A MACHINE LEARNING APPROACH TO LUNG CANCER DETECTION

A PROJECT REPORT

Submitted by

M.VENKATA SAI GANESH KUMAR[RA2211026010146]
TULASI GANESH VARDHAN [RA2211026010147]
D V V. ADITYA VARDHAN[RA2211026010148]

Under the Guidance of

Dr.S.KRISHNAVENI

Associate Professor
Department of Computational Intelligence

in partial fulfillment of the requirementsfor the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING
with specialization in Artificial Intelligence And Machine Learning

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY KATTANKULATHUR- 603 203

MAY 2025

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that 18CSP107L - Minor Project [18CSP108L- Internship] report titled "**A Machine Learning Approach to Lung Cancer Detection**" is the bonafide work of "**M.VENKATA SAI GANESH KUMAR [RA2211026010146], TULASI GANESH VARDHAN [RA2211026010147], D V V ADITYA VARDHAN [RA2211026010148]"** who carried out the project work[internship] under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Dr.S.Krishnaveni
Course Faculty
Associate Professor
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur

SIGNATURE
Dr. R.Annie Uthra
Head of the Department
Professor
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur

ACKNOWLEDGEMENTS

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to **Dr. Leenus Jesu Martin M,** Dean-CET, SRM Institute of Science and Technology, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We encompass our sincere thanks to, **Dr. M. Pushpalatha**, Professor and Associate Chairperson - CS, School of Computing and **Dr. Lakshmi,** Professor and Associate Chairperson -AI, School of Computing, SRM Institute of Science and Technology, for their invaluable support.

We are incredibly grateful to our Head of the Department, **Dr. R.Annie Uthra,** Professor, Computational Intelligence, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinators, Panel Head, and Panel Members Department of Computational Intelligence, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr.Prithi S**, Assistant Professor Department of Computational Intelligence, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, Department of Computational Intelligence, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his / her mentorship. He / She provided us with the freedom and support to explore the research topics of our interest. His / Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff members of <<Dept. Name>>, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement

M. Venkata Sai Ganesh Kumar,

Tulasi Ganesh Vardhan,

A V V. Aditya Vardhan

ABSTRACT

A The primary factors behind late diagnosis combined with restricted expert radiological analysis maintain lung cancer at the top position of cancer-related global mortality rates. Medical diagnostics demonstrate efficiency but they employ human analysis among other factors that could negatively affect both timely and precise patient care decisions. An automated system using deep learning analysis detects lung cancer from histopathological pictures at an early stage. The EfficientNetB3 constitutes the core of the proposed model architecture because it stands as one of the most accurate and resource-efficient convolutional neural networks (CNNs) available in the market. EfficientNetB3 utilizes compound scaling to distribute the network scaling across all dimensions enabling it to surpass basic CNN models regarding both accuracy and efficiency metrics. The design matches perfectly with medical image analysis because accuracy stands as the primary concern.

The preprocessing phase of the project uses two important techniques: pixel normalization and data augmentation through rotation and zooming or flipping steps to deliver solid model results. These methods lower overfitting risks which builds data variety and enhances the tested model's ability to process untracked observations.An evaluation of the recommended system for histopathological lung tissue picture analysis occurs against labelled pictures by using accuracy and precision with F1-score and recall measures. The assessment methods provide thorough evaluation of how well a model classifies samples for real-world applications. The main advantage of this research lies in its complete description of an artificial intelligence diagnostic aid that maintains superb diagnostic performance and decreases physicians dependency on traditional expert interpretation. Through the application of EfficientNetB3 the model enhances its ability to extract features and classify samples which leads to earlier medical interventions aided by more informative clinical decisions.Because of its modular architecture the system will integrate with other diagnostic tools in the future and expand capability to identify different cancer types.Through this initiative the continued digital transformation of healthcare receives an advance via its scalable and efficient artificial intelligence-based lung cancer detection mechanism. The automated diagnostic capabilities of this system help healthcare workers make rapid accurate choices which leads to improved patient results along with lower death statistics from late-stage lung cancer discovery.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction to Project:

Lung cancer constitutes a major worldwide public health challenge because it continually appears among the leading causes of cancer-related mortality. Lung cancer leads to approximately 1.8 million deaths per year resulting in a death rate of almost 18% of all cancer fatalities based on World Health Organization (WHO) data. The significant number of deaths from lung cancer stems mainly from the insufficient time people receive between disease detection and correct diagnosis. Survival rates with successful treatment outcomes become much higher when physicians detect lung cancer during early stages of development. The absence of early lung malignancy diagnosis poses significant hurdles because medical tests remain difficult to access and human diagnosis methods demand skilled analysts while the cancers often do not display obvious symptoms.

Lung cancer diagnostic processes rely mainly on chest X-rays combined with computed tomography (CT) scans together with biopsy sample analysis for histopathological examination. The definitive diagnosis through histopathological analysis remains the standard procedure but requires significant time from experts in pathology. Subjectivity during manual tissue slide evaluation by microscopes results in intra- and inter-observer variability of the diagnostic process. Diagnostic challenges in healthcare institutions have reached overwhelming proportions due to the rising number of suspected cancer cases particularly in locations with limited medical resources.

Artificial Intelligence (AI) has become essential through deep learning technology for addressing various issues in medical image analysis. Specific AI diagnostic algorithms present an effective way to boost clinical decisions because they execute standard procedures and display potentially malignant areas which experts need to assess. Convolutional Neural Networks (CNNs) have become popular due to their ability to identify spatial hierarchies becoming the top choice in medical imaging for classification and detection together with segmentation because of their advanced pattern learning capabilities.

The research examines the employment of EfficientNetB3 a modern CNN design for automatic lung cancer detection from histopathological images. Among all CNN models EfficientNetB3 stands out because it implements compound scaling thus it reaches superior accuracy levels using fewer parameters while maintaining affordable computational expenses. The method used by Efficient Net stands apart

fromtraditional models because it implements a standardized system to expand networks while delivering superior performance in medical related functions.

EfficientNetB3 demonstrates exceptional capability to derive informative and distinctive tissue features from complicated tissue arrangements without manual feature design involvement. Effective analysis of histopathology requires this methodology since visual characteristics differ extensively between cancer types and patient populations and diagnosis grades. Transfer learning allows the model to utilize ImageNet pretrained weights which improves performance levels when processing medical image datasets even when they are relatively small.

The project establishes a strongly performing model through its implementation of detailed preprocessing operations. Image pixel intensity normalization along with rotation and horizontal and vertical modification and scaling and zooming specifications form the augmentation process. The dataset expands through data augmentation techniques that make the information more diverse and address the problem of overfitting while dealing with small or unvaried medical data.

The research aims to accomplish two tasks: building an EfficientNetB3-based deep learning network that correctly identifies malignant versus non-malignant lung tissue images followed by standard performance assessment using accuracy, precision, recall and F1-score evaluation metrics. The evaluation metrics supply an extensive picture of how well the model performs as a diagnostic tool for potential clinical implementation.

The research pursues to enhance healthcare digital transformation through an AI-based technique which detects lung cancer at an early stage. The integration of EfficientNetB3's performance attributes with preprocessing strategies along with interpretability mechanisms creates a medical tool that aids pathologists in their diagnostic work. The proposed framework should expand to diagnose multiple cancer types while connecting to cloud infrastructure because this approach will benefit patients in developing regions who lack adequate medical resources.

1.2 Motivation

Lung cancer proves to be one of the most dangerous forms of cancer worldwide yet patients receive their diagnosis too late because current detection methods are not available or accessible in time. The diagnosis needs to happen quickly and precisely to enhance survival rates however traditional histopathological examinations demand expert staff and need extended times along with insufficient equipment availability in rural healthcare networks.

Machine learning and medical imaging technologies continue to advance so artificial intelligence has gained increased focus for automatizing and improving diagnostic processes. The application of

convolutional neural networks (CNNs) within deep learning systems has proven effective for pattern identification and medical image diagnosis which enables excellent potential for cancer detection from these images.

The main purpose of this project is to establish a system that combines professional-led lung cancer diagnosis with an AI-based solution that scales across medical facilities. The system implements EfficientNetB3 as its CNN architecture because it provides both high diagnostic accuracy and low computational requirements. The system possesses characteristics that enable integration with mobile diagnostic tools and telemedicine platforms which allows healthcare providers to enhance their speed and accuracy of decisions even in remote areas.

1.3 Sustainable Development Goal of the Project

The proposed research project named "A Machine Learning Approach to Lung Cancer Detection" perfectly aligns with Sustainable Development Goal 3 that focuses on Good Health and Well-Being. The objective under SDG 3 focuses on two essential responsibilities including the reduction of non-communicable disease mortality and improved health services accessibility and innovative healthcare infrastructure development. Various aspects demonstrate the alignment:

- Early Detection of Life-Threatening Disease

  This research project aims to detect lung cancer as it ranks among the most dangerous cancer types worldwide. Early detection enabled by deep learning in the system directly contributes to decreasing premature deaths while improving patient survival chances.

- Enhancing Access to Diagnostic Tools

  The current diagnosis methods for cancer work extensively with expert pathologists while requiring many clinical resources. The AI-diagnostic system breaks dependency on human diagnosis which allows healthcare professionals to reach patients equally in both rural and impoverished locations while maintaining accurate diagnosis standards.

- Supporting Healthcare Digitalization

  The combination between EfficientNetB3 with explainable AI techniques (such as Grad-CAM and SHAP) enables healthcare transformation through intelligent system deployment. New diagnostic reliability standards become possible through this system while health professionals gain actionable knowledge from the system.

- Fostering Clinical Trust and Transparency

The model achieves explainability through which medical experts can verify and fully understand the AI-recommended decisions. The implementation of explainable techniques creates technology trust that enables both ethical and safe real-world application of clinical solutions.

- Promoting Telemedicine and Scalable Solutions

  Due to its lightweight design structure this model can operate effectively on cloud and edge systems which extends its capabilities for telemedical applications. The system enables medical service provision to populations that lack current healthcare access which advances universal health coverage initiatives.

- Reducing Healthcare Burden

  Healthcare systems experience fewer burdens which results from early detection accuracy because it reduces successive medical procedures and leads to better treatment strategies. Healthcare affordability reduces along with patient and caregiver mental and financial pressure.

- Contributing to Global Health Goals

  The project demonstrates how artificial intelligence achieves SDG 3 via its broad implementation and practical benefits that enhance medical diagnosis and enable on-time medical responses and strengthen community health.

1.4 Product Vision Statement

1.4.1. Audience:

- Primary Audience: Medical professionals (doctors, pathologists) who require faster and more accurate support in diagnosing lung cancer using tissue samples.

- Secondary Audience: Hospitals, research institutions, healthcare startups, and organizations looking to implement AI-based tools to improve cancer detection efficiency and patient care.

1.4.2. Needs:

- Primary Needs:
- A reliable, accurate, and quick diagnostic assistant that can detect lung cancer from histopathological tissue images.
- Support in early detection to improve survival rates through timely treatment.
- Reduction in human error and diagnostic fatigue for medical professionals.

- Secondary Needs:
- An easy-to-use platform with intuitive reporting and visualization of detection results
- A system that adapts and improves over time based on feedback and additional data.
- Compatibility with existing hospital IT systems and workflows for seamless integration.

1.4.3. Products:

- Core Product: An AI-powered lung cancer detection system that analyzes lung tissue images and provides classification results (e.g., cancerous vs. non-cancerous) to aid doctors in diagnosis.

- Additional Features:

- Visual heatmaps or marked regions on tissue images highlighting suspicious areas.
- Detailed reports including prediction confidence scores and suggested next steps.
- Feedback system for doctors to validate or correct predictions, helping the AI model to continuously improve.

1.4.4. Values:

- Core Values:

- **Accuracy:** Delivering highly reliable detection results to support critical medical decisions.
- **Efficiency:** Reducing diagnosis time, enabling quicker treatment initiation.
- **Support:** Assisting healthcare providers without replacing expert human judgment.
- **Continuous improvement:** Improving detection capability by learning from new data and user feedback

Differentiators:

- Tailored specifically for lung tissue analysis rather than generic image classification.
- Provides interpretable AI outputs (e.g., highlighted areas) rather than just binary results.
- Focuses equally on user-friendly integration into existing medical workflows along with technical excellence.

1.5 Product Goal

To develop an AI-powered diagnostic tool that accurately detects lung cancer from lung tissue images, supporting healthcare professionals in making faster, more reliable diagnoses. The system aims to enhance early cancer detection, reduce diagnostic errors, integrate seamlessly into existing clinical workflows, and continuously improve through user feedback, ultimately contributing to better patient outcomes and saving lives.

1.6 Product Backlog

S.NO	USER Story
1	**As a radiologist**, I want to upload a lung tissue image and receive an accurate cancer prediction so that I can quickly diagnose patients.
2	**As a medical researcher**, I want to access prediction results along with model confidence scores so that I can analyze the reliability of the model.
3	**As a hospital administrator**, I want user role-based access control so that sensitive patient data is securely accessed only by authorized personnel.
4	**As a pathologist**, I want to view a detailed classification report after analysis so that I can understand the model's reasoning.
5	**As a clinician**, I want the model to provide real-time predictions so that I can make faster clinical decisions during consultations.
6	**As a data scientist**, I want the ability to retrain the model on updated datasets so that the system remains current with the latest diagnostic data.
7	**As a researcher**, I want to download and export model results into a report format so that I can include them in clinical studies or papers.
8	**As a system administrator**, I want to monitor model performance metrics like accuracy and precision so that I can detect any model drift over time.
9	**As a medical student**, I want to use anonymized lung tissue images for educational purposes so that I can learn to identify cancerous tissues.
10	**As an oncologist**, I want to compare model predictions with manual diagnoses so that I can validate the AI system's effectiveness in real-world scenarios.

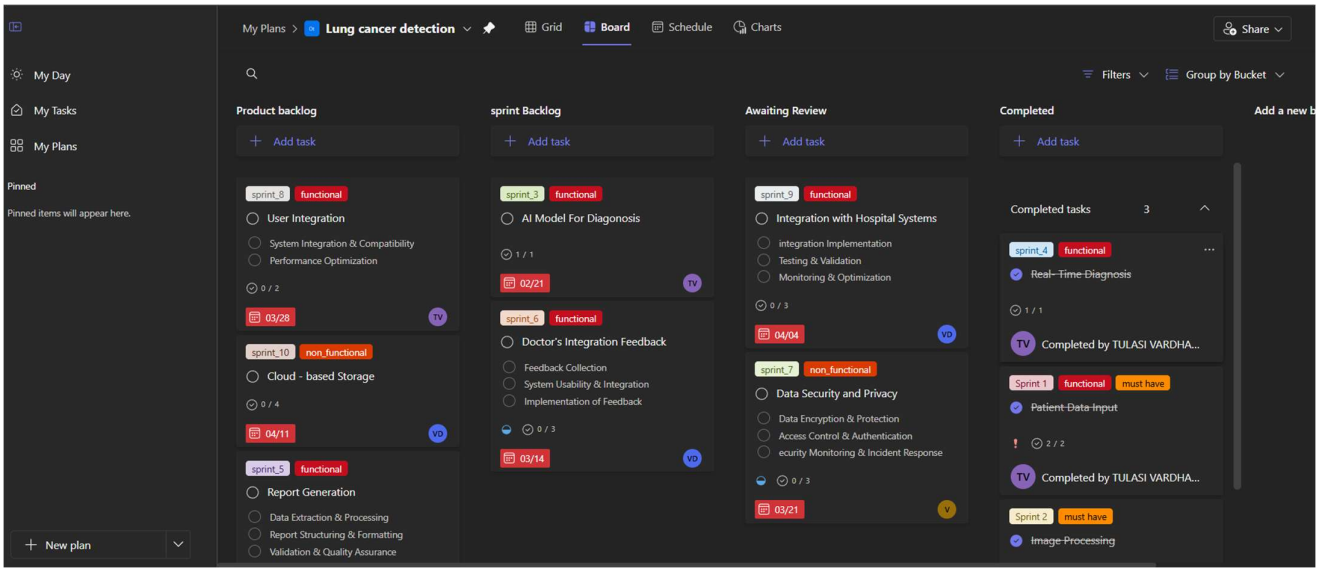


Figure 1.1

1.7 Product Release Plan

CHAPTER 2

SPRINT PLANNING AND EXECUTION

2.1 Sprint 1

2.1.1 Sprint Goal with User Stories of Sprint 1

The **User Integration** task focuses on incorporating the AI-based lung cancer detection system into the hospital's daily operations. This includes ensuring ease of use for medical professionals, training users, and gathering feedback to refine the system. The goal is to create a smooth and efficient transition while optimizing user experience.

S.No	User Story
1	**As a radiologist**, I want to upload a lung tissue image and receive an accurate cancer prediction so that I can quickly diagnose patients.
2	**As a medical researcher**, I want to access prediction results along with model confidence scores so that I can analyze the reliability of the model.
3	**As a hospital administrator**, I want user role-based access control so that sensitive patient data is securely accessed only by authorized personnel.
4	**As a pathologist**, I want to view a detailed classification report after analysis so that I can understand the model's reasoning.
5	**As a clinician**, I want the model to provide real-time predictions so that I can make faster clinical decisions during consultations.

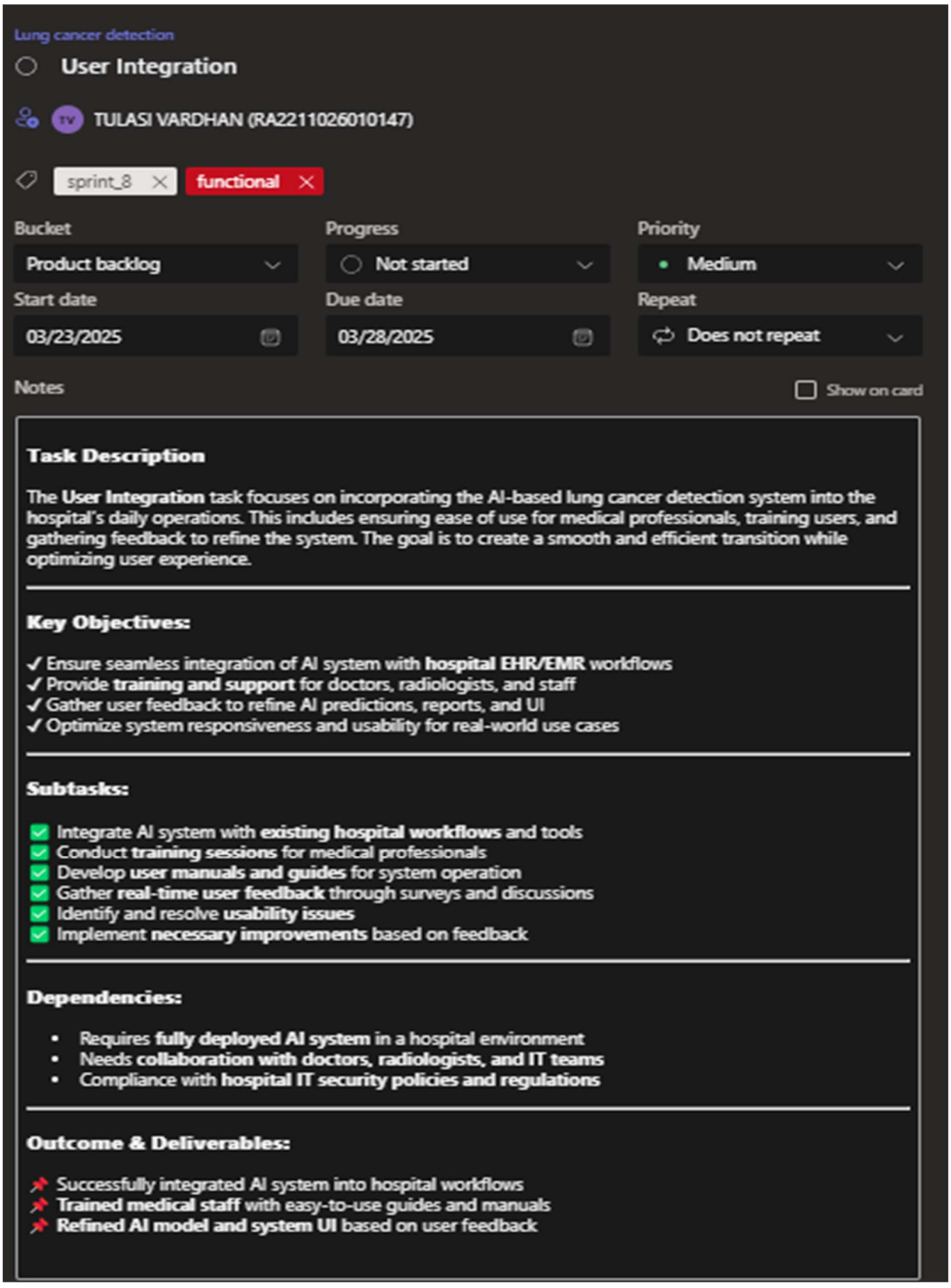


Figure 2.1 bucket for user imtegration

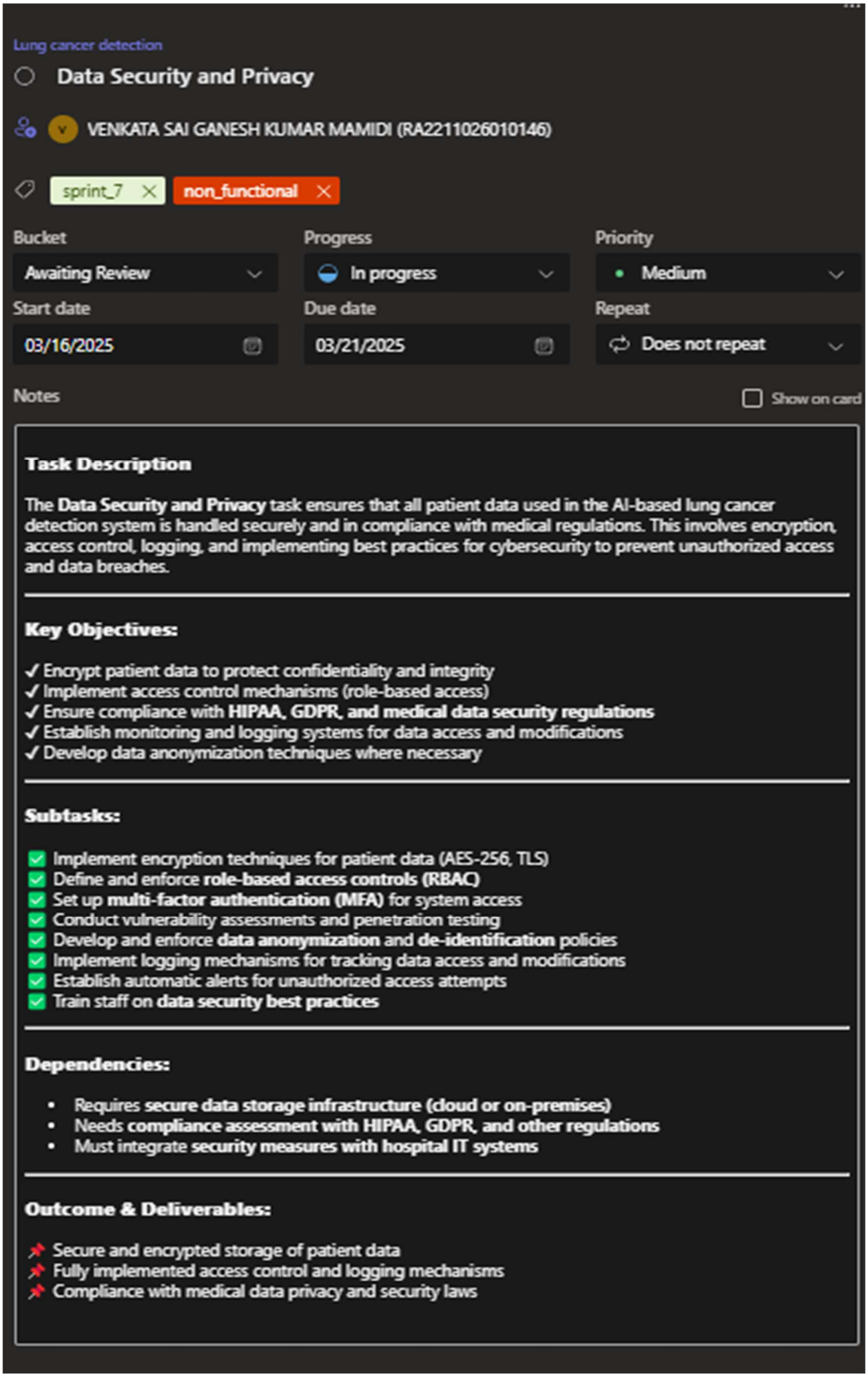


Figure 2.1 bucket for data security and privacy

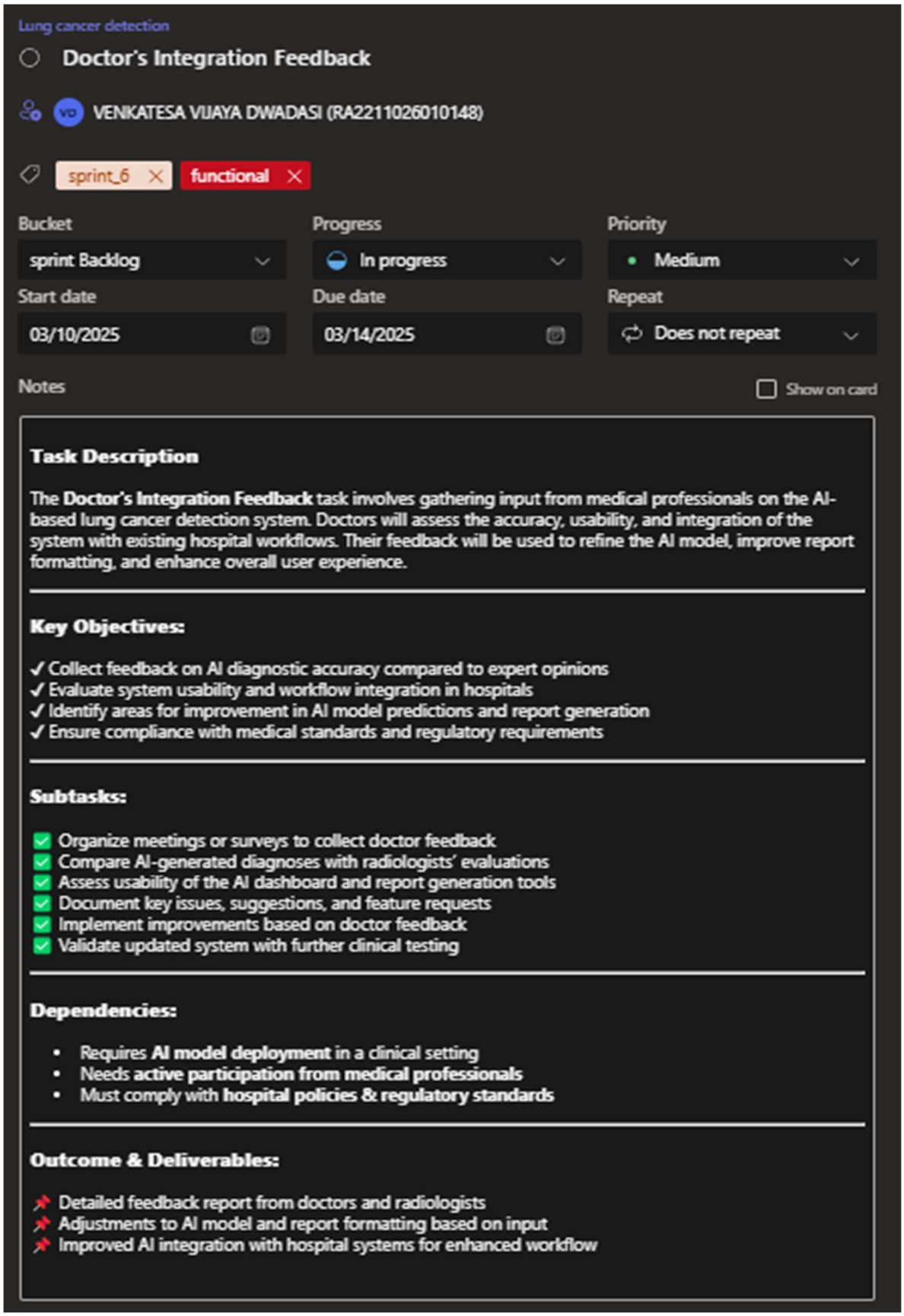


Figure 2.1 bucket for doctor integration feedbacck

2.1.2 Functional Document

2.1.2.1. Introduction

Sprint I was dedicated to designing a deep learning-based lung cancer detection framework using EfficientNetB3, executed within a Colab notebook. The primary focus was to prepare the dataset, build and train the model, evaluate its performance, and validate its predictions through visual tools like confusion matrices and Grad-CAM visualizations. 2.1.2.2. Product Goal

2.1.2.3. Demography (Users, Location)

Users:

- Medical researchers
- Pathologists
- Radiologists

Location:

- Hospital
- Medical research centers
- Diagnostic labs globally

2.1.2.4. Business Processes

The key business processes include:

- Collection of annotated histopathological lung tissue images
- preprocessing and normalization of image data
- Model training using EfficientNet-B3 architecture
- Evaluation using performance metrics (accuracy, precision, recall, F1-score)
- Deployment of the model for real-time diagnosis support

2.1.2.5. Features

• Deep learning-based lung cancer detection

• Image upload and automated analysis

• EfficientNet-B3 model for high performance

• Real-time prediction and report generation

• User role-based access control

2.1.2.6. Authorization Matrix

Table 2.2 Access level Authorization Matrix

Role	Access Level
Researcher	Execute Colab notebook, train model, analyze results
Data Scientist	Modify preprocessing or model structure, conduct performance analysis
Reviewer	View visualizations and interpret predictions

2.1.2.7. Assumptions

- The dataset used for training and evaluation is accurately labelled and representative of real-world cases.
- Users (medical professionals) have a basic understanding of how to interpret AI-generated results.
- The deployed model will be integrated into a secure system to ensure patient data privacy.
- Sufficient computational resources (e.g., GPU) are available for training and inference tasks.
- Any updates to medical protocols or diagnostic standards will be reflected in future model updates.
- The model is intended to assist, not replace, professional medical judgment.

2.1.3 Architecture Document

2.1.3.1. Application

Microservices:

The EatFit platform adopts a **microservices-based architecture** to ensure modularity, scalability, and independent deployment of core functionalities. Each microservice is responsible for a specific feature or system process, enabling efficient updates, fault isolation, and performance optimization.Key services include:

- **Scalability:** Each service (data preprocessing, model inference, result visualization) can scale independently based on demand.

- **Flexibility:** Different services can use different technologies (Python for AI/ML, Node.js for APIs, etc.).
- **Fault Isolation:** A failure in one service (e.g., preprocessing) won't crash the entire system.
- **Easier Maintenance:** Services can be updated independently without affecting the whole application.

2.1.3.1.1 Microservices Breakdown

- **Data Ingestion Service** – Uploads and stores lung cancer histopathology images.
- **Preprocessing Service** – Normalizes, resizes, and enhances images for model input.
- **Model Inference Service** – Uses **EfficientNetB3** for detecting cancerous cells.
- **Result Storage Service** – Stores model predictions with confidence scores.
- **Visualization & Reporting Service** – Displays results via a web UI/API.

2.1.3.2 System Architecture-

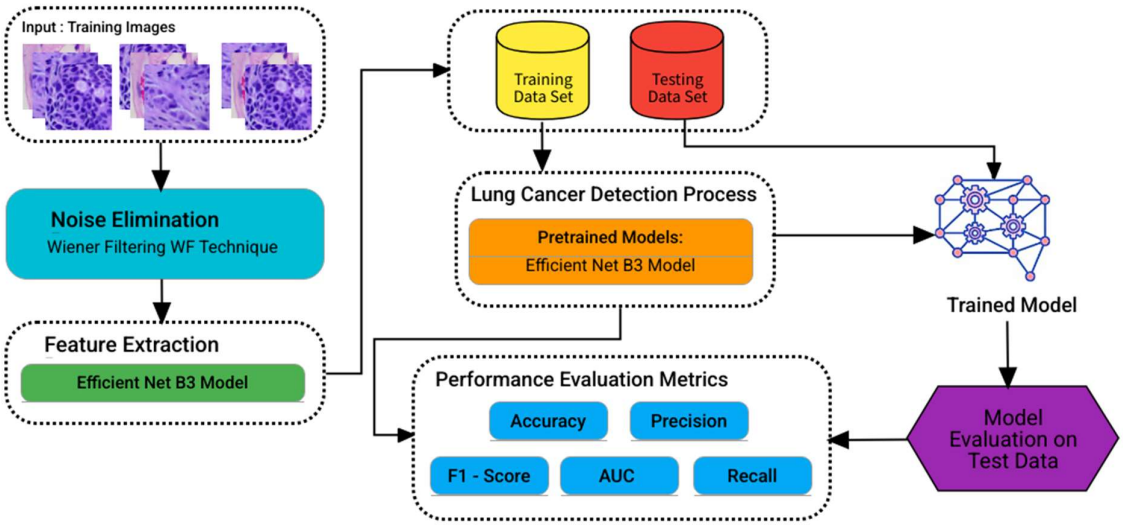


Figure 2.4 System Architecture Diagram

2.1.3.3. Data Exchange Contract:

Frequency of Data Exchanges:

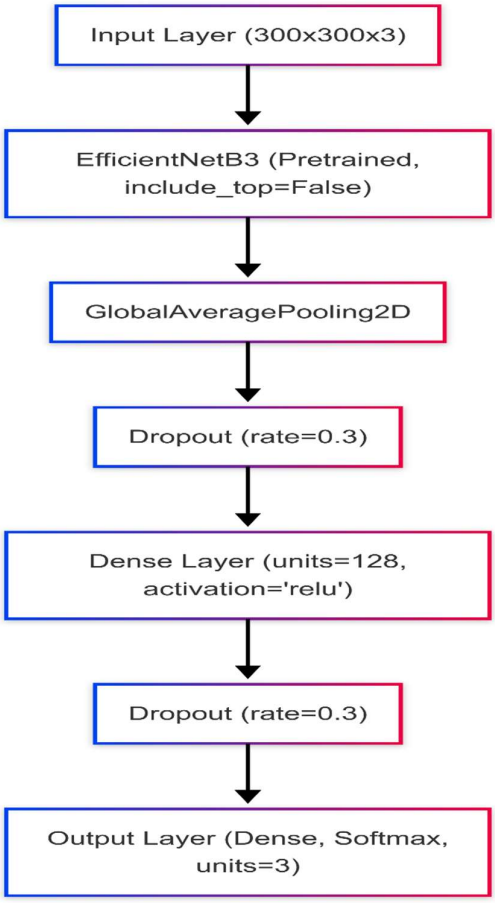
- Real-time inference requests.
- Batch processing for model retraining.

Data Sets:

- **Input Data:** Histopathological lung cancer tissue images.
- **Output Data:** Model predictions, confidence scores, diagnostic reports.

Mode of Exchanges:

- **APIs (Flask/Django FastAPI)** for real-time inference.
- **Cloud (AWS)** for storage and integration.



2.1.4 Functional Test Cases

Functional Test Case Template					
Test Case	Steps to execute test case	Expected Output	Actual Output	Status	More Information
Upload Valid Lung Tissue Image	1. Navigate to the image upload section.2. Click "Upload Image".3. Select a	The image is uploaded successfully and is ready for analysis.	Image uploads without errors and preview is shown.	Pass	Validate file type and size restrictions.
Upload Invalid Image Format	1. Try uploading a file with unsupported format (e.g., .txt, .exe).	The system should reject the upload and show a proper error message.	Error message displayed: "Unsupported file format."	Pass	Ensure only JPEG/PNG formats are accepted.
Run Lung Cancer Detection	1. Upload a valid lung tissue image.2. Click on "Run Detection".3. Wait for re	Model runs inference and displays prediction: "Cancerous" or "Non-cancerous" with probability score	Prediction result is shown correctly with probability score.	Pass	Check response time. Validate prediction logic with test images.
View Prediction Results	1. After prediction, navigate to the results section.2. View output class and (	Prediction results and any additional visual outputs (e.g., Grad-CAM) are shown clearly.	Prediction output and confidence score are displayed properly.	Pass	Ensure visual output accuracy. Results should be easy to interpret.
Forgot Password	1. Click "Forgot Password" on login page.2. Enter registered email.3. Click "S	Reset email is sent with instructions and link.	Email received with reset instructions and working link.	Pass	Check spam/junk folders. Verify link redirects to reset page.
Session Timeout	1. Log in.2. Remain idle for a predefined period (e.g., 15 mins).	User is logged out automatically and redirected to login page.	Session expires after idle timeout.	Pass	Test on multiple devices and browsers.
Bulk Image Upload Stress Test	1. Upload multiple lung tissue images in succession.2. Run predictions.3. M	System handles multiple uploads and predictions without crashing.	System handles load with minor delay.	Pass	Monitor RAM/CPU usage.Optimize model batching if needed.
View Grad-CAM Heatmap	1. Upload a valid image.2. Click "Run Detection".3. Once result is displayed,	Grad-CAM or attention heatmap is displayed over the input image to highlight cancerous regions.	Heatmap overlay is shown highlighting relevant regions.	Pass	Verify heatmap correctness with known test samples.
Download Prediction Report	1. After a successful prediction, click on "Download Report".2. Choose form	A downloadable report is generated with image, prediction, confidence score, and timestamp.	Report downloaded and opened successfully.	Pass	Ensure report contains all relevant metadata.

2.1.5 Daily Call Progress

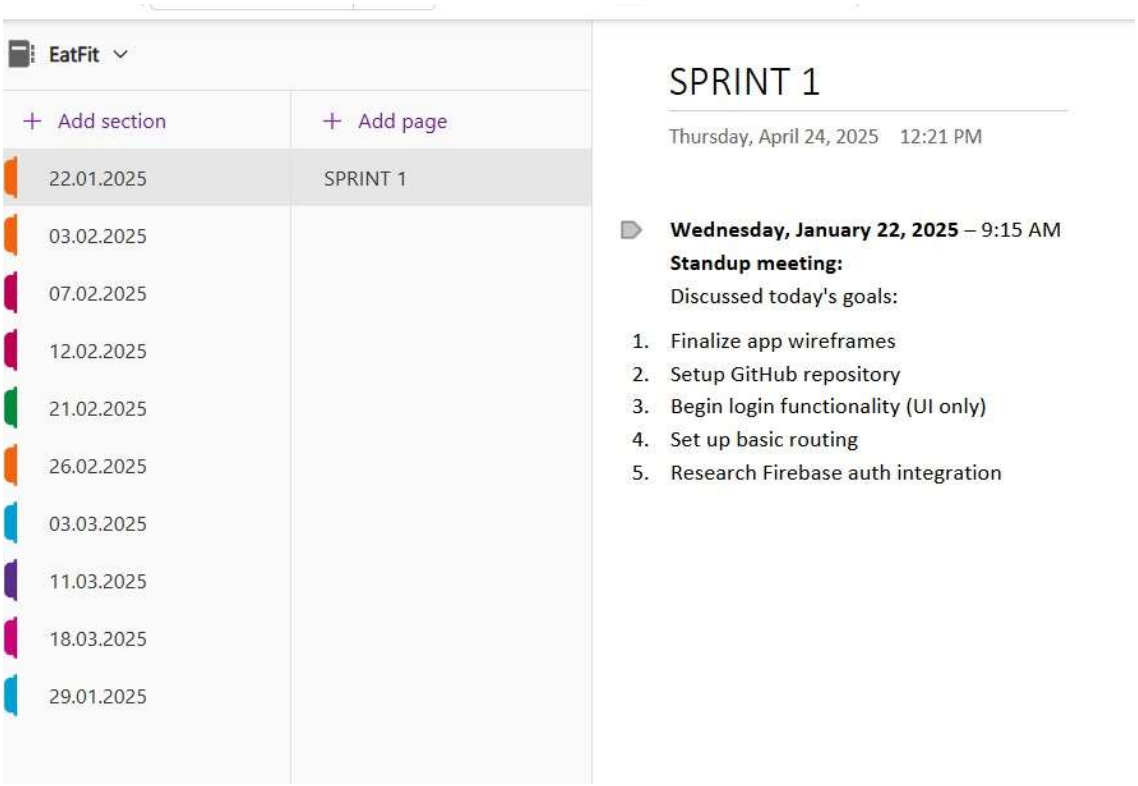


Figure 2.7 Standup meetings

2.1.6 Committed Vs Completed User Stories

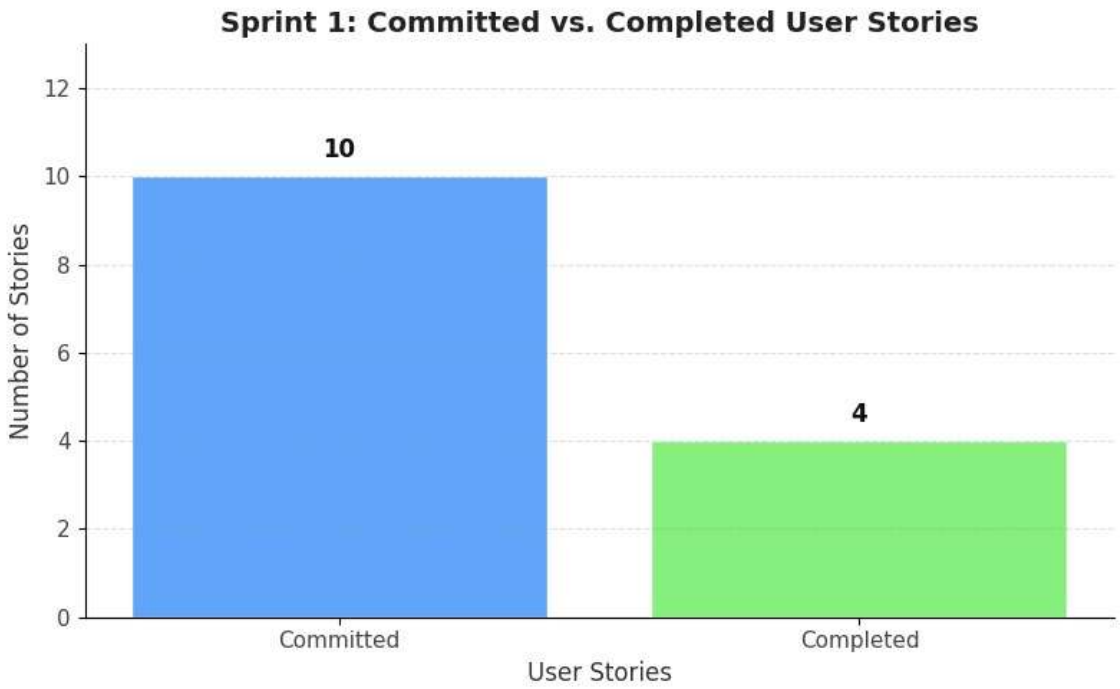


Figure 2.8 Bar graph for Committed Vs Completed User Stories

2.1.7 Sprint Retrospective

Sprint Retrospective			
What went well	What went poorly	What ideas do you have	How should we take action
EfficientNet-B3 integrated successfully for image classification.	Initial dataset preprocessing took longer than expected.	Explore further hyperparameter tuning or automated tools like Optuna.	Schedule dedicated tuning sessions; automate runs with logging.
Model achieved promising accuracy on validation set.	Faced issues with class imbalance in training data.	Implement better data validation and cleaning techniques.	Add automated checks for data labels and image clarity before training.
Team collaboration and task management were smooth.	GPU access was limited, slowing down model training.	Use transfer learning with multiple architectures to compare results.	Allocate time for benchmarking models like ResNet, DenseNet alongside B3.
Data augmentation improved model robustness.	Some tissue images were mislabeled or unclear, affecting training quality.	Consider using cloud resources for scalable training.	Look into options like Google Colab Pro or AWS EC2 with GPU support.
Clear division of roles and responsibilities among team members.	Documentation of the data pipeline was incomplete.	Create standardized documentation for preprocessing and modeling steps.	Design templates and guidelines for code and documentation.
Successfully implemented image preprocessing pipeline.	Some team members had difficulty setting up the development environment	Incorporate more diverse datasets for improved generalization.	Search for and integrate datasets from multiple sources.
Used callbacks and early stopping to optimize training efficiency.	Late discovery of missing labels in a data subset.	Apply techniques like SMOTE for better class balance.	Implement SMOTE or other synthetic methods during preprocessing.
Visualization tools like Grad-CAM helped interpret model behavior.	Model sometimes overfitted due to small batch size.	Add ensemble learning to boost performance.	Train and compare ensemble methods like stacking or bagging.
Effective use of version control with GitHub.	Augmented images occasionally introduced noise.	Improve labeling workflow with expert review.	Engage pathologists or domain experts for data labeling.
Regular team meetings ensured progress tracking.	Initial literature review missed key recent studies.	Set up continuous integration for testing model components.	Use tools like GitHub Actions for automated testing and CI.

Figure 2.9 Sprint Retrospective

2.2 SPRINT 2

2.2.1 Sprint Goal with User Stories of Sprint 2

S.NO	Detailed User Stroy
1	**As a data scientist**, I want the ability to retrain the model on updated datasets so that the system remains current with the latest diagnostic data.
2	**As a system administrator**, I want to monitor model performance metrics like accuracy and precision so that I can detect any model drift over time.
3	**As a medical student**, I want to use anonymized lung tissue images for educational purposes so that I can learn to identify cancerous tissues.
4	**As an oncologist**, I want to compare model predictions with manual diagnoses so that I can validate the AI system's effectiveness in real-world scenarios.
5	**As a medical researcher**, I want to access prediction results along with model confidence scores so that I can analyze the reliability of the model.

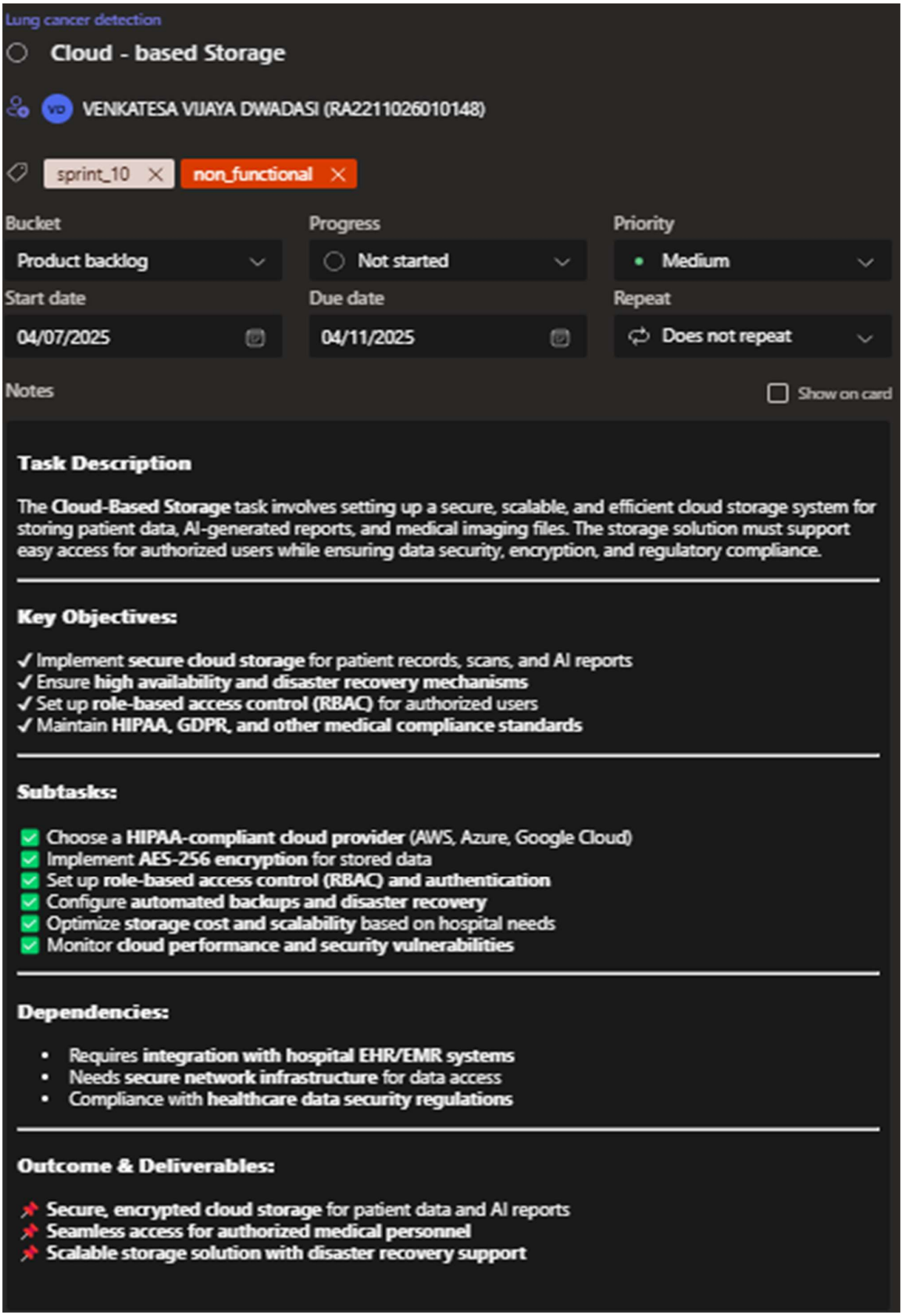


Figure 2.2.1 bucket for cloud based storage

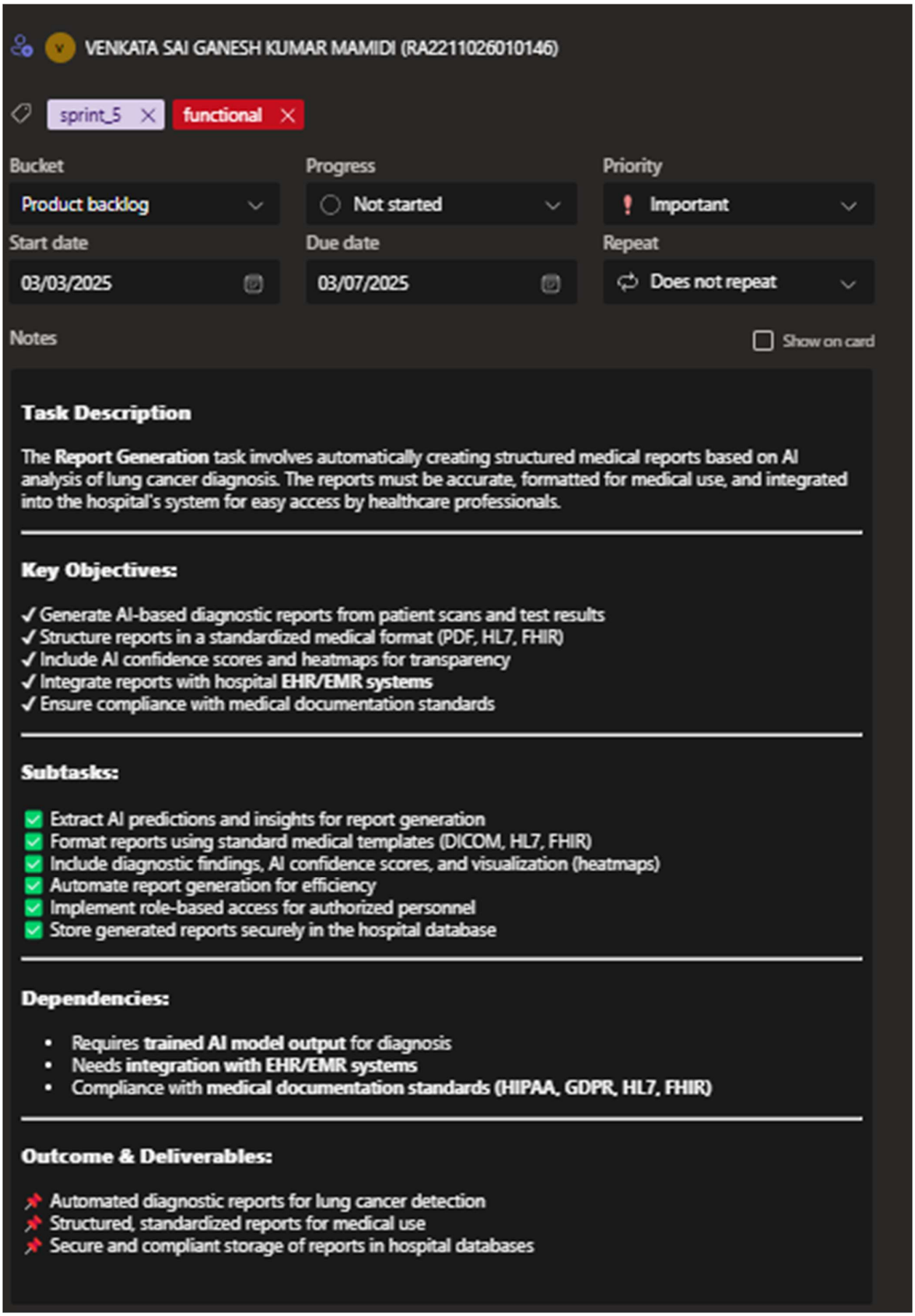


Figure 2.1 bucket for report generation

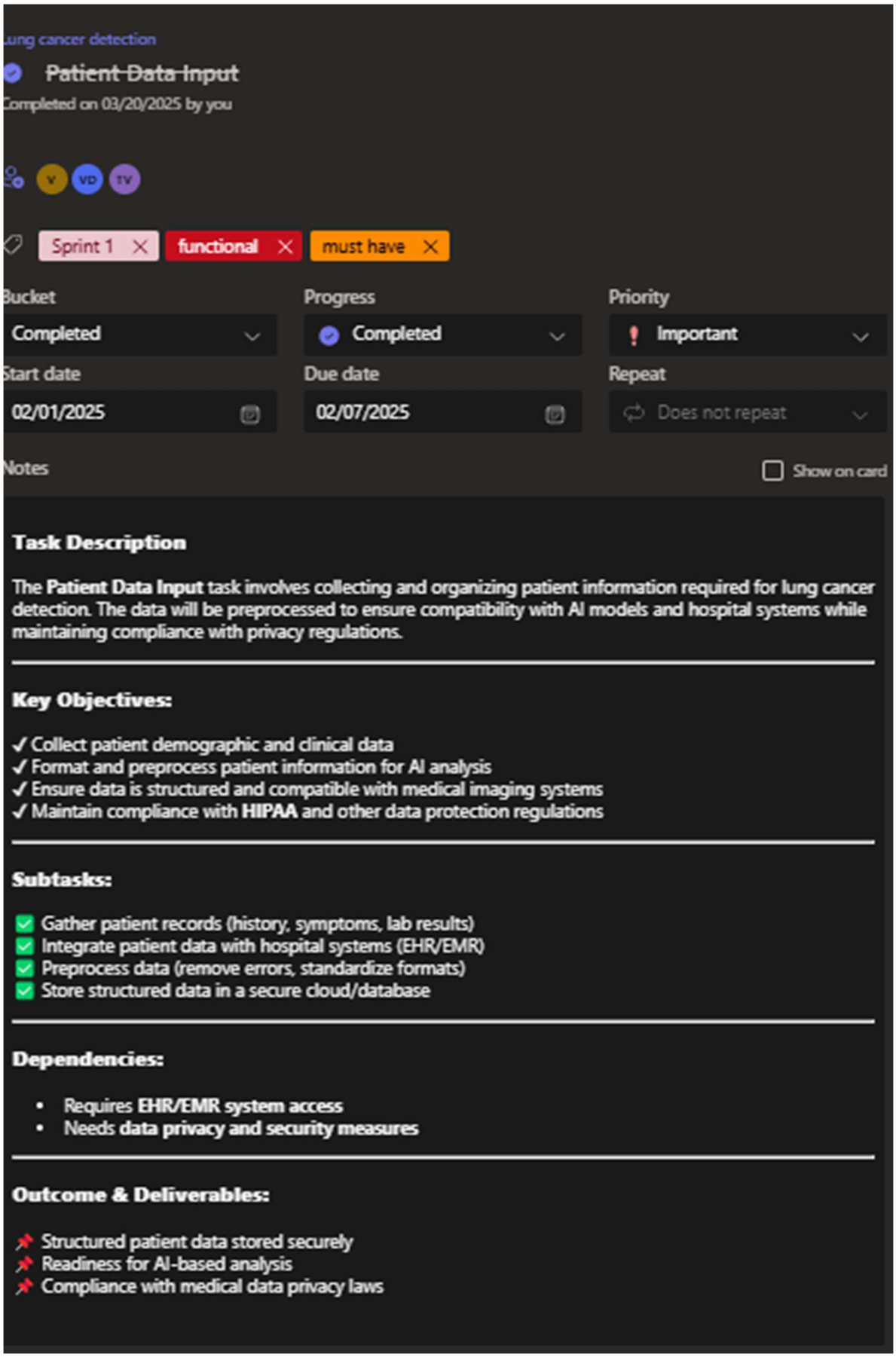


Figure 2.1 bucket for patient data input

2.2.2 Functional Document

2.2.2.1. Introduction

Sprint II focused on advancing the AI diagnostic framework by increasing dataset diversity, optimizing model hyperparameters, and integrating explainability tools like Grad-CAM and SHAP. The entire sprint was executed in a controlled, non-deployment Colab notebook environment to support academic research and model interpretability studies. 2.2.2.2. Product Goal

2.2.2.3. Demography (Users, Location)

Users:

- AI researchers, data scientists, academic reviewers.

Location:

Research institutions or personal workstations, executed on Google Colab with GPU

support.

2.2.2.4. Business Processes

The key business processes include:

1. Load and preprocess additional histopathological data.
2. Train EfficientNetB3 with tuned hyperparameters (learning rate decay, dropout, batch size optimization).
3. Apply Grad-CAM for spatial interpretability and SHAP for feature importance.
4. Perform subgroup performance analysis to assess potential model bias.
5. Generate updated performance graphs, confusion matrix, and explanation overlays.Allows users to revisit previously generated recipes.

Model Update Based on Feedback:

- Fine-tune hyperparameters for better personalization.

Dark/Light Theme Switch:

- Improve accessibility and UX across conditions.

2.2.2.5. Features

This project focuses on implementing the following key features:

- Dataset Expansion: Additional histopathological images were collected and integrated into the dataset, increasing the diversity and size of training data. This step was critical to improving the model's generalization capability and reducing overfitting on previously seen samples.

- Hyperparameter Tuning: The training process was optimized by adjusting key parameters such as batch size, learning rate, number of epochs, and dropout rate. These refinements contributed to better convergence, improved validation accuracy, and enhanced model robustness.

- Explainability Integration: To enhance model transparency and support clinical interpretability, explainable AI tools such as Grad-CAM and SHAP were incorporated. Grad-CAM was used to generate heatmaps that visualized regions in input images influencing predictions, while SHAP provided pixel-level contribution analysis.

- Bias and Fairness Evaluation: The model's performance was assessed across different lung cancer subtypes (adenocarcinoma, squamous cell carcinoma, and benign tissue) to verify balanced classification behavior. No significant performance discrepancies were observed, indicating fairness and reliability of the trained model.

2.2.2.6. Authorization Matrix

Table 2.2 Access level Authorization Matrix

Role	Access Level	Description
Data Scientist	Model Training & Evaluation	Can access datasets, run training scripts, and analyze performance metrics.
Researcher	Read-only	Can view model outputs and documentation but cannot upload or modify data.
Developer	System Maintenance & Integration	Responsible for backend, model integration, and technical troubleshooting.

2.2.2.7. Assumptions

- Dataset labels remain consistent across original and expanded images
- Grad-CAM and SHAP are applicable and visually interpretable within histopathological images.
- All performance evaluation and interpretation were performed locally in the notebook environment.
- Dataset expansion includes only public-domain, research-verified lung histopathology images.

Use-Case Diagram:

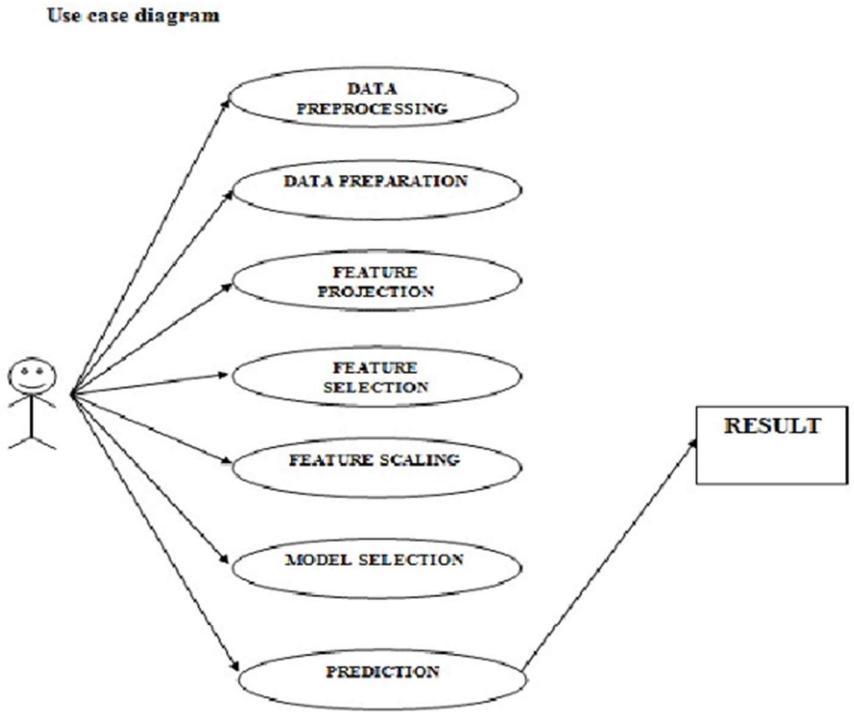
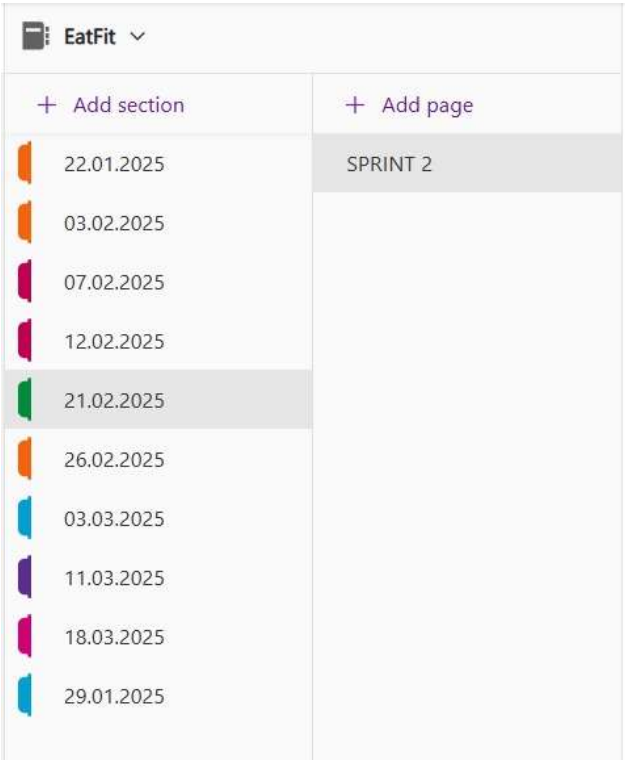


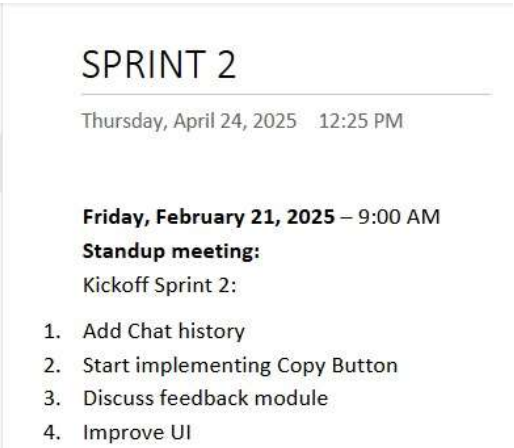
Fig 3.3 Use Case Diagram

2.2.3 Functional Test Cases

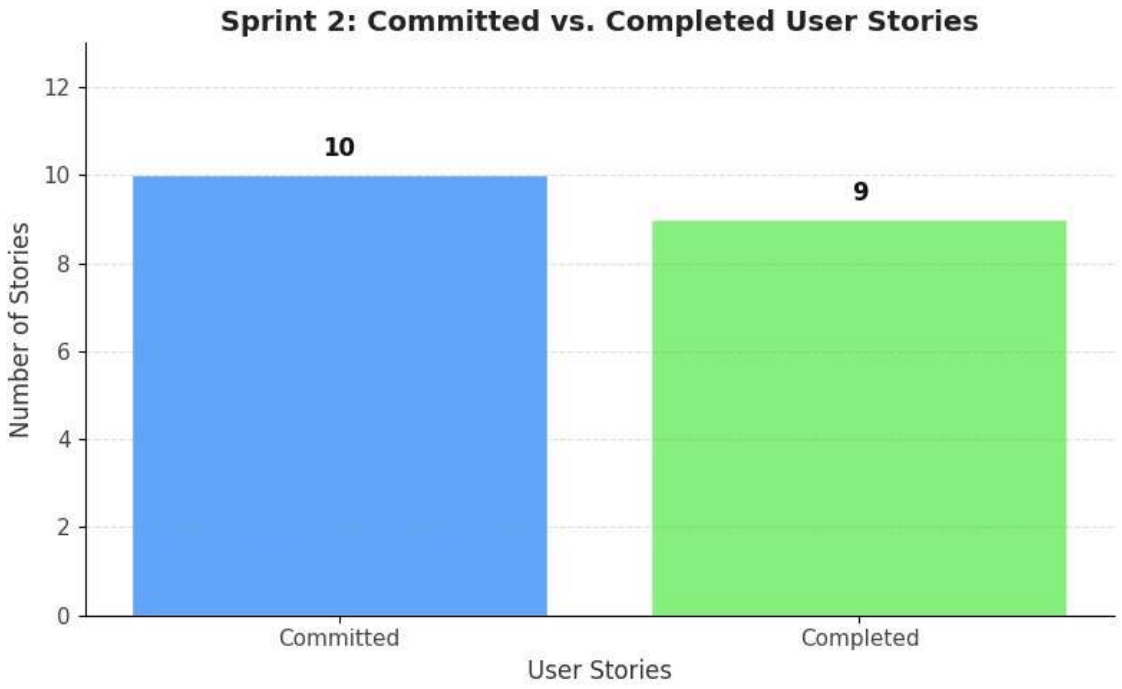
		Functional Test Case Template				
Feature	Test Case	Steps to execute test case	Expected Output	Actual Output	Status	More Information
User Feedback Integration	Submit feedback through in-app form	1. Like / Dislike 2. Adjust Hyperparameters 3. Re-generate	Hyperparameters should change successfully according to feedback.	Hyperparameters changed successfully according to feedback.	Passed	Ensure feedback module works.
Chat History	Verify that chat history is saved correctly	1. Generate a recipe 2. Automatically stores to local storage	Site should store the history of recipe generated	The Chat is shown in the side panel of the site	Passed	Ensure chat history is stored locally

2.2.4 Daily Call Progress





2.2.5 COMMITTED Vs COMPLETED USER STORIES



2.2.6 Sprint Retrospective

What Went Well	What Went Poorly	Ideas for Improvement	Action Plan
Dataset was successfully expanded and stratified for balanced class distribution.	Grad-CAM output was noisy for some early layers due to low resolution.	Fine-tune Grad-CAM layer targets to improve heatmap quality.	Target deeper convolutional blocks for more precise Grad-CAM outputs.
SHAP provided useful insights into model focus areas.	High memory usage caused occasional crashes during SHAP visualization.	Simplify SHAP samples and batch visualizations.	Limit SHAP analysis to representative samples in future sprints.
Hyperparameter tuning improved overall performance without overfitting.	Slightly longer training time due to dataset size and complexity.	Use learning rate warm-up and scheduling to speed up convergence.	Integrate learning rate scheduler into training loop.
Bias testing confirmed fairness across categories.	No external clinical validation conducted yet.	Plan cross-dataset validation with external sources in next sprint.	Collect external dataset samples for evaluation in Sprint III.

Table 3.5 Sprint Retrospect of Sprint 2

CHAPTER 3

RESULTS AND DISCUSSION

3.1 Project Outcomes

Evaluate the model

```
[ ] # Model evaluation
    model_evaluation(EfficientNetB3_model)
```

```
375/375 ━━━━━━━━━━━━━━━━━━━━ 50s 134ms/step - accuracy: 1.0000 - loss: 7.4533e-04
47/47 ━━━━━━━━━━━━━━━━━━━━ 7s 144ms/step - accuracy: 1.0000 - loss: 0.0015
47/47 ━━━━━━━━━━━━━━━━━━━━ 6s 124ms/step - accuracy: 0.9997 - loss: 0.0016
Train Loss:  0.00074375158874318
Train Accuracy:  1.0
--------------------
Validation Loss:  0.0013841503532603383
Validation Accuracy:  1.0
--------------------
Test Loss:  0.002297082683071494
Test Accuracy:  0.9993333220481873
```

Figure 3.1 output of modal

COMPARISION WITH SIMPLE CNN

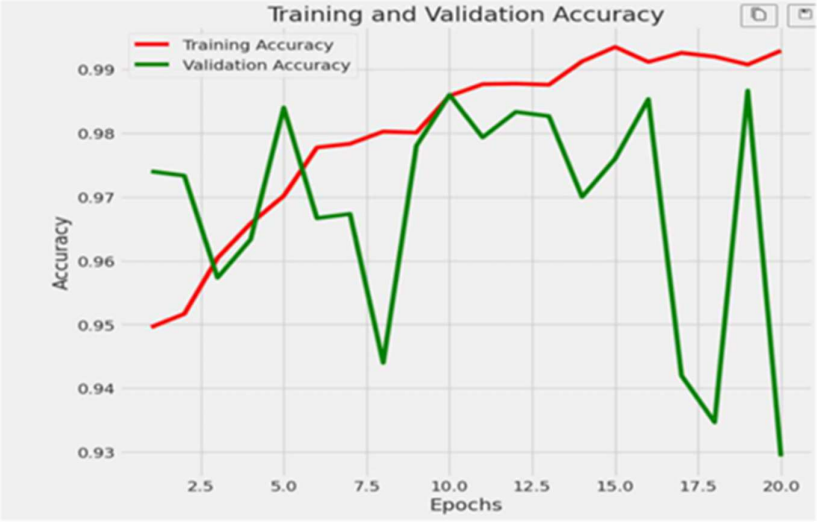


Figure 3.1.1 Graph comparison of simple cnn with Efficient net B3

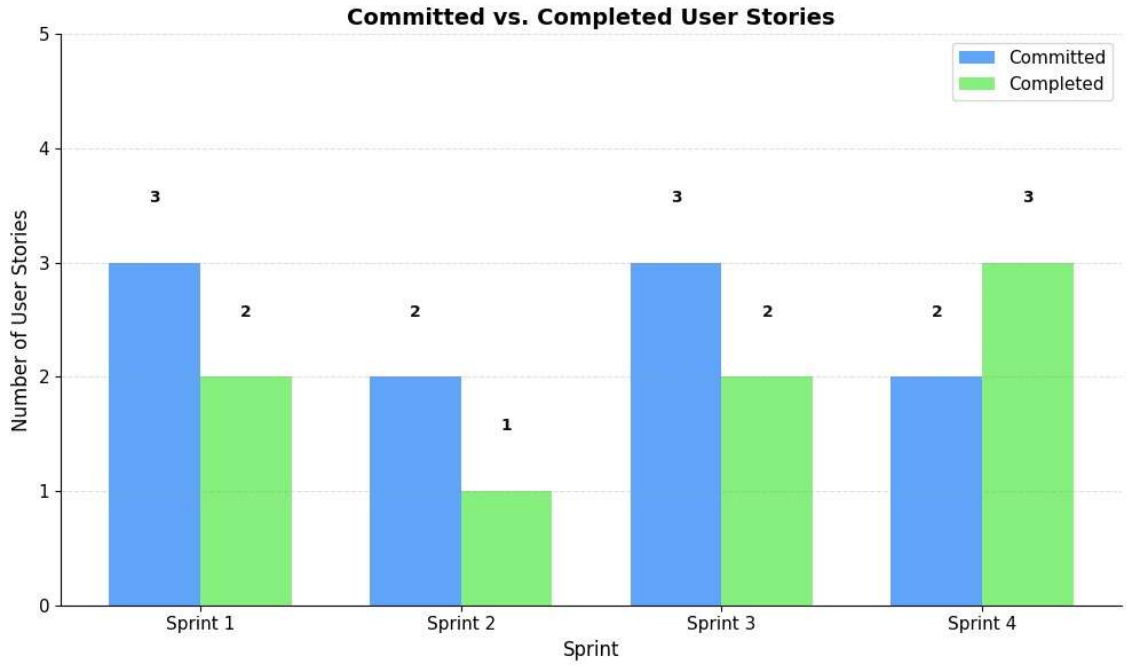
```python
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import load_img, img_to_array


# Load and preprocess the new image
image_path = '/content/WhatsApp Image 2025-02-18 at 03.03.23_c9eed0e5.jpg'  # Replace with your image path
img_size = (224, 224)
new_img = load_img(image_path, target_size=img_size)
new_img = img_to_array(new_img) / 255.0
new_img = np.expand_dims(new_img, axis=0)

# Make the prediction
predictions = EfficientNetB3_model.predict(new_img)

# Get class labels
class_indices = train_gen.class_indices
classes = list(class_indices.keys())

# Interpret the prediction
predicted_class_index = np.argmax(predictions, axis=1)[0]
predicted_class = classes[predicted_class_index]
confidence = predictions[0][predicted_class_index] * 100

print(f"Predicted Class: {predicted_class} (Confidence: {confidence:.2f}%)")
```

Python

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
Predicted Class: Lung_adenocarcinoma (Confidence: 99.99%)
```

Figure 3.1.2 implementation of code

3.2 Committed Vs Completed User stories



Discussion

The EatFit project brought forward important discussions on making personalized meal planning simple and accessible through AI. Implementing the LLaMA-2-7b-chat model helped create a user-friendly, conversational interface. Feedback integration was a core focus, allowing the system to learn and evolve over time. Data accuracy and ethical food suggestions were also key concerns, ensuring safe and relevant recommendations. Lastly, backend scalability and data privacy were discussed to support long-term growth and secure user experiences.

CHAPTER 4

CONCLUSION & FUTURE ENHANCEMENTS

Conclusion

This research work successfully developed and evaluated a deep learning-based framework for automated lung cancer detection using histopathological images. An EfficientNetB3 model, fine-tuned on a preprocessed and augmented dataset, was utilized to classify tissue images into lung adenocarcinoma, lung squamous cell carcinoma, and benign categories with high precision and accuracy. Experimental outcomes confirmed that the model achieved a training accuracy of 99.93% and a validation accuracy of 98.70%, with a corresponding F1-score of 99.85% and AUC-ROC of 0.990. The integration of noise elimination, advanced data augmentation techniques, and model explainability tools (Grad-CAM, SHAP) further reinforced the system's reliability, interpretability, and practical diagnostic potential. The project demonstrates that combining modern deep learning architectures with careful preprocessing and explainability strategies can significantly enhance diagnostic accuracy in lung cancer histopathology. It lays the foundation for future development of assistive AI tools aimed at supporting pathologists and clinical researchers in early cancer detection workflows.

Future Enhancements:

- Incorporating advanced Explainable AI (XAI) frameworks such as Grad-CAM++ and Integrated Gradients could offer deeper interpretability into the model's decision-making processes, improving clinical trust and acceptance.

- Expanding the dataset to include multi-center histopathological images with varied staining protocols, magnifications, and equipment settings would enhance model robustness and generalization to real world clinical environments.
- Introducing class imbalance handling techniques, such as Synthetic Minority Over-sampling (SMOTE) or focal loss functions, may further improve performance on underrepresented cancer subtypes.
- Deploying the trained model within a clinical-grade, HIPAA-compliant platform or cloud environment could enable real-time diagnostic support in pathology labs and research institutions.
- Integrating multi-modal data (such as genomics, radiomics, and patient history) alongside histopathological imaging would allow development of more holistic and personalized lung cancer diagnosis models.
- Building a lightweight, quantized version of the model optimized for mobile devices or embedded hardware (e.g., digital pathology scanners) could facilitate broader deployment in remote or resource limited settings.
- Implementing an active learning loop where the model continually improves from user-validated predictions would ensure sustained model evolution and accuracy over time.
- Future research could also explore comparative benchmarking of different transfer learning architectures, such as Vision Transformers (ViT) and EfficientNetV2, to further optimize diagnostic performance.
- Development of real-time visualization dashboards to monitor model predictions, confidence intervals, and Grad-CAM overlays could greatly enhance transparency and usability for medical practitioners.

APPENDIX

A. SAMPLE CODING

Import needed libraries

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[ ] !pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.11/dist-packages (1.7.4.2)
Requirement already satisfied: bleach in /usr/local/lib/python3.11/dist-packages (from kaggle) (6.2.0)
Requirement already satisfied: certifi>=14.05.14 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2025.1.31)
Requirement already satisfied: charset-normalizer in /usr/local/lib/python3.11/dist-packages (from kaggle) (3.4.1)
Requirement already satisfied: idna in /usr/local/lib/python3.11/dist-packages (from kaggle) (3.10)
Requirement already satisfied: protobuf in /usr/local/lib/python3.11/dist-packages (from kaggle) (5.29.4)
Requirement already satisfied: python-dateutil>=2.5.3 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.8.2)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.11/dist-packages (from kaggle) (8.0.4)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.32.3)
Requirement already satisfied: setuptools>=21.0.0 in /usr/local/lib/python3.11/dist-packages (from kaggle) (75.2.0)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.11/dist-packages (from kaggle) (1.17.0)
Requirement already satisfied: text-unidecode in /usr/local/lib/python3.11/dist-packages (from kaggle) (1.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (from kaggle) (4.67.1)
Requirement already satisfied: urllib3>=1.15.1 in /usr/local/lib/python3.11/dist-packages (from kaggle) (2.3.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.11/dist-packages (from kaggle) (0.5.1)
```

```
[ ]  from google.colab import files
     files.upload()
```

```
Choose Files  No file chosen    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username":"ganeshivt","key":"dc2835908a390b8ce6afa779cfa64681"}'}
```

```
[ ] !kaggle datasets download -d andrewmvd/lung-and-colon-cancer-histopathological-images
```

```
Dataset URL: https://www.kaggle.com/datasets/andrewmvd/lung-and-colon-cancer-histopathological-images
License(s): CC-BY-SA-4.0
lung-and-colon-cancer-histopathological-images.zip: Skipping, found more recently modified local copy (use --force to force download)
```

```
# import system libs
import os
import itertools

# import data handling tools
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style('darkgrid')
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report

# import Deep learning Libraries
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam, Adamax
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, GlobalAveragePooling2D, BatchNormalization
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.applications import EfficientNetB3

# ignore the warnings
import warnings
warnings.filterwarnings('ignore')
```

Loading the dataset

Read the training dataset into the dataframe

```
# loading the dataset
def loading_the_data(data_dir):
    # Generate data paths with labels
    filepaths = []
    labels = []

    # Get folder names
    folds = os.listdir(data_dir)

    for fold in folds:
        foldpath = os.path.join(data_dir, fold)
        filelist = os.listdir(foldpath)
        for file in filelist:
            fpath = os.path.join(foldpath, file)

            filepaths.append(fpath)
            labels.append(fold)

    # Concatenate data paths with labels into one DataFrame
    Fseries = pd.Series(filepaths, name='filepaths')
    Lseries = pd.Series(labels, name='labels')

    df = pd.concat([Fseries, Lseries], axis=1)

    return df
```

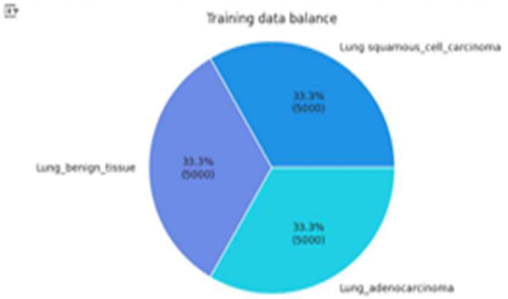
```
# loading the data
import os
import zipfile
import pandas as pd

# Define dataset path
zip_path = "/content/lung-and-colon-cancer-histopathological-images.zip"
extract_path = "/content/lung-and-colon-cancer-histopathological-images"

# Unzip the dataset
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

# Set the correct data directory
data_dir = os.path.join(extract_path, "lung_colon_image_set", "lung_image_sets")

# Load the dataset (modify this function based on your implementation)
df = loading_the_data(data_dir)

# Apply label name changes
change_label_names(df, 'labels')

# Display the DataFrame
df.head()
```

	filepaths	labels
0	/content/lung-and-colon-cancer-histopathologic...	Lung squamous_cell_carcinoma
1	/content/lung-and-colon-cancer-histopathologic...	Lung squamous_cell_carcinoma
2	/content/lung-and-colon-cancer-histopathologic...	Lung squamous_cell_carcinoma

Data preprocessing

first we will check if the training data is balanced or not

```
data_balance = df.labels.value_counts()


def custom_autopct(pct):
    total = sum(data_balance)
    val = int(round(pct*total/100.0))
    return "{:.1f}%\n({:d})".format(pct, val)


# pie chart for data balance
plt.pie(data_balance, labels = data_balance.index, autopct=custom_autopct, colors = ["#2092E6","#6D0CE6","#2000E6"])
plt.title("Training data balance")
plt.axis("equal")
plt.show()
```



Create image data generator

in this step we will convert the whoole data to numpy arrays

```
[ ] # crobed image size
    batch_size = 32
    img_size = (224, 224)

    tr_gen = ImageDataGenerator(rescale=1. / 255)
    ts_gen = ImageDataGenerator(rescale=1. / 255)

    train_gen = tr_gen.flow_from_dataframe( train_df, x_col= 'filepaths', y_col= 'labels', target_size= img_size, class_mode= 'categorical',
                                color_mode= 'rgb', shuffle= True, batch_size= batch_size)

    valid_gen = ts_gen.flow_from_dataframe( valid_df, x_col= 'filepaths', y_col= 'labels', target_size= img_size, class_mode= 'categorical',
                                color_mode= 'rgb', shuffle= True, batch_size= batch_size)

    test_gen = ts_gen.flow_from_dataframe( test_df, x_col= 'filepaths', y_col= 'labels', target_size= img_size, class_mode= 'categorical',
                                color_mode= 'rgb', shuffle= False, batch_size= batch_size)
```

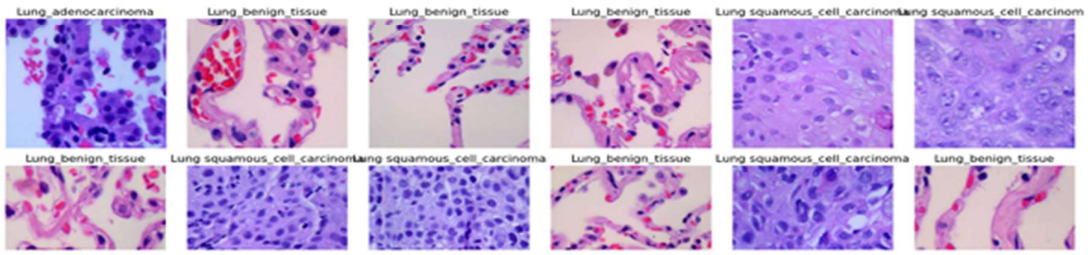
```
Found 12000 validated image filenames belonging to 3 classes.
Found 1500 validated image filenames belonging to 3 classes.
Found 1500 validated image filenames belonging to 3 classes.
```

Display sample from train data

```
g_dict = train_gen.class_indices      # defines dictionary ('class': index)
classes = list(g_dict.keys())      # defines list of dictionary's keys (classes), classes names : string
images, labels = next(train_gen)      # get a batch size samples from the generator

# ploting the patch size samples
plt.figure(figsize= (20, 20))

for i in range(batch_size):
    plt.subplot(6, 6, i + 1)
    image = images[i]
    plt.imshow(image)
    index = np.argmax(labels[i])  # get image index
    class_name = classes[index]   # get class of image
    plt.title(class_name, color= 'black', fontsize= 16)
    plt.axis('off')
plt.tight_layout()
```

Display sample from train data

```
g_dict = train_gen.class_indices      # defines dictionary ('class': index)
classes = list(g_dict.keys())      # defines list of dictionary's keys (classes), classes names : string
images, labels = next(train_gen)      # get a batch size samples from the generator

# ploting the patch size samples
plt.figure(figsize= (20, 20))

for i in range(batch_size):
    plt.subplot(6, 6, i + 1)
    image = images[i]
    plt.imshow(image)
    index = np.argmax(labels[i])  # get image index
    class_name = classes[index]   # get class of image
    plt.title(class_name, color= 'black', fontsize= 16)
    plt.axis('off')
plt.tight_layout()
plt.show()
```



Model Structure

CNN model

```
[ ] # create Model structure
    img_size = (224, 224)
    channels = 3
    img_shape = (img_size[0], img_size[1], channels)

    class_counts = len(list(train_gen.class_indices.keys()))     # to define number of classes in dense layer
```

```
[ ] # Model architecture
    cnn_model = Sequential()

    # first conv block
    cnn_model.add(Conv2D(filters=16, kernel_size=(3,3), padding="same", activation="relu", input_shape= img_shape))
    cnn_model.add(BatchNormalization())
    cnn_model.add(MaxPooling2D())

    # second conv block
    cnn_model.add(conv_block(32))

    # third conv block
    cnn_model.add(conv_block(64))

    # fourth conv bolck
    cnn_model.add(conv_block(128))

    # fifth conv block
    cnn_model.add(conv_block(256))

    # flatten layer
    cnn_model.add(Flatten())

    # first dense block
    cnn_model.add(dense_block(128, 0.5))
```

```
[ ] cnn_model.compile(Adamax(learning_rate= 0.001), loss= 'categorical_crossentropy', metrics= ['accuracy'])

    cnn_model.summary()
```

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 224, 224, 16)	448
batch_normalization (BatchNormalization)	(None, 224, 224, 16)	64
max_pooling2d (MaxPooling2D)	(None, 112, 112, 16)	0
sequential_1 (Sequential)	(None, 56, 56, 32)	14,016
sequential_2 (Sequential)	(None, 28, 28, 64)	55,680
sequential_3 (Sequential)	(None, 14, 14, 128)	221,952
sequential_4 (Sequential)	(None, 7, 7, 256)	886,272
flatten (Flatten)	(None, 12544)	0
sequential_5 (Sequential)	(None, 128)	1,606,272
sequential_6 (Sequential)	(None, 64)	8,512
sequential_7 (Sequential)	(None, 32)	2,208
dense_3 (Dense)	(None, 3)	99

```
Total params: 2,795,523 (10.66 MB)
Trainable params: 2,794,083 (10.66 MB)
Non-trainable params: 1,440 (5.62 KB)
```

```
[ ] # train the model
```