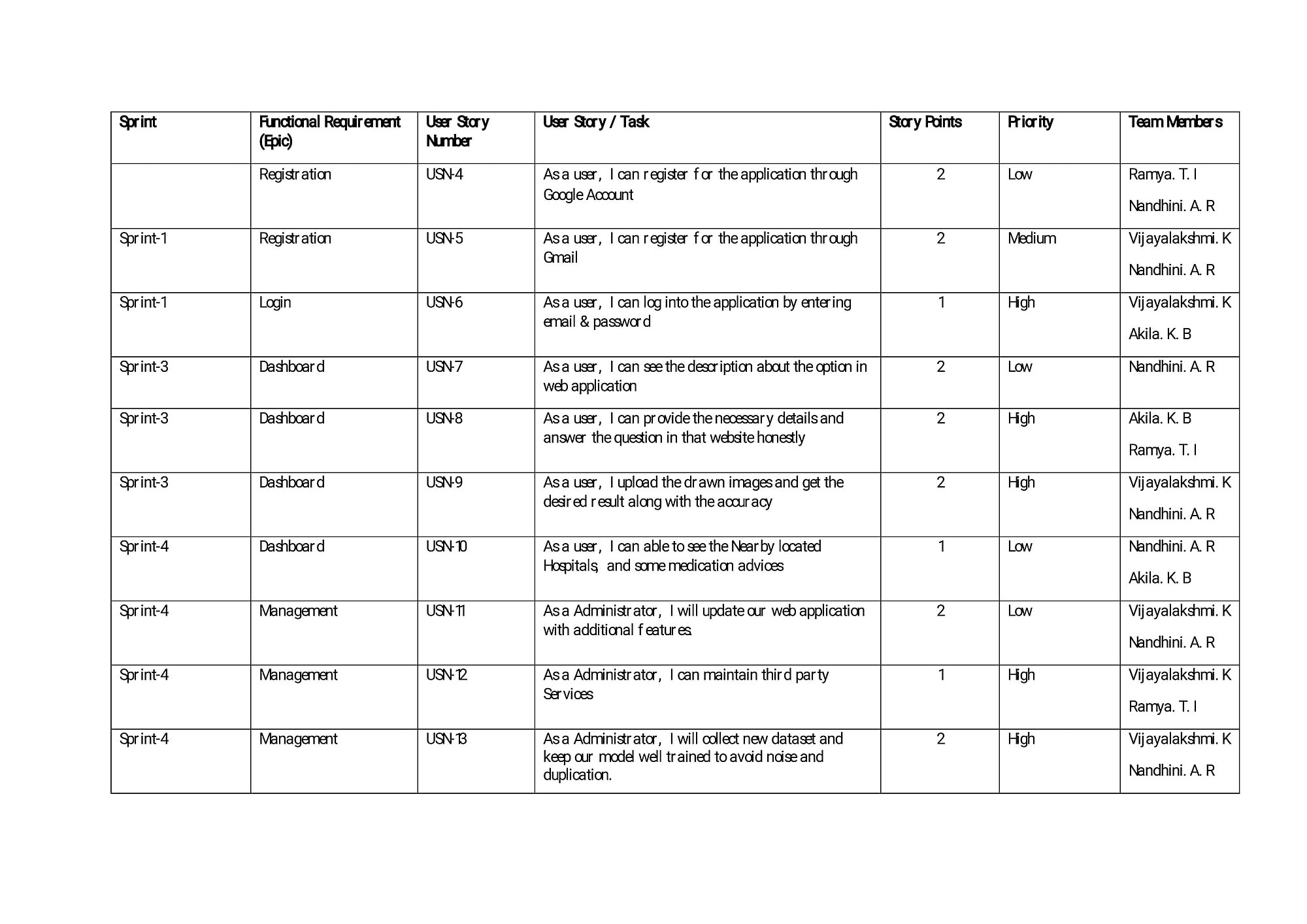
**5.2 Solution & Technical Architecture**

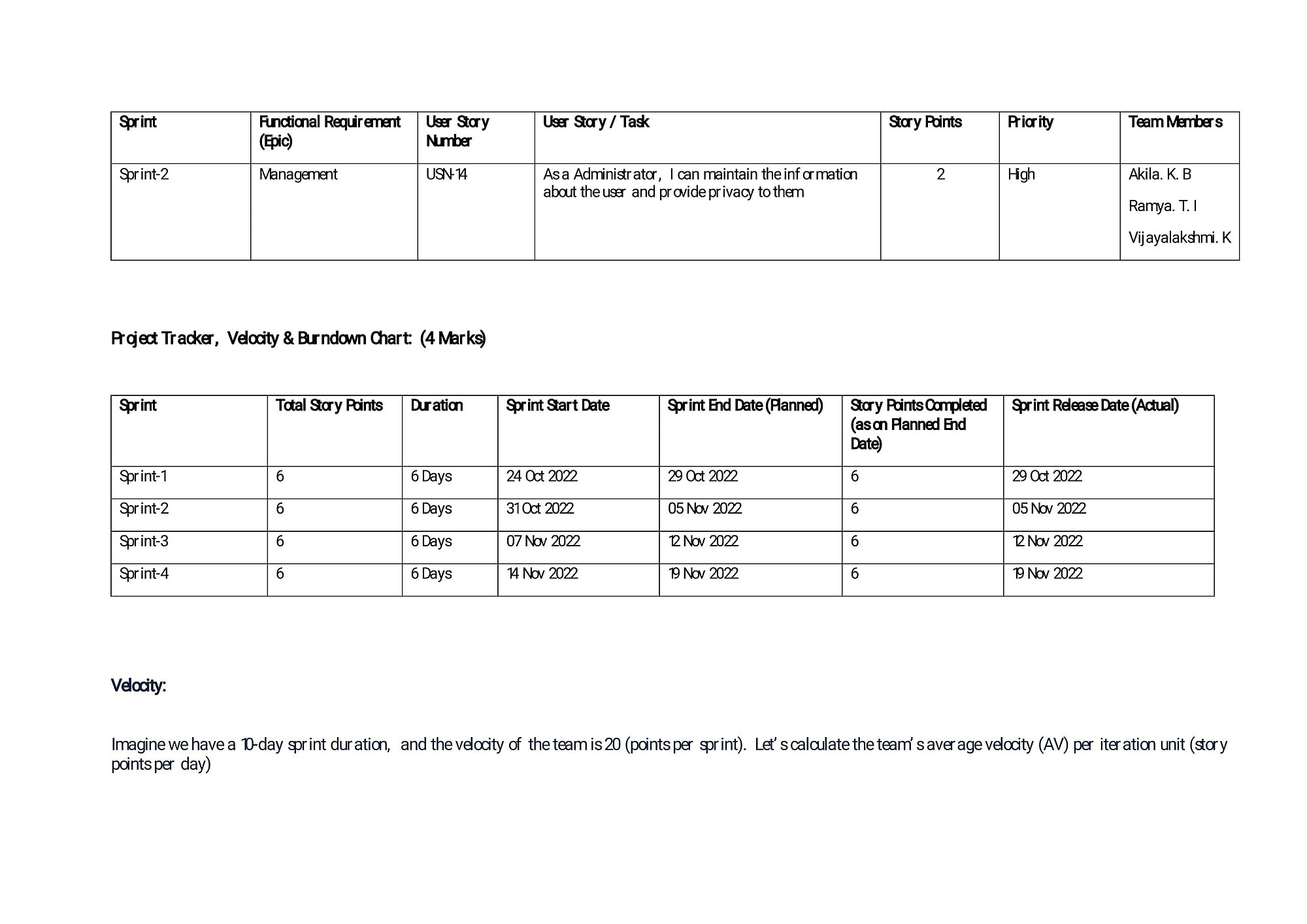


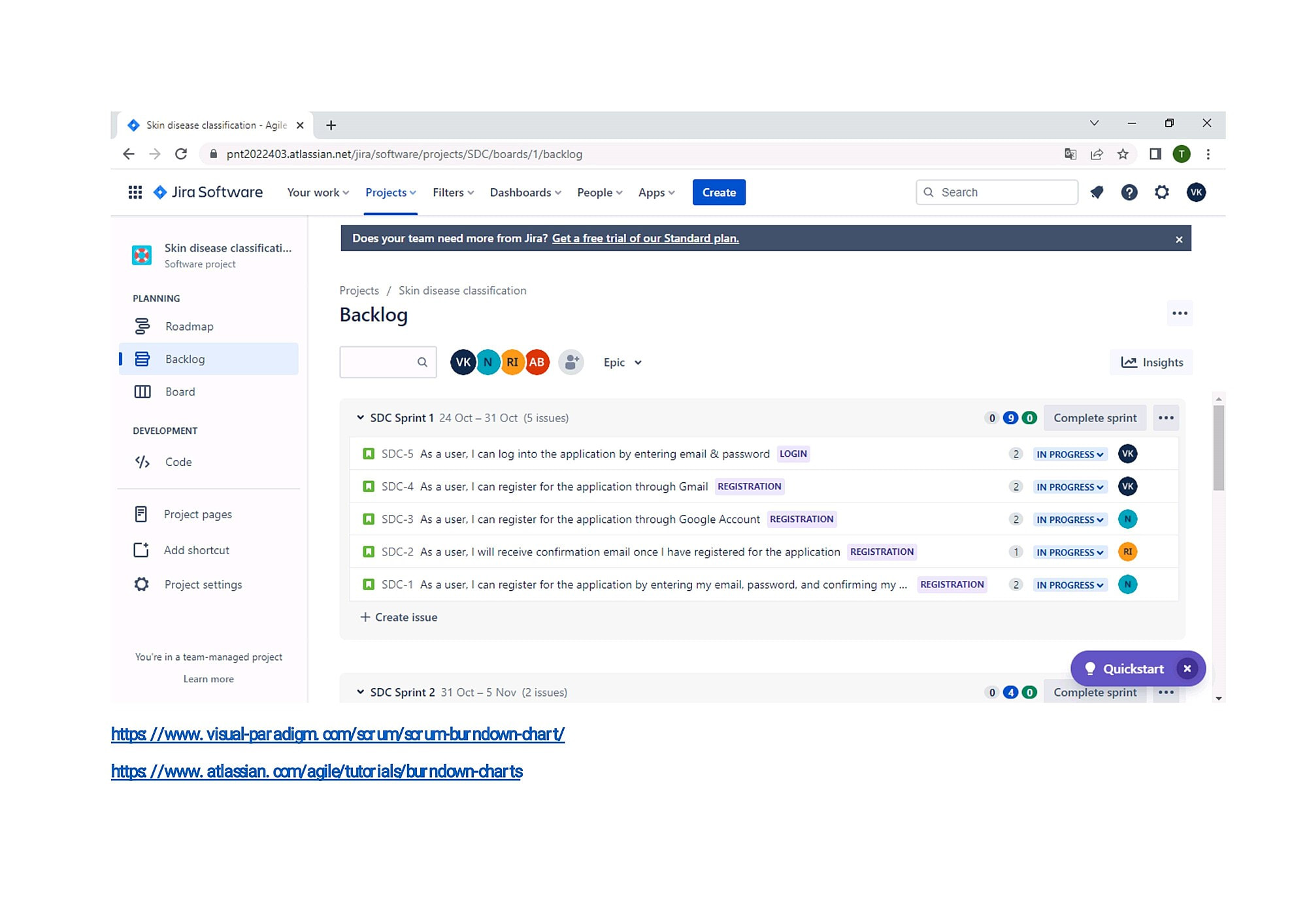
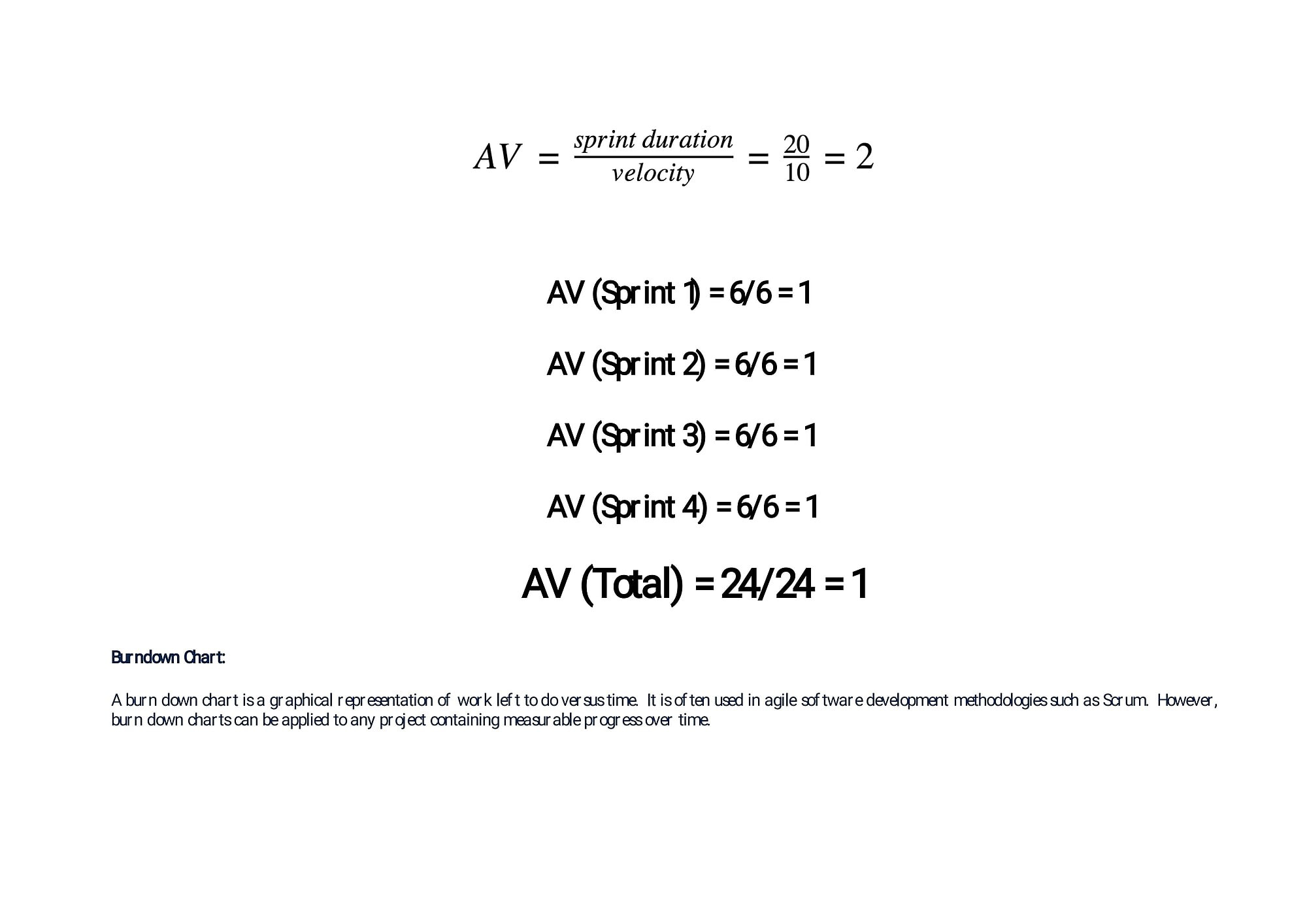
**6.Project planning and scheduling**

**6.1 Sprint planning and estimation**



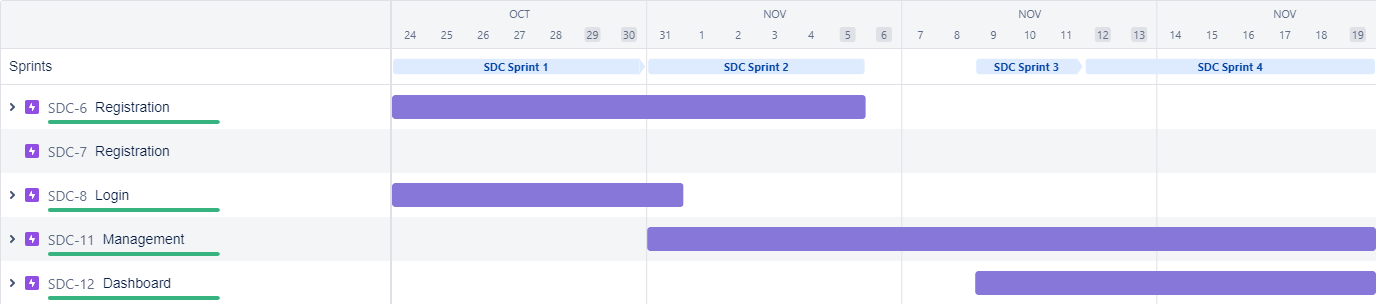




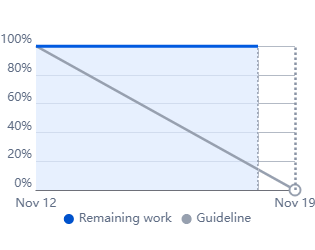




**6.2 Sprint delivery schedule**



**6.3 Reports from JIRA**



**7.Coding and Solutioning**

**import re**

**import numpy as np**

**import os**

**from flask import Flask, app,request,render\_template**

**import sys**

**from flask import Flask, request, render\_template, redirect, url\_for**

**import argparse**

**from tensorflow import keras**

**from PIL import Image**

**from timeit import default\_timer as timer**

**import test**

**from pyngrok import ngrok**

**import pandas as pd**

**import numpy as np**

**import random**

**def get\_parent\_dir(n=1):**

**""" returns the n-th parent dicrectory of the current**

**working directory """**

**current\_path = os.path.dirname(os.path.abspath(\_\_file\_\_))**

**for k in range(n):**

**current\_path = os.path.dirname(current\_path)**

**return current\_path**

**src\_path=r'C:\Users\ELCOT\Desktop\yolo\_structure\2\_Training\src'**

**print(src\_path)**

**utils\_path=r'C:\Users\ELCOT\Desktop\yolo\_structure\Utils'**

**print(utils\_path)**

**sys.path.append(src\_path)**

**sys.path.append(utils\_path)**

**import argparse**

**from keras\_yolo3.yolo import YOLO, detect\_video**

**from PIL import Image**

**from timeit import default\_timer as timer**

**from utils import load\_extractor\_model, load\_features, parse\_input, detect\_object**

**import test**

**import utils**

**import pandas as pd**

**import numpy as np**

**from Get\_File\_Paths import GetFileList**

**import random**

**os.environ["TF\_CPP\_MIN\_LOG\_LEVEL"] = "3"**

**# Set up folder names for default values**

**data\_folder = os.path.join(get\_parent\_dir(n=1), "yolo\_structure", "Data")**

**image\_folder = os.path.join(data\_folder, "Source\_Images")**

**image\_test\_folder = os.path.join(image\_folder, "Test\_Images")**

**detection\_results\_folder = os.path.join(image\_folder, "Test\_Image\_Detection\_Results")**

**detection\_results\_file = os.path.join(detection\_results\_folder, "Detection\_Results.csv")**

**model\_folder = os.path.join(data\_folder, "Model\_Weights")**

**model\_weights = os.path.join(model\_folder, "trained\_weights\_final.h5")**

**model\_classes = os.path.join(model\_folder, "data\_classes.txt")**

**anchors\_path = os.path.join(src\_path, "keras\_yolo3", "model\_data", "yolo\_anchors.txt")**

**FLAGS = None**

**from cloudant.client import Cloudant**

**# Authenticate using an IAM API key**

**client = Cloudant.iam('b98d469c-855b-437b-a3e5-3343900c27ca-bluemix','9bGgTxaHZ06Dst2Mq3XEzPCJMs5KtMUR9gfYF-d5x1aP', connect=True)**

**# Create a database using an initialized client**

**my\_database = client.create\_database('my\_database')**

**app=Flask(\_\_name\_\_)**

**port\_no=5000**

**ngrok.set\_auth\_token("2H7aM94zEuTa40t3J6jKpIqWAc3\_B2UxzZs6qxetntgadxQW")**

**public\_url = ngrok.connect(port\_no).public\_url**

**print(f"To acces the Gloable link please click {public\_url}")**

**#default home page or route**

**@app.route('/')**

**def index():**

**return render\_template('index.html')**

**@app.route('/index.html')**

**def home():**

**return render\_template("index.html")**

**#registration page**

**@app.route('/register')**

**def register():**

**return render\_template('register.html')**

**@app.route('/afterreg', methods=['POST'])**

**def afterreg():**

**x = [x for x in request.form.values()]**

**print(x)**

**data = {**

**'\_id': x[1], # Setting \_id is optional**

**'name': x[0],**

**'psw':x[2]**

**}**

**print(data)**

**query = {'\_id': {'$eq': data['\_id']}}**

**docs = my\_database.get\_query\_result(query)**

**print(docs)**

**print(len(docs.all()))**

**if(len(docs.all())==0):**

**url = my\_database.create\_document(data)**

**#response = requests.get(url)**

**return render\_template('register.html', pred="Registration Successful, please login using your details")**

**else:**

**return render\_template('register.html', pred="You are already a member, please login using your details")**

**#login page**

**@app.route('/login')**

**def login():**

**return render\_template('login.html')**

**@app.route('/afterlogin',methods=['POST'])**

**def afterlogin():**

**user = request.form['\_id']**

**passw = request.form['psw']**

**print(user,passw)**

**query = {'\_id': {'$eq': user}}**

**docs = my\_database.get\_query\_result(query)**

**print(docs)**

**print(len(docs.all()))**

**if(len(docs.all())==0):**

**return render\_template('login.html', pred="The username is not found.")**

**else:**

**if((user==docs[0][0]['\_id'] and passw==docs[0][0]['psw'])):**

**return redirect(url\_for('prediction'))**

**else:**

**print('Invalid User')**

**@app.route('/logout')**

**def logout():**

**return render\_template('logout.html')**

**@app.route('/prediction')**

**def prediction():**

**return render\_template('prediction.html',path="../static/img/6623.jpg",)**

**@app.route('/result',methods=["GET","POST"])**

**def res():**

**# Delete all default flags**

**parser = argparse.ArgumentParser(argument\_default=argparse.SUPPRESS)**

**"""**

**Command line options**

**"""**

**f = request.files['file']**

**f.save("./drive/MyDrive/IBM\_PROJECT/Flask/static/img/"+f.filename)**

**parser.add\_argument(**

**"--input\_path",**

**type=str,**

**default=image\_test\_folder,**

**help="Path to image/video directory. All subdirectories will be included. Default is "**

**+ image\_test\_folder,**

**)**

**parser.add\_argument(**

**"--output",**

**type=str,**

**default=detection\_results\_folder,**

**help="Output path for detection results. Default is "**

**+ detection\_results\_folder,**

**)**

**parser.add\_argument(**

**"--no\_save\_img",**

**default=False,**

**action="store\_true",**

**help="Only save bounding box coordinates but do not save output images with annotated boxes. Default is False.",**

**)**

**parser.add\_argument(**

**"--file\_types",**

**"--names-list",**

**nargs="\*",**

**default=[],**

**help="Specify list of file types to include. Default is --file\_types .jpg .jpeg .png .mp4",**

**)**

**parser.add\_argument(**

**"--yolo\_model",**

**type=str,**

**dest="model\_path",**

**default=model\_weights,**

**help="Path to pre-trained weight files. Default is " + model\_weights,**

**)**

**parser.add\_argument(**

**"--anchors",**

**type=str,**

**dest="anchors\_path",**

**default=anchors\_path,**

**help="Path to YOLO anchors. Default is " + anchors\_path,**

**)**

**parser.add\_argument(**

**"--classes",**

**type=str,**

**dest="classes\_path",**

**default=model\_classes,**

**help="Path to YOLO class specifications. Default is " + model\_classes,**

**)**

**parser.add\_argument(**

**"--gpu\_num", type=int, default=1, help="Number of GPU to use. Default is 1"**

**)**

**parser.add\_argument(**

**"--confidence",**

**type=float,**

**dest="score",**

**default=0.25,**

**help="Threshold for YOLO object confidence score to show predictions. Default is 0.25.",**

**)**

**parser.add\_argument(**

**"--box\_file",**

**type=str,**

**dest="box",**

**default=detection\_results\_file,**

**help="File to save bounding box results to. Default is "**

**+ detection\_results\_file,**

**)**

**parser.add\_argument(**

**"--postfix",**

**type=str,**

**dest="postfix",**

**default="\_disease",**

**help='Specify the postfix for images with bounding boxes. Default is "\_disease"',**

**)**

**yolo = YOLO(**

**\*\*{**

**"model\_path": FLAGS.model\_path,**

**"anchors\_path": FLAGS.anchors\_path,**

**"classes\_path": FLAGS.classes\_path,**

**"score": FLAGS.score,**

**"gpu\_num": FLAGS.gpu\_num,**

**"model\_image\_size": (416, 416),**

**}**

**)**

**img\_path="/drive/MyDrive/IBM\_PROJECT/Flask/static/img/"+f.filename**

**prediction, image,lat,lon= detect\_object(**

**yolo,**

**img\_path,**

**save\_img=save\_img,**

**save\_img\_path=FLAGS.output,**

**postfix=FLAGS.postfix,**

**)**

**yolo.close\_session()**

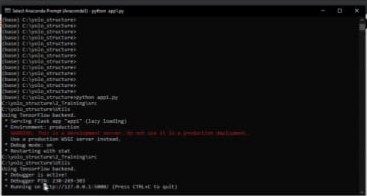
**return render\_template('prediction.html',prediction=str(prediction),path="../static/img/"+f.filename)**

**""" Running our application """**

**if \_\_name\_\_ == "\_\_main\_\_":**

**app.run(port=port\_no)**

**8.Testing**



**9.Result**

**In this research the method of detection wad designed by using pretrained Convolution neural network and SVM.**

**10.Advantages**

• It Provides a free basic dermoscopic tutorial for the general public.

• This system can be used by dermatologists to give a better diagnosis and treatment to the patients.

• The system can be used to diagnose skin diseases at a lower cost.

• This system can be used to detect and classify diseases as well as their severity.

**Disadvantages**

• Internet connectivity is the major issue.

• In case of low Quality images it may show inaccurate information.

• Everyone is not aware of this application

**11.Conclusion**

The system proposed is a Skin Disease Detection System. This system uses test images to detect if it is healthy or not

This system can be used by dermatologists to give a better diagnosis and treatment to the patients.

The system can be used to diagnose skin diseases at a lower cost.

In future, this system can be improved to detect and classify more diseases as well as their severity

**12.Future Scope**

Using imaging methods, it could be possible for deep learning to assist or even replace dermatologists in the diagnosis of skin disease in the near future.Try to add more images with clarity, same size-eg:(16\*16pixels) .

**13.Appendix**

Source code : https://github.com/IBM-EPBL/IBM-Project-48369-1660807101/tree/main/APPLICATION%20BUILDING/Build%20Python%20Code

Github link : https://github.com/IBM-EPBL/IBM-Project-48369-1660807101

Project Demo Link : https://drive.google.com/file/d/1cfWk7LEETsffwwQcetQr3RzbGoVhS022/view?usp=drivesdk

You tube Link : https://youtu.be/xZqy3gQZHgw

https://youtu.be/DbyX9kSoB0c