



Kammavari Sangham (R) 1952, K.S.Group of Institutions

# K. S INSTITUTE OF TECHNOLOGY

📍 No.14, Raghuvanahalli, Kanakapura Road, Bengaluru - 560109

Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi, **Accredited by NBA , NAAC & IEI**

## **Department of Computer Science and Engineering Major Project Phase - II (BCS786)**

**“PREDICTION OF ENDOMETRIAL CANCER AND ITS GRADE USING  
IMAGE PREPROCESSING AND MACHINE LEARNING”**

**Batch No.:32**

1KS22CS093 Neha V  
1KS22CS105 Prakruthi G P  
1KS22CS107 Preethal Dsouza  
1KS22CS123 S HYMA

**Guided By:  
Dr.Vijayalaxmi Mekali  
Professor**

# Contents

- Introduction
- Literature Survey
- Problem statement and objectives
- System Requirement Specification (SRS)
- Software Design Specification (SDS)
- Technology/Tools used
- Implementation of module with codes
- Results
- Future Enhancements
- References

# Introduction

- Endometrial cancer is the most common cancer of the female reproductive system.
- It typically occurs in the lining of the uterus (endometrium), especially in postmenopausal women.
- Recent advancements in especially Image Processing and Machine Learning (ML) offer new possibilities for early detection and classification.
- Early diagnosis is crucial for effective treatment and improved survival.
- Convolutional Neural Networks (CNN) are powerful deep learning models that automatically learn and extract spatial patterns from medical images.
- Transfer learning enhances performance even with limited medical imaging data.
- Provides a comprehensive review of current methodologies in Image Preprocessing and Machine Learning for endometrial cancer prediction and grading.

# LITERATURE SURVEY

s.no	Title of the paper	Author	Journal & Publication year of paper	Methodology	Outcome
1	<b>Automatic Segmentation of Endometrial Cancer on Ultrasound Images</b>	Lidiya Lilly Thampi	international journal of intelligent systems and applications in engineering, 2023	SRAD filtering, Otsu thresholding, PDE-based edge detection for ultrasound images	Improved segmentation in noisy ultrasound data; applicable for non-invasive diagnostics, though not based on histopathological images
2	<b>Cancer Detection Using Image Processing and Machine Learning</b>	Dr. Anita Dixit	Scientific reports,2023	ML methods like KNN, LDA, SVM on medical images (MRI, thermographs); feature extraction on color, shape, structure	Effective feature engineering improves classification; not endometrial-specific but adaptable



# LITERATURE SURVEY

s.no	Title of the paper	Author	Journal & Publication year of paper	Methodology	Outcome
3	<b>Image Analysis Detection of Nuclei Cells in Histopathological Images</b>	Dr. Shoba R. Patil	Computers in Biology and Medicine, 2021	Morphological ops, watershed transform, k-means, color deconvolution on H&E-stained slides	Semi-automated pipeline for better segmentation; still needs manual steps
4	<b>Transfer Learning-Based Endometrial Cancer Detection</b>	Sarah Lee & David Wilson	Science Direct, 2020	Transfer learning with pre-trained models (ResNet, VGG16); fine-tuning on small datasets	Improved accuracy and faster convergence; dependent on domain similarity



# LITERATURE SURVEY

s.n o	Title of the paper	Author	Journal & Publication year of paper	Methodology	Outcome
5	<b>Histopathological Image Analysis for Endometrial Cancer Detection</b>	Emily Davis & Robert Brown	Journal of Medical Imaging, 2019	Classical image processing + machine learning; focus on segmentation and feature extraction	Highlights challenges (e.g., image quality, staining); no experiments; synthesis of existing methods emphasizing standardized preprocessing
6	<b>Challenges and Future Directions in Histopathological</b>	Richard Anderson & Jessica Taylor	Medical Image Analysis, 2018	Review of current limitations; future directions include explainable AI, federated learning, and collaboration	Strategic insights; lacks implementation or validation



# Problem statement and objectives

## Problem Statement:

The current diagnosis of endometrial cancer is slow, prone to errors, and highly dependent on human interpretation. This project proposes an automated system using image preprocessing and machine learning to accurately detect and grade endometrial cancer from biopsy images, aiming to improve diagnostic reliability and patient outcomes.

## Objectives:

1. Biopsy images of endometrial tissue are taken as input for analysis .These images undergo pre-processing to enhance their quality, remove noise, and standardize features for better analysis.
2. The extracted features are then subjected to feature selection, where only the most relevant features for classification are retained.
3. The selected features are fed into a CNN-based classification model, which learns to differentiate between various types and grades of endometrial conditions.
4. The type of condition, such as Endometrial Adenocarcinoma, Endometrial Hyperplasia, Endometrial Polyp, or Normal Endometrium . The grade of cancer, categorized as High, Intermediate, or Low.



# System Requirement Specification (SRS)

## 1. Hardware Requirements

- Intel i5
- 16 GB RAM
- 512 GB Storage

## 2. Software Requirements

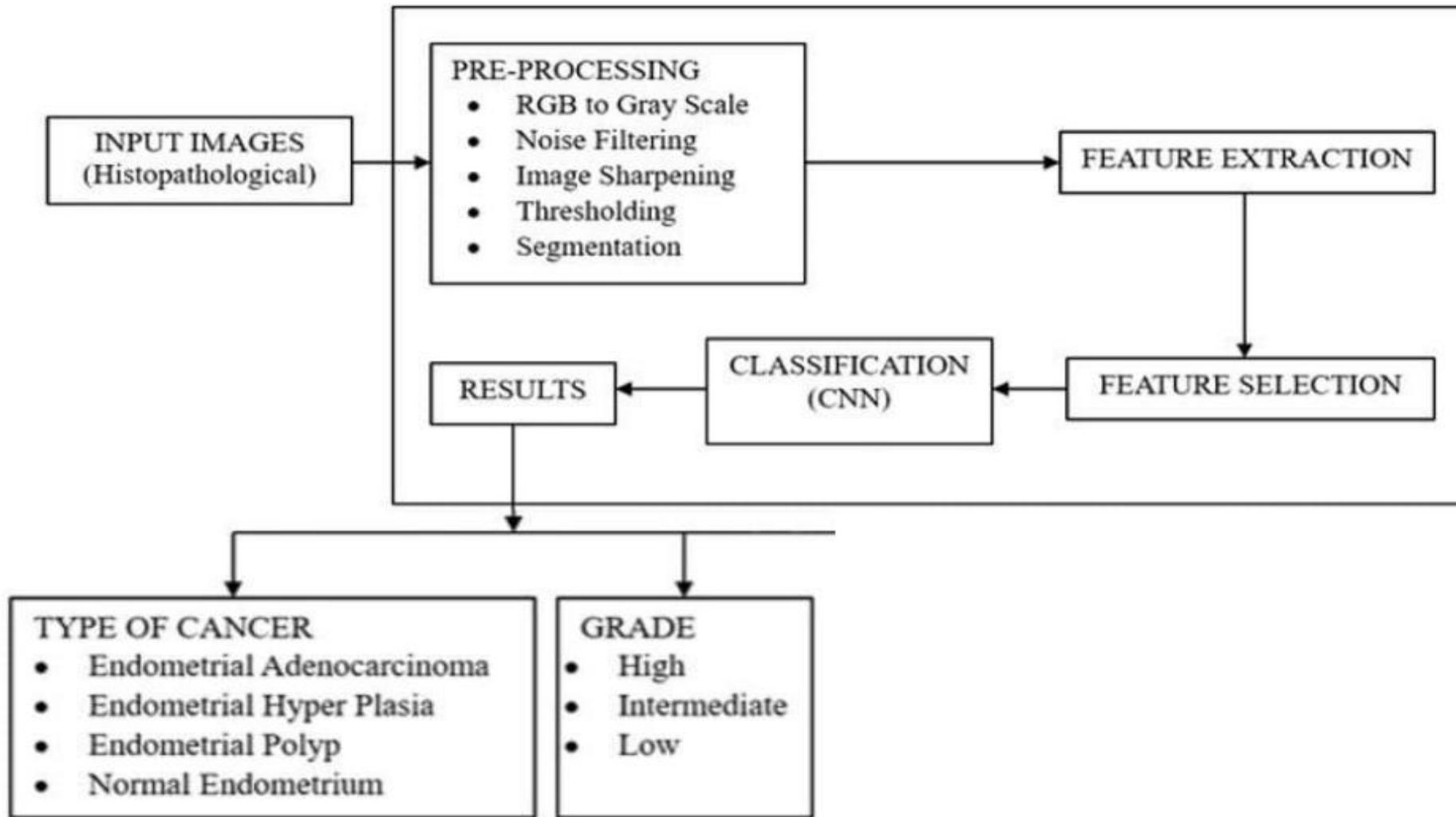
- OS: Windows 11
- Language: Python
- Tools: VS Code, OpenCV, Scikit-learn, Flask, Jupyter Notebook

## 3. Functional Requirements

- Image Collection & preprocessing
- ROI(Region of Interest Segmentation) & feature extraction
- CNN-based classification (cancer detection & grading)
- Real-time prediction

# Software Design Specification (SDS)

## 1. System Architecture:





# Software Design Specification (SDS)

## 2. Major Modules:

- Preprocessing – Grayscale, noise removal, thresholding, sharpening
- Segmentation – ROI extraction of uterus region
- Feature Extraction – GLCM, PCA for statistical features
- Classification – CNN model for cancer/grade detection

## 3. Data Flow Input:

Histopathology images

- Processing: Image → Features → CNN
- Output: Cancer prediction with grade



# Technology/Tools used

## 1. Programming Language:

- Python

## 2. Libraries & Frameworks:

- **OpenCV** – Image preprocessing (grayscale, thresholding, sharpening, noise removal)
- **TensorFlow / Keras** – CNN model training and classification
- **Scikit-learn** – Machine learning utilities (metrics, preprocessing)
- **NumPy, Pandas, Matplotlib** – Data handling and visualization

## 3. Development Tools:

- Visual Studio Code (IDE)
- Jupyter Notebook (model training , testing & analysis)

## 4. Web Tools:

- Flask (Web interface for predictions)



# Implementation of module with codes

```
import os
import shutil
import numpy as np
import cv2
import sqlite3

from pathlib import Path
from flask import Flask, render_template, request, url_for

import matplotlib
matplotlib.use('Agg') # Use non-GUI backend for matplotlib to avoid threading issues
import matplotlib.pyplot as plt

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import load_img, img_to_array

# --- PATH SETUP ---
BASE_DIR = Path(__file__).resolve().parent
TEST_DIR = BASE_DIR / "test"
STATIC_DIR = BASE_DIR / "static"
UPLOAD_DIR = STATIC_DIR / "images"
MODEL_PATH = BASE_DIR / "Convolutional_Neural_Network.h5"

UPLOAD_DIR.mkdir(parents=True, exist_ok=True) # make sure static/images exists

# Flask app
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/userlog', methods=['GET', 'POST'])
def userlog():
    if request.method == 'POST':
        connection = sqlite3.connect('user_data.db')
        cursor = connection.cursor()

        name = request.form['name']
        password = request.form['password']

        query = "SELECT name, password FROM user WHERE name = ? AND password = ?"

        cursor.execute(query, (name, password))
        result = cursor.fetchall()

        if len(result) == 0:
            return render_template('index.html', msg='Sorry, Incorrect Credentials Provided, Try Again')
        else:
            return render_template('userlog.html')

    return render_template('index.html')

@app.route('/userreg', methods=['GET', 'POST'])
def userreg():
    if request.method == 'POST':
        connection = sqlite3.connect('user_data.db')
        cursor = connection.cursor()

        name = request.form['name']
        password = request.form['password']
        mobile = request.form['phone']
        email = request.form['email']

        command = """CREATE TABLE IF NOT EXISTS user(name TEXT, password TEXT, mobile TEXT, email TEXT)"""
        cursor.execute(command)
        cursor.execute("INSERT INTO user VALUES (?, ?, ?, ?)", (name, password, mobile, email))
        connection.commit()

        return render_template('index.html', msg='Successfully Registered')

    return render_template('index.html')

@app.route('/developer.html')
def developer():
    return render_template('developer.html')

@app.route('/image', methods=['GET', 'POST'])
def image():
    if request.method == 'POST':
```

```
        cursor.execute(query, (name, password))
        result = cursor.fetchall()

        if len(result) == 0:
            return render_template('index.html', msg='Sorry, Incorrect Credentials Provided, Try Again')
        else:
            return render_template('userlog.html')

    return render_template('index.html')

@app.route('/userlog.html')
def userlogg():
    return render_template('userlog.html')

@app.route('/developer.html')
def developer():
    return render_template('developer.html')

@app.route('/image', methods=['GET', 'POST'])
def image():
    if request.method == 'POST':
```



# Implementation of module with codes

```
# clear old images
for file in UPLOAD_DIR.glob("*"):
    file.unlink()

fileName = request.form['filename']
src_path = TEST_DIR / fileName
dst_path = UPLOAD_DIR / fileName

if not src_path.exists():
    return render_template('userlog.html', msg=f"File {fileName} not found in test folder.")

shutil.copy(src_path, dst_path)

image = cv2.imread(str(src_path))

# Grayscale conversion
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
luminosity_gray = 0.21 * image[:, :, 2] + 0.72 * image[:, :, 1] + 0.07 * image[:, :, 0]
cv2.imwrite(str(STATIC_DIR / 'luminosity_gray.jpg'), luminosity_gray)

# Median filter
median_filtered = cv2.medianBlur(luminosity_gray.astype("uint8"), 5)
cv2.imwrite(str(STATIC_DIR / 'median_filtered.jpg'), median_filtered)

# Unsharp masking
gaussian = cv2.GaussianBlur(median_filtered, (9, 9), 10.0)
unsharp_image = cv2.addWeighted(gray_image, 1.5, gaussian, -0.5, 0)
cv2.imwrite(str(STATIC_DIR / 'unsharp_masked.jpg'), unsharp_image)

# Thresholding
_, gaussian_threshold = cv2.threshold(unsharp_image, 127, 255, cv2.THRESH_BINARY)
cv2.imwrite(str(STATIC_DIR / 'gaussian_thresholded.jpg'), gaussian_threshold)

# Edge-based segmentation
edges = cv2.Canny(gaussian_threshold, 100, 200)
cv2.imwrite(str(STATIC_DIR / 'edge_based_segmentation.jpg'), edges)

# White areas represent tumor/lymph nodes/metastasis regions
gray_for_contour = cv2.imread(str(src_path), cv2.IMREAD_GRAYSCALE)
blurred = cv2.GaussianBlur(gray_for_contour, (5, 5), 0)
_, binary_mask = cv2.threshold(blurred, 127, 255, cv2.THRESH_BINARY)
```

```
model = load_model(str(MODEL_PATH))
path = str(dst_path)

def prepare_test_image(path):
    img = load_img(path, target_size=(128, 128), color_mode="grayscale")
    x = img_to_array(img)
    x = x / 255.0
    return np.expand_dims(x, axis=0)

result = model.predict(prepare_test_image(path))
class_result = np.argmax(result, axis=1)

if class_result[0] == 0:
    str_label = "Endometrial Adenocarcinoma"
    stage = "stage1 cancer"
elif class_result[0] == 1:
    str_label = "Endometrial Hyperplasia"
    stage = "stage2 cancer"
elif class_result[0] == 2:
    str_label = "Endometrial Polyp"
    stage = "stage3 cancer"
else:
    str_label = "Normal Endometrium"
    stage = "Normal"

accuracy = f"The predicted image of {str_label} is with an accuracy of {result[0][class_result[0]]*100:.2f}%"

dic = {
    "EA": float(result[0][0]),
    "EH": float(result[0][1]),
    "EP": float(result[0][2]),
    "NE": float(result[0][3]),
}
fig = plt.figure(figsize=(5, 5))
plt.bar(dic.keys(), dic.values(), color="maroon", width=0.3)
plt.xlabel("Comparison")
plt.ylabel("Accuracy Level")
plt.title("Accuracy Comparison between Endometrium Cancer")
plt.savefig(STATIC_DIR / "matrix.png")
plt.close(fig)
```



# Implementation of module with codes

```
ImageDisplay = [
    f"/static/images/{fileName}",
    "/static/luminosity_gray.jpg",
    "/static/median_filtered.jpg",
    "/static/unsharp_masked.jpg",
    "/static/gaussian_thresholded.jpg",
    "/static/edge_based_segmentation.jpg",
    "/static/matrix.png",
]

labels = [
    "Original",
    "Gray Image",
    "Median Filter",
    "Unsharp Masking",
    "Gaussian Threshold",
    "Edge Based (Canny)",
    "Graph",
]

return render_template(
    "results.html",
    labels=labels,
    status=str_label,
    accuracy=accuracy,
    cell_count=cell_count,
    grade=grade,
    stage=stage,
    ImageDisplay=ImageDisplay,
    n=len(ImageDisplay),
)

return render_template("userlog.html")

@app.route("/logout")
def logout():
    return render_template("index.html")
```

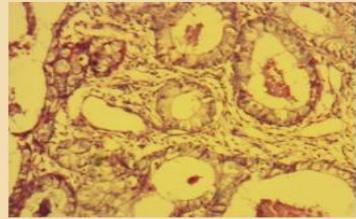
```
@app.route("/logout")
def logout():
    return render_template("index.html")

if __name__ == "__main__":
    app.run(debug=True, use_reloader=False)
```

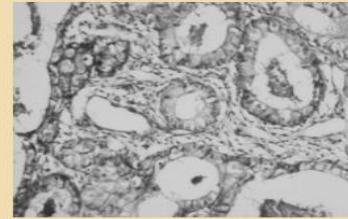
# Results

## ENDOMETRIUM CANCER DETECTION

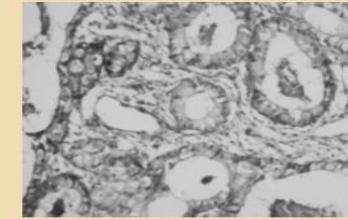
Original



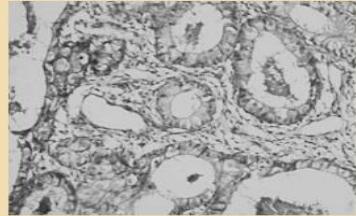
Gray Image



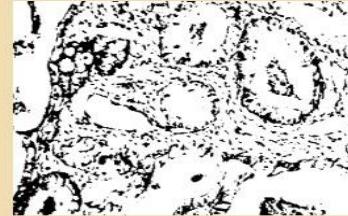
Median Filter



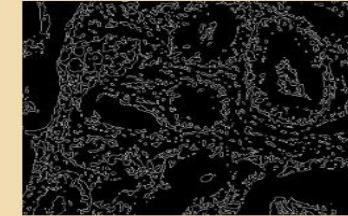
Unsharp Masking



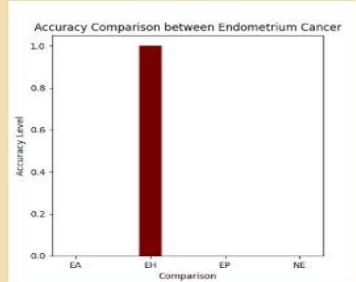
Gaussian Threshold



Edge Based (Canny)



Graph



Result

Status: Endometrial Hyperplasia  
 Accuracy: The predicted image of Endometrial Hyperplasia is with an accuracy of 99.99%  
 Cancer Grade: Low Grade (Well-Organized)  
 Stage: stage2 cancer

## ENDOMETRIUM CANCER DETECTION

Original



Gray Image



Median Filter



Unsharp Masking



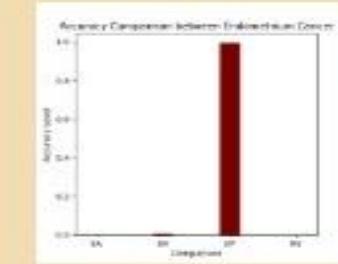
Gaussian Threshold



Edge Based (Canny)



Graph



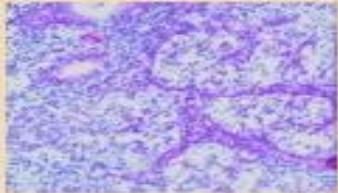
Result

Status: Endometrial Polyp  
 Accuracy: The predicted image of Endometrial Polyp is with an accuracy of 99.99%  
 Cancer Grade: High Grade (Disorganized/Solid Growth)  
 Stage: stage3 cancer

# Results

## ENDOMETRIUM CANCER DETECTION

Original



Gray Image



Median Filter



Unsharp Masking



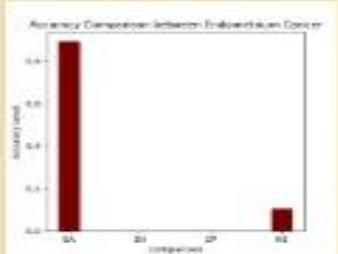
Gaussian Threshold



Edge Based (Canny)



Graph



Result

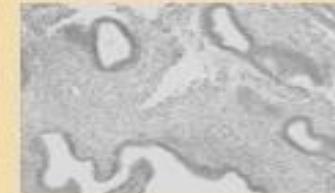
Status: Endometrial Adenocarcinoma  
 Accuracy: The predicted image of Endometrial Adenocarcinoma is with an accuracy of 86.40%  
 Cancer Grade: Low Grade (Well-Organized)  
 Stage: stage1 cancer

## ENDOMETRIUM CANCER DETECTION

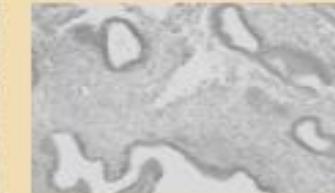
Original



Gray Image



Median Filter



Unsharp Masking



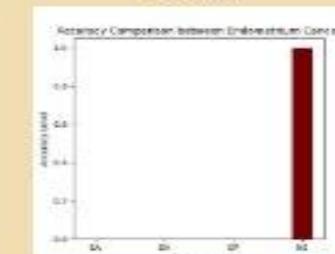
Gaussian Threshold



Edge Based (Canny)



Graph



Result

Status: Normal Endometrium  
 Accuracy: The predicted image of Normal Endometrium is with an accuracy of 86.46%  
 Cancer Grade: Low Grade (Well-Organized)  
 Stage: Normal

**Endometrial Adenocarcinoma**

**Normal Endometrium**



# Future Enhancements

**Mobile/Web App:** Deploy as a portable tool for healthcare professionals.

## References

1. E. Davis and R. Brown, “Histopathological Image Analysis for Endometrial Cancer Detection: A Review,” in International Journal of Biomedical Imaging, vol. 2019, Article ID 1294571, 2019.
2. S. Lee and D. Wilson, “Transfer Learning-Based Endometrial Cancer Detection in Histopathological Images,” in IEEE Access, vol. 9, pp. 139405–139414, 2021.
3. R. Anderson and J. Taylor, “Histopathological Image Analysis for Endometrial Cancer: Challenges and Future Directions,” in Computers in Biology and Medicine, vol. 145, p. 105434, 2022.
4. S. R. Patil, “Detection of Nuclei Cell in Histopathological Images of Uterine Cancer: Adenocarcinoma of Endometrium,” in Procedia Computer Science, vol. 167, pp. 2391–2399, 2020.
5. A. Dixit, “Cancer Detection using Image Processing and Machine Learning,” in International Journal of Engineering and Advanced Technology (IJEAT), vol. 9, no. 5, pp. 2310–2315, June 2020.
6. L. L. Thampi, “An Automatic Segmentation of Endometrial Cancer on Ultrasound Images,” in International Journal of Scientific Research in Computer Science, Engineering and Information Technology, vol. 6, no. 2, pp. 456–462, 2021.