Final Project

Oral Cancer Prediction

Team Members:

1-Rahma Ayman Salah
2-Rahma Ashraf Abdelaziz
3-Shahd Ayman Khairy
4-Ibrahim Belal Mohamed

Oral Cancer Prediction Project: Final Report

1. Project Overview

Oral cancer refers to malignant growth that develops in any part of the mouth or oral cavity, including the lips, tongue, cheeks, floor of the mouth, and throat. It is a subtype of head and neck cancer and ranks among the most common cancers worldwide. The disease often begins as a painless lesion or patch, which can progress to severe pain, difficulty swallowing, and even death if untreated.

Key risk factors include:

- **Tobacco use** (smoking and chewing)
- **Excessive alcohol consumption**
- Prolonged exposure to **UV radiation**
- **Human papillomavirus (HPV) infection**
- Poor oral hygiene and chronic irritation

Early detection is crucial because the prognosis significantly improves when oral cancer is identified in its initial stages. However, many patients are diagnosed late due to the asymptomatic nature of early lesions or lack of access to timely medical care.

Oral cancer poses a growing threat to global health, especially in high-incidence regions such as India, Pakistan, Sri Lanka, and Taiwan. Early and accurate detection is crucial for improving survival rates and reducing the economic burden associated with late-stage diagnosis and treatment.

This dataset provides detailed information on individuals, including their demographics, lifestyle risk factors, clinical symptoms, cancer staging, and treatment details. The primary objective is to build a machine learning classification model that can accurately predict whether an individual has oral cancer based on these features

2. Project Importance:

This project aims to develop an effective machine learning-based tool to predict the likelihood of oral cancer from patient data, including demographic, lifestyle, and clinical symptom features. The significance of this work includes:

- Early Diagnosis:
  Automated prediction models can flag high-risk individuals early, allowing timely clinical examination and intervention, which can drastically improve survival rates.
- Healthcare Efficiency:
  Predictive analytics help prioritize patients for diagnostic tests, optimizing the use of medical resources and reducing unnecessary procedures.
- Public Health Impact:
  Identifying key risk factors through data analysis can inform targeted awareness campaigns and prevention strategies to reduce oral cancer incidence.
- Supporting Clinical Decisions:
  Integrating this predictive model into clinical workflows assists healthcare providers with evidence-based decision-making, especially in resource-limited settings.
- Extensibility to Multimodal Data:
  By extending the model to incorporate image analysis through neural networks, this project moves towards a comprehensive diagnostic system combining patient history with visual inspection, enhancing accuracy.

3. Project Objective:

Develop a classification model that predicts the presence of oral cancer (Oral Cancer (Diagnosis) = Yes/No) using features such as:

- Risk factors (e.g., tobacco, alcohol, HPV, betel quid use)
- Symptoms (e.g., oral lesions, difficulty swallowing)
- Clinical indicators (e.g., tumor size, cancer stage)
- Demographic and lifestyle factors

4. About Data Set:

This dataset is a structured compilation of real-world oral cancer cases collected from various regions, designed to support early detection and risk prediction of oral cancer. It encompasses a diverse set of features including demographic, behavioral, clinical, and economic attributes, providing a comprehensive view of patient health and lifestyle factors related to oral cancer risk.

- **Source**

The dataset is compiled from global health statistics and multiple epidemiological studies focusing on oral cancer. It aggregates real-world patient data from regions with high prevalence rates, capturing both clinical observations and demographic trends to provide a rich source for analysis.

- **Purpose**

The primary purpose of this dataset is to support the development of machine learning models aimed at **early detection and classification of oral cancer**. It facilitates research into identifying key risk factors, symptom patterns, and demographic influences, enabling healthcare professionals and researchers to better understand oral cancer progression and improve diagnostic accuracy.

- **Scope**

  - The dataset mainly covers **high-incidence countries** such as India, Pakistan, Sri Lanka, and Taiwan, where oral cancer cases are more prevalent due to lifestyle and environmental factors.
  - It also includes emerging cases from **Western countries**, reflecting the global spread and rising concern over oral cancer.
  - The dataset's comprehensive feature set allows exploration of regional disparities, lifestyle impacts, and socioeconomic conditions affecting oral cancer risk and outcomes.
  - Its scope supports cross-regional studies and development of generalized predictive models adaptable to diverse populations.

- **Target Variable**

  Oral Cancer (Diagnosis):

  Type: Categorical (Yes/No)

  Goal: Predict whether an individual is diagnosed with oral cancer.

Dataset Features Overview

The dataset comprises a comprehensive set of features that capture key factors associated with oral cancer. These features span demographic, behavioral, clinical, and economic dimensions, offering a rich foundation for predictive modeling and risk analysis.

Feature	Type	Description
ID	Categorical (Unique)	Unique identifier for each patient record.
Country	Categorical	Country of the individual, helping to identify regional trends.
Age	Numerical	Age of the individual in years.
Gender	Categorical	Biological sex of the individual (Male or Female).
Tobacco Use	Binary	Indicates history of tobacco consumption (Yes/No).
Alcohol Consumption	Binary	Indicates history of alcohol consumption (Yes/No).
HPV Infection	Binary	Indicates presence of human papillomavirus infection (Yes/No).
Betel Quid Use	Binary	Indicates use of betel quid, a known regional carcinogen.
Chronic Sun Exposure	Binary	Reflects long-term exposure to sunlight, especially for lip-related cancer.

Poor Oral Hygiene	Binary	General indicator of oral health and hygiene status.
Diet (Fruits & Vegetables Intake)	Ordinal (Low/Moderate/High)	Reflects diet quality based on intake of fruits and vegetables.
Family History of Cancer	Binary	Indicates if there is a genetic predisposition to cancer.
Compromised Immune System	Binary	Whether the individual has a weakened immune system.
Oral Lesions	Binary	Presence of visible oral lesions.
Unexplained Bleeding	Binary	Indicates bleeding without an apparent cause.
Difficulty Swallowing	Binary	Common symptom associated with more advanced cases.
White or Red Patches in Mouth	Binary	Visual indicators of pre-cancerous conditions.
Tumor Size (cm)	Numerical	Measured size of the tumor in centimeters.
Cancer Stage	Ordinal (0–4)	Clinical stage of the cancer (0 = No cancer).
Treatment Type	Categorical	Type of treatment received (e.g., Surgery, Radiation, Chemotherapy).
Survival Rate (5-Year, %)	Numerical	Percentage likelihood of 5-year survival post-diagnosis.
Cost of Treatment (USD)	Numerical	Estimated financial cost of treatment in U.S. dollars.
Economic Burden (Lost Workdays)	Numerical	Estimated annual workdays lost due to the condition.
Early Diagnosis	Binary	Indicates whether the cancer was detected at an early stage.
Oral Cancer (Diagnosis)	Binary (Target)	**Target variable** — whether the individual has been diagnosed with oral cancer (Yes/No).

Data Exploration

Initial analysis involved:

- Summary statistics and distribution plots for all features
- Class imbalance analysis for the target
- Correlation heatmap and pairwise comparisons
- Identification of region-specific trends

Insights:

- Class distribution was highly imbalanced toward non-cancer cases
- Higher cancer rates in individuals with tobacco, alcohol, and HPV history
- Visual patterns indicated strong relationships between symptoms and diagnosis

Data Preprocessing

We applied several preprocessing steps:

- Missing value imputation (mode/mean based)
- Label Encoding and One-Hot Encoding of categorical variables
- Standardization using Standard Scaler
- Outlier handling and normalization of skewed features
- PCA for dimensionality reduction while retaining 95% variance

We also identified the risk of data leakage and took steps to avoid it by isolating preprocessing steps and ensuring no target leakage from future information. For feature selection, we performed correlation analysis and used feature importance scores from tree-based models to retain only the most significant variables.

Challenges:

Addressing Class Imbalance

The dataset suffered from imbalance, with cancer stages cases. This posed a risk of biased model performance.

Solutions:

- SMOTE (Synthetic Minority Oversampling Technique): To synthetically generate new minority samples
- Class Weights: Adjusted training weights for balanced error cost
- Under sampling: Reduced majority class to balance the dataset

Feature Selection & Data Leakage Prevention

To reduce overfitting and improve model generalization:

- Correlation matrix used to drop redundant features
- Tree-based models used to extract feature importance
- Data leakage was mitigated by separating target-related fields and ensuring leakage-free pipelines

We discovered that some features (e.g., Treatment Type, Cancer Stage) might directly reflect diagnosis, so we treated them cautiously or removed them in early prediction models.

Insights and Business Implications

Top Risk Factors for Oral Cancer:

- Alcohol Consumption (0.48) and HPV Infection (0.44) have the strongest correlations with Risk_Score, even slightly stronger than Tobacco Use (0.39)—contrary to common assumptions.

- Betel Quid Use (0.44) is equally significant as HPV, highlighting its underrated role in oral cancer risk.

Diagnosis Drivers:

- Tumor Size (0.77) is the single strongest predictor of Oral Cancer Diagnosis, far outweighing other factors.
- Surprisingly, Symptom_Count (-0.01) shows almost no correlation with diagnosis, meaning the number of symptoms matters less than specific symptoms.

Symptom-Specific Insights:

- White/Red Patches (0.53) and Oral Lesions (0.51) correlate most strongly with Symptom_Count, making them key clinical warning signs.
- Difficulty Swallowing (0.49) and Unexplained Bleeding (0.45) also contribute significantly.

Unexpected Weak Links:

- Poor Oral Hygiene has near-zero correlation with everything (ranging from -0.00 to 0.01), suggesting it may be overestimated as a standalone risk factor.
- Gender, Age, or Immune Status (if included) don’t appear strongly linked—fraud detection-style demographics aren’t major players here.

Key Takeaway:

Oral cancer risk is driven by lifestyle factors (alcohol, HPV, betel quid) and tumor progression, while diagnosis relies heavily on tumor size and specific symptoms—not just symptom quantity.

Feature Importance Insights

An in-depth analysis using both **Random Forest feature importance** and **Fisher Score ranking** revealed the most influential predictors of oral cancer:

- **Tumor Size (cm)** emerged as the strongest predictor in both methods. Its dominance underscores the critical role of tumor progression in cancer diagnosis—patients with larger tumors are far more likely to test positive for oral cancer.
- **Symptom Count** ranked second in importance, particularly under the Fisher Score. This indicates that having multiple symptoms—regardless of their individual type—greatly increases the likelihood of a positive diagnosis.
- **Tobacco Use** and **HPV Infection** were consistently among the top behavioral and biological risk factors. This finding supports established medical literature, reinforcing their known link to oral cancer development.
- Symptoms such as **Difficulty Swallowing**, **Oral Lesions**, and **Unexplained Bleeding** also showed strong diagnostic relevance. Though their contribution in the Random Forest model was moderate, their high Fisher Scores suggest they're critical for early screening.
- Conversely, features like **Betel Quid Use**, **Alcohol Consumption**, and **Compromised Immune System** had relatively lower importance scores. This could point to weaker correlations in this specific dataset or limited sample representation.

Key Takeaway:
Focusing on **tumor size**, **overall symptom burden**, and high-risk behavioral/biological factors like **tobacco use** and **HPV infection** can significantly enhance the predictive accuracy of oral cancer models. These insights also guide clinical priorities, emphasizing the importance of early symptom recognition and lifestyle risk monitoring.

Model Development and Evaluation

In this phase, we developed and evaluated several machine learning models to predict the presence of **oral cancer** based on clinical, behavioral, and symptomatic features. The goal was to build a robust, generalizable model that accurately classifies patients into "Yes" (cancer) or "No" (no cancer) classes.

1. Data Preparation for Modeling

Before applying any machine learning algorithm, we:

- **Preprocessed the data**: including handling missing values, encoding categorical features, and normalization where needed.
- **Split the dataset**: into training and test sets (commonly 80/20 or stratified to ensure class balance).
- **Addressed imbalance** (if needed): using techniques like class_weight, SMOTE, or adjusting scale_pos_weight in models like XGBoost.

2. Models Applied

We experimented with a variety of **supervised classification models** to explore performance and generalizability:

Tree-Based Models

- **Decision Tree**: Simple, interpretable model that performed with perfect accuracy, though may overfit.
- **Random Forest**: Ensemble of decision trees; provided high accuracy and feature importance insights.
- **AdaBoost**: Combines weak learners to build a strong classifier. Delivered perfect metrics in this dataset.
- **LightGBM**: Gradient boosting framework that is fast and efficient; achieved high performance.

Probabilistic Models

- **Gaussian Naive Bayes**: Assumes independence between features.

- **Quadratic Discriminant Analysis (QDA)**: Models class-specific variance. Perfect accuracy but potential overfitting.

Distance-Based Model

- **K-Nearest Neighbors (KNN)**: Simple yet powerful; performed nearly perfect with high recall and precision.

Linear and Margin-Based Models

- **Logistic Regression**: Interpretable and reliable; achieved high precision and recall.
- **Support Vector Machine (SVM - RBF Kernel)**: Powerful for non-linear decision boundaries; excellent performance.
- **Stochastic Gradient Descent (SGD) Classifier**: Fast, scalable linear model suitable for large datasets.

Advanced Models

- **XGBoost**: Regularized gradient boosting model. After tuning (scale_pos_weight), it delivered near-perfect results.
- **Artificial Neural Network (ANN)**: We also applied a basic neural network model using Keras, which gave good results after tuning hidden layers and epochs.

3. Evaluation Metrics

We used four key metrics for evaluation:

- **Accuracy**: Overall correctness of the model.
- **Precision**: How many predicted positives were actually correct.
- **Recall**: How many actual positives were detected (important in medical diagnosis).
- **F1 Score**: Harmonic mean of precision and recall, balancing both metrics.

All models were evaluated on the **same test set** to ensure fair comparison.

4. CNN and Transfer Learning (for Image-Based Models)

To expand our exploration, we also incorporated **image data** related to oral cancer (like lesion or tissue images from online datasets). For this, we:

- Applied **Convolutional Neural Networks (CNNs)** to capture spatial patterns in medical images.
- Used **Transfer Learning** techniques with pre-trained models (e.g., VGG16, ResNet50,etc) to reduce training time and boost accuracy on limited image data.

These deep learning models showed promising results and opened a path for combining **symptom-based prediction + image-based detection**.

5. Feature Importance

Using **Random Forest** and **Fisher Score**, we analyzed the impact of each feature:

- **Tumor Size (cm)**: Most critical feature.
- **Symptom Count**: Strong indicator of cancer presence.
- **HPV Infection, Tobacco Use**: Key behavioral and biological risk factors.
- **Difficulty Swallowing, Oral Lesions**: High Fisher Score relevance.

These insights helped guide model tuning and clinical interpretation.

5. Final Observations

Model	Comments
Gaussian Naive Bayes	Perfect scorer; verify on external datasets for generalization.
K-Nearest Neighbors	Near-perfect, very stable and interpretable.
XGBoost	Best trade-off between precision and recall after tuning.
SVM / SGD / Logistic	Strong linear baselines; useful for quick and interpretable results.

Model Performance Summary

Model	Accuracy	Precision	Recall	F1 Score
SVM (RBF Kernel)	0.985070	1.0	0.970317	0.984935
K-Nearest Neighbors	0.998842	1.0	0.997698	0.998848
Gaussian Naive Bayes	1.000000	1.0	1.000000	1.000000
AdaBoost	1.000000	1.0	1.000000	1.000000
SGD Classifier	0.989336	1.0	0.978798	0.989285
Quadratic Discriminant Analysis	1.000000	1.0	1.000000	1.000000

Project Deployment

To operationalize the trained models and make them accessible to users, the project was deployed using MLflow and Flask—two powerful tools in the machine learning deployment pipeline.

- MLflow was used to track experiments, manage different model versions, and package models for deployment. With its mlflow.sklearn.log_model() and model registry functionality, each model's metrics and artifacts were logged and tracked. This enabled easy comparison and rollback between different model versions during development.
- The selected model was then served via a Flask API. Flask provided a lightweight web framework to create REST endpoints for prediction. The Flask app loads the saved model (logged by MLflow) and exposes an /analyze endpoint that accepts input data (e.g., medical image features or preprocessed data) and returns the predicted oral cancer diagnosis.

- For production readiness:
    - CORS and error handling were integrated into the API.
    - Input validation was performed to ensure robust interaction.
    - The API was tested using tools like Postman and cURL.

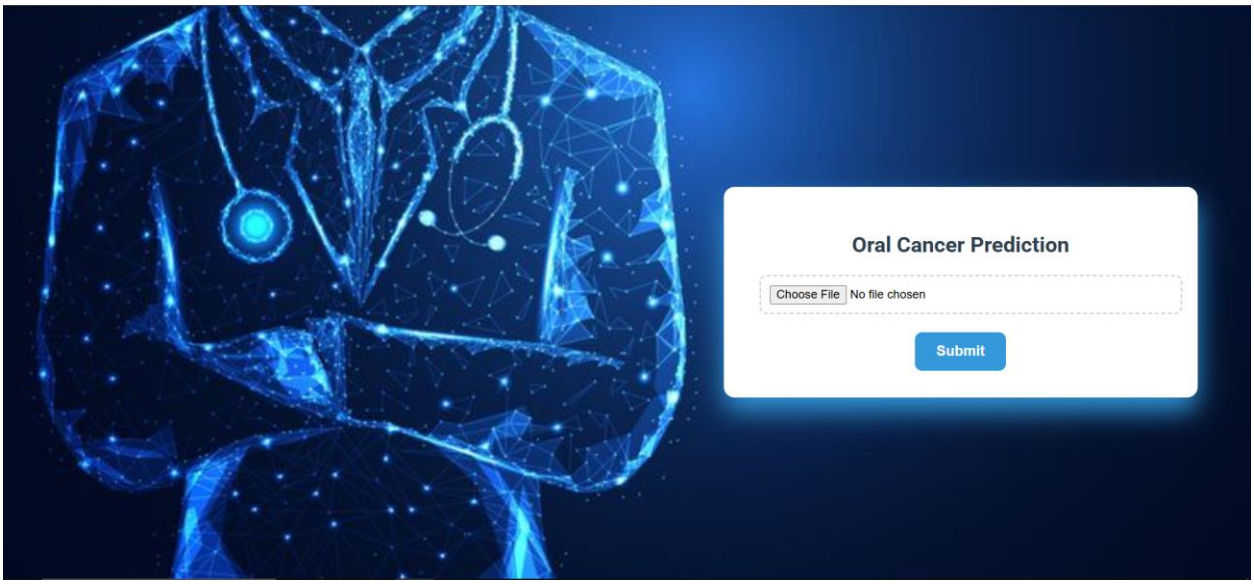
This architecture allows seamless integration with front-end or hospital information systems and can be containerized using Docker for broader deployment (e.g., on AWS, Azure, or local hospital servers).

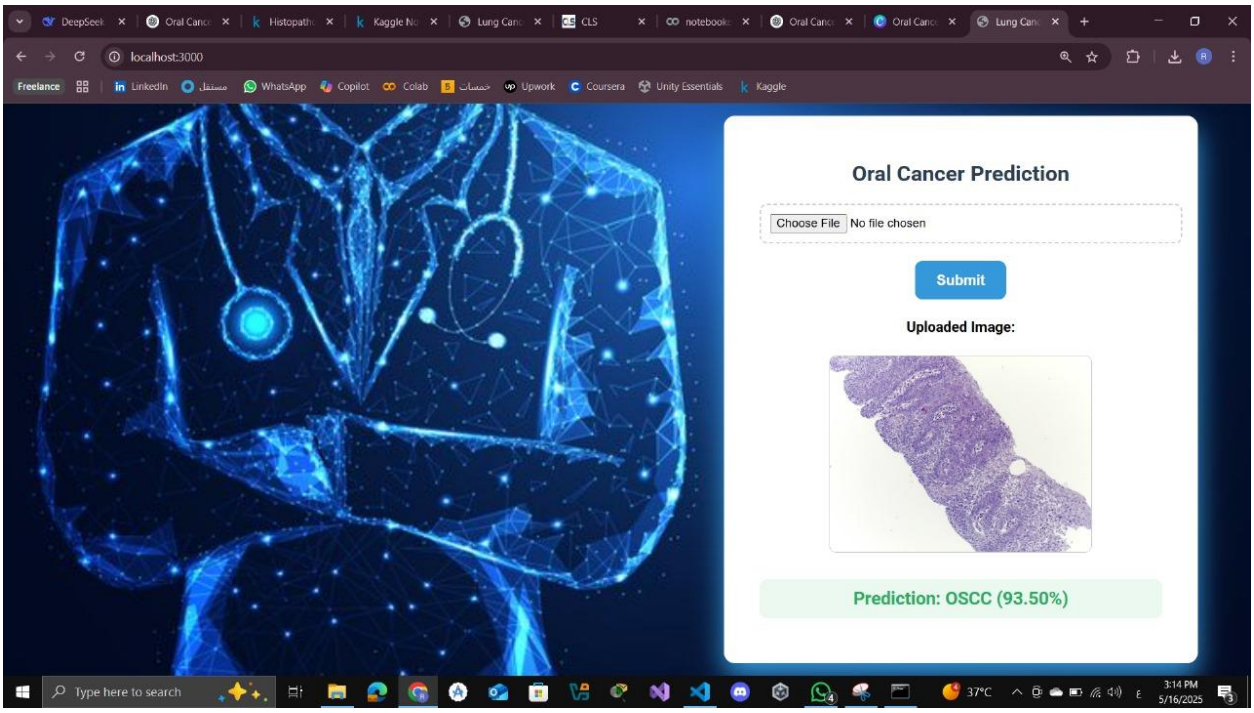
Include a diagram showing how different components interact:

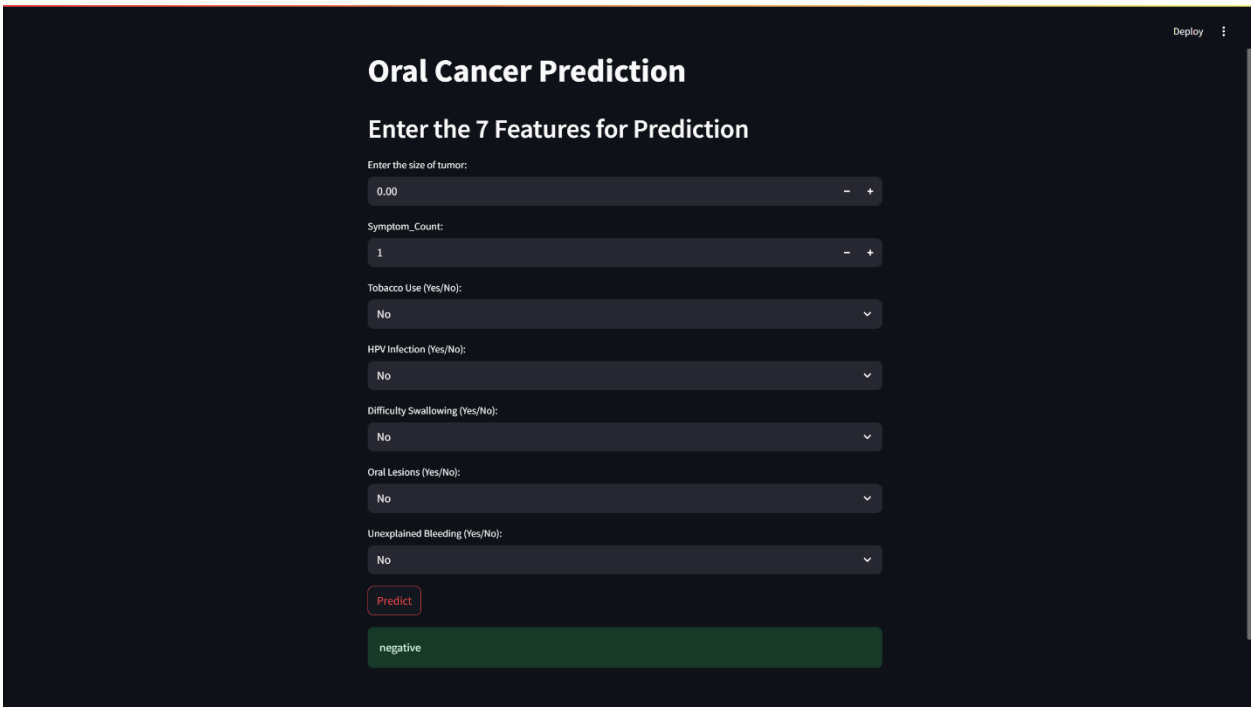
- User (Doctor) → Frontend UI (optional)
- Frontend → Flask API
- Flask API → ML Model (via MLflow)
- Model returns results → API → Frontend/User

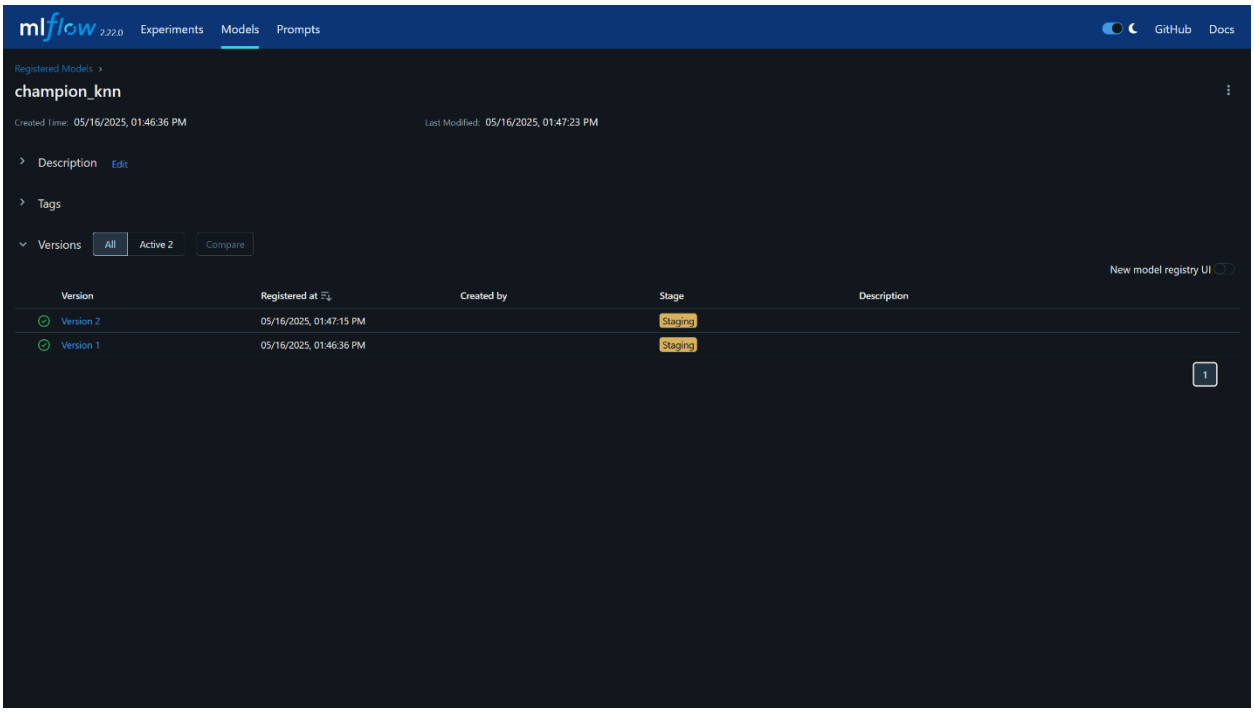
This visually clarifies the deployment flow.

And this the deployment Style with images data:






Deploy
Oral Cancer Prediction
Enter the 7 Features for Prediction
Enter the size of tumor:
0.00
Symptom_Count:
1
Tobacco Use (Yes/No):
No
HPV Infection (Yes/No):
No
Difficulty Swallowing (Yes/No):
No
Oral Lesions (Yes/No):
No
Unexplained Bleeding (Yes/No):
No
Predict
negative

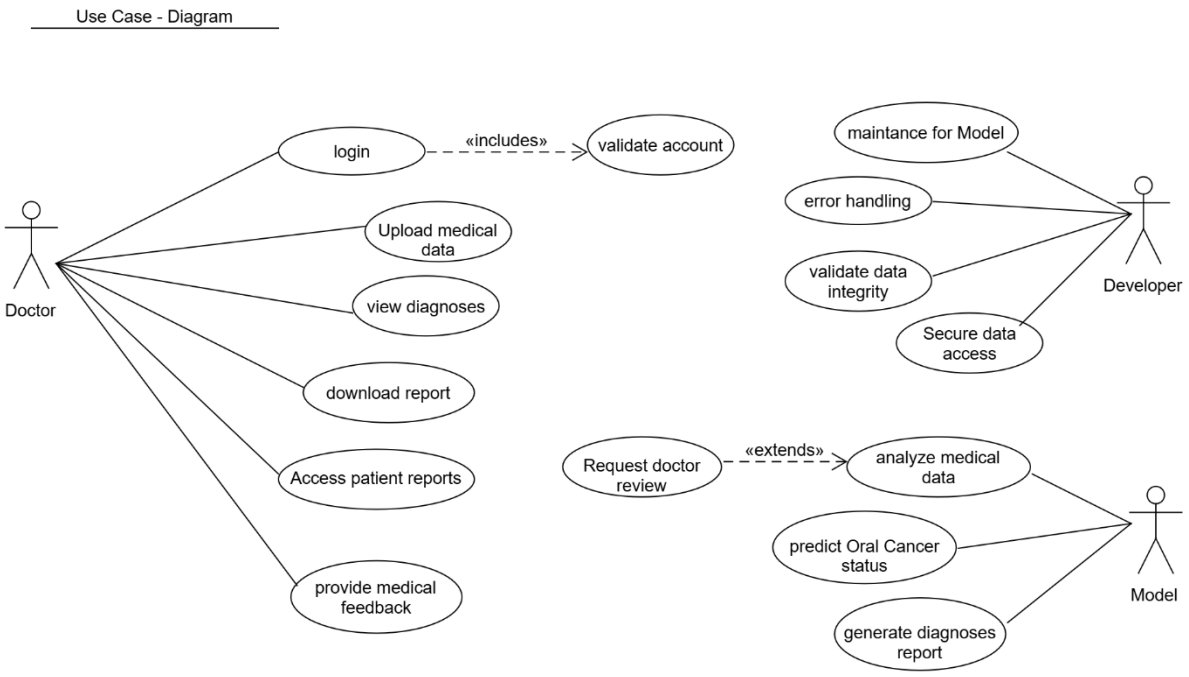

mlflow 2.22.0
Experiments
Models
Prompts
GitHub
Docs
Registered Models
champion_knn
Created Time: 05/16/2025, 01:46:36 PM
Last Modified: 05/16/2025, 01:47:23 PM
Description
Edit
Tags
Versions
All
Active 2
Compare
New model registry UI
Version
Registered at
Created by
Stage
Description
Version 2
05/16/2025, 01:47:15 PM
Staging
Version 1
05/16/2025, 01:46:36 PM
Staging
1

Software Engineering Diagrams

As part of our system analysis and design phase, we developed several Software Engineering Diagrams to visualize and understand the structure and functionality of our project.

1. Use Case Diagram

We created a detailed Use Case Diagram , which highlights the different interactions between the system's actors and its functionalities.



In this diagram, three main actors are involved:

- **Doctor**: Performs critical functions such as logging in, uploading medical data, viewing diagnoses, downloading reports, accessing patient records, and providing medical feedback. The "login" use case includes account validation to ensure secure access, while the "analyze medical data" use case can optionally extend to "request doctor review" for collaborative diagnosis.
- **Developer**: Ensures the system's reliability and robustness through use cases like model maintenance, error handling, validating data integrity, and securing data access.
- **Model (AI System)**: Represents the machine learning model, which is responsible for analyzing the uploaded medical data, predicting oral cancer status, and generating diagnosis reports.

This diagram helps to clearly define the **functional requirements** and **responsibilities** of each actor in the system. It also aids in communication among stakeholders and guides the development process by laying out the system's core functionalities and dependencies.

Use Cases

1.1 Login:

Use Case ID	1
Use Case Name	Login
Actors	Doctor
Preconditions	- The doctor is registered in the system. - The doctor has valid login credentials.
Normal flow	- The doctor navigates to the login page. - The doctor enters their username and password. - The system verifies the credentials. - If valid, the doctor is granted access and redirected to the system.
Postconditions	- If login is successful, the doctor gains access. - If login fails, access is denied, and the doctor may need to reset credentials.
Alternative Flow	If credentials are incorrect: • The system displays an error message: "Invalid username or password." • The doctor gets three attempts before temporary account lockout. - If locked out, the doctor must reset the password via email verification.
Exceptions	- Database Connection Failure: If the system cannot connect to the database, it

	shows "Login service unavailable. Please try again later." - Session Timeout: If the doctor does not log in within a specified time, the session expires, and they must re-enter credentials.

1.2 Upload medical data

Use Case ID	2
Use Case Name	Upload medical data
Actors	Doctor
Preconditions	- The doctor must be logged in. - The patient's data must be in an acceptable format (CSV, JSON).
Normal flow	- The doctor selects "Upload Medical Data." - The system provides options to: • Upload a file (CSV, JSON). • Manually enter data. - The doctor uploads the file. - The system validates the data format and stores it.
Postconditions	- If successful, the system stores the data. - If failed, no data is stored, and the doctor must try again.
Alternative Flow	If the uploaded file is not formatted correctly: • The system rejects the file. • Displays: "Invalid file format. Please upload a valid CSV or JSON file."

	The doctor must correct and re-upload
Exceptions	- **File Too Large:** If the uploaded file exceeds the size limit, the system rejects it with an error message. - **Network Interruption:** If the network disconnects during upload, the doctor must retry.

1.3 View diagnoses

Use Case ID	3
Use Case Name	View diagnoses
Actors	Doctor
Preconditions	- The system has stored diagnosis reports.
Normal flow	- The doctor selects "View Diagnoses." - The system retrieves and displays diagnosis reports.
Postconditions	- If reports exist, they are displayed. - If reports are missing, the doctor is notified.
Alternative Flow	- if no reports exist, the system displays: • "No diagnoses available for this patient."
Exceptions	- **Database Failure:** If the system cannot retrieve reports due to a database error, an error message is displayed. - **Permission Issues:** If the doctor does not have permission to view certain reports, access is denied.

1.4 Download Report

Use Case ID	4
Use Case Name	Download report
Actors	Doctor
Preconditions	- A diagnosis report must be available.
Normal flow	- The doctor selects a patient's report. - The system generates a downloadable PDF file. - The doctor downloads the report.
Postconditions	- If successful, the doctor has a local copy of the report. - If failed, no download occurs, and an error is displayed.
Alternative Flow	if the report is missing or corrupted: • The system displays: "Report unavailable. Please try again later."
Exceptions	- **Insufficient Storage:** If the device has no storage space, the download fails. - **File Corruption:** If the report file is corrupted, it cannot be downloaded.

1.5 Provide Medical Feedback

Use Case ID	5
Use Case Name	Download report
Actors	Doctor
Preconditions	- The doctor has reviewed the diagnosis.
Normal flow	- The doctor selects "Provide Feedback." - The system prompts for comments. - The doctor submits feedback. - The system stores the feedback.
Postconditions	- If feedback is given, it is stored. - If feedback is missing, the system prompts for input.
Alternative Flow	If the doctor leaves the feedback blank: • The system displays: "**Feedback cannot be empty.**" • The doctor must enter a comment.
Exceptions	- **System Crash:** If the system crashes before saving, feedback is lost. - **Database Error:** If the system fails to save feedback, an error is displayed.

1.6 Predict Oral Cancer Status

Use Case ID	6
Use Case Name	Predict oral cancer status
Actors	AI model
Preconditions	- The system must have patient medical data.
Normal flow	- The AI model processes the patient data. - It extracts key features. - The model predicts the cancer probability score. - The system displays results.
Postconditions	- If successful, the system presents the prediction. - If failed, the doctor must input more complete data..
Alternative Flow	If patient data is incomplete: • The system displays: "Error: Insufficient data for analysis."
Exceptions	- **Model Crash:** If the AI model fails, the system returns an error message. - **Unexpected Output:** If the model generates unexpected results, an alert is raised

1.7 Analyze Medical Data

Use Case ID	7
Use Case Name	Analyze medical data
Actors	AI model
Preconditions	- The system has patient medical records. - The AI model is trained on medical datasets.
Normal flow	- The doctor uploads patient medical data. - The AI model scans and analyzes the data. - The model identifies potential indicators of oral cancer. - The results are stored and made available for diagnosis.
Postconditions	- If successful, the system provides a structured analysis. - If unsuccessful, the doctor must manually analyze the data.
Alternative Flow	If the input data is incomplete, the model requests additional information.
Exceptions	- **Corrupted Data:** The input file is unreadable. - **Low Accuracy:** The model lacks confidence in its predictions.

1.8 Generate Diagnosis Report

Use Case ID	8
Use Case Name	Generate Diagnosis Report
Actors	AI model
Preconditions	The system has successfully analyzed patient data.
Normal flow	- The AI model compiles its findings. - A structured report is generated with key insights. - The doctor can review and download the report.
Postconditions	- If successful, the doctor receives a well-organized medical report. - If unsuccessful, manual report generation is needed.
Alternative Flow	If additional details are required, the doctor can request further analysis.
Exceptions	- **Report Formatting Error:** The system fails to generate a readable document. - **Missing Data:** Some required information is absent from the report.

1.9 Maintain Model

Use Case ID	9
Use Case Name	Maintain model
Actors	Developer
Preconditions	- The system has an existing AI model. - The developer has administrative access to modify the model.
Normal flow	- The developer navigates to the model maintenance section. - The developer uploads new training data if necessary. - The model undergoes retraining. - The system validates the model's new accuracy. - The updated model is deployed.
Postconditions	- If successful, the updated AI model is deployed. - If unsuccessful, the previous model remains in use.
Alternative Flow	If the new model performs worse than the previous version: • The system logs the issue. • The developer receives an alert to review the training process.
Exceptions	- **Data Format Error:** Training data is corrupted or incorrectly formatted.

	- **Model Training Failure:** The AI model fails to converge or reaches an invalid state. - **Deployment Failure:** The system cannot replace the old model due to server issues.

1.10 Error Handling

Use Case ID	10
Use Case Name	Error handling
Actors	Developer
Preconditions	- An error occurs during system operation. - The developer has system access to debug and fix issues.
Normal flow	- The system logs an error related to the AI model. - The developer reviews error logs. - The developer identifies the root cause of the issue. - The developer applies a fix. - The developer tests the fix to ensure resolution. - The system is updated with the corrected code.
Postconditions	- If successful, the system runs without errors. - If unsuccessful, the system may still have bugs, requiring further debugging.
Alternative Flow	If the error cannot be fixed immediately, a temporary solution is applied.
Exceptions	- **Unidentified Issue:** The developer is unable to diagnose the error. - **Fix Causes Additional Issues:** The implemented fix leads to other system failures.

1.11 Validate Data Integrity

Use Case ID	11
Use Case Name	Validate data integrity
Actors	Developer
Preconditions	- The system has patient data stored in a database. - The developer has access to check data integrity.
Normal flow	- The developer initiates a data validation check. - The system scans for missing, duplicate, or corrupted data. - If issues are found, the developer is alerted. - The developer cleans and corrects the data. - The system updates its records.
Postconditions	- If successful, the database remains accurate and reliable. - If unsuccessful, medical reports may contain inconsistencies.
Alternative Flow	If the system has an auto-correction mechanism, it attempts to fix minor issues automatically.
Exceptions	- **Data Loss:** Critical medical data is missing and cannot be recovered. - **Unauthorized Modification:** Data has been tampered with by an unauthorized user.

1.12 Secure Data Access

Use Case ID	12
Use Case Name	Secure Data Access
Actors	Developer
Preconditions	- The system stores sensitive patient data. - The developer has security clearance.
Normal flow	- The developer reviews current security protocols. - The developer updates encryption and authentication mechanisms. - The system applies security updates. - The developer tests for vulnerabilities. - The system logs security enhancements.
Postconditions	- If successful, patient data remains secure. - If unsuccessful, security vulnerabilities persist.
Alternative Flow	If an unauthorized access attempt is detected, the system alerts the developer and locks the account.
Exceptions	- **Data Breach:** An external attack bypasses security measures. - **System Downtime:** Security updates cause temporary unavailability.

2. Class Diagram

The Class Diagram represents the static structure of the system and provides a blueprint for how the system's components (classes) interact with each other. It defines the system's objects, their attributes, operations (methods), and the relationships among them. Below is a detailed description of each class used in our oral cancer detection system.

Description of Classes and Their Roles:

- **Doctor**
  This class represents the medical professionals who interact with the system. Doctors can log in, upload medical data such as images, view diagnoses, download patient reports, access historical patient information, and provide medical feedback. Their role is crucial for both data input and validating the system's outputs.

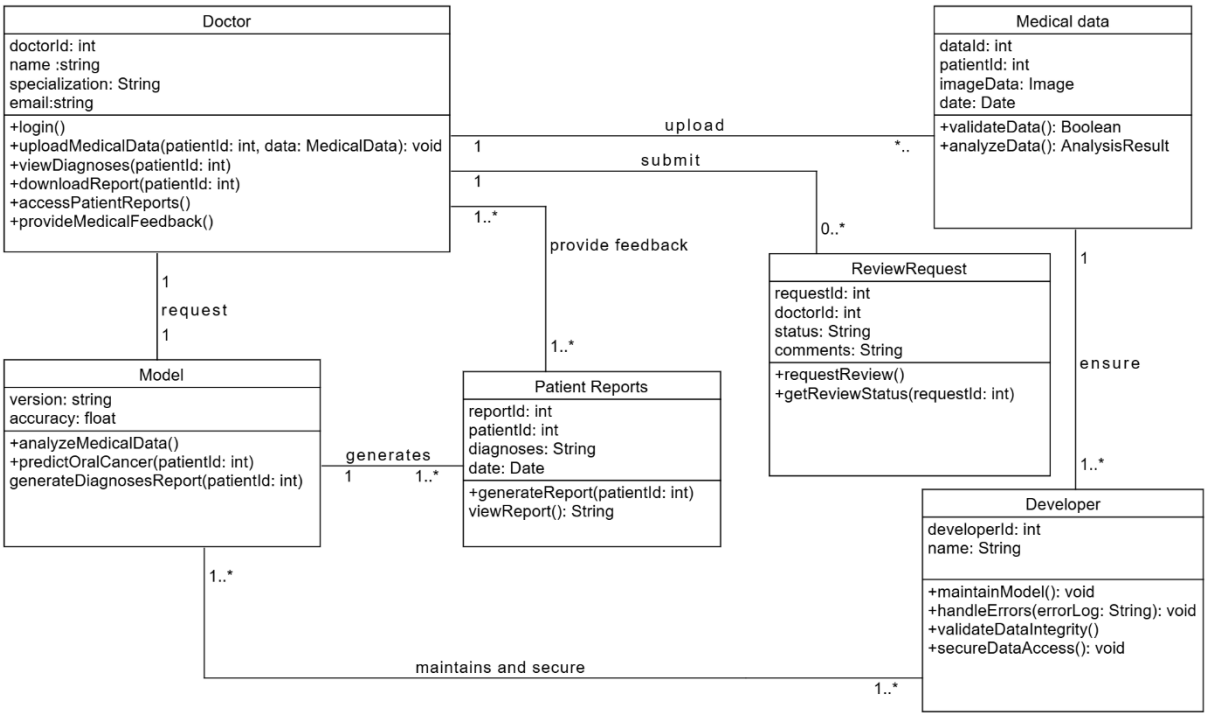
- **Medical Data**
  Medical Data encapsulates the details of the patient's medical records, primarily image data and relevant metadata like patient ID and date of record. This class includes methods to validate the data and initiate analysis, ensuring that only high-quality, accurate data feeds into the model.

- **Model**
  The Model class represents the machine learning component responsible for analyzing medical data and predicting oral cancer status. It supports versioning and tracks accuracy. It performs core functions such as analyzing input data, making predictions, and generating diagnostic reports.

- **Patient Reports**
  This class stores and manages diagnosis reports generated from model predictions and doctor feedback. It includes functionality to generate and view reports, allowing doctors to track patient history and review previous diagnostics.

- **Review Request**
  Review Request manages requests for additional review or second opinions by doctors. It tracks the status of these requests and any associated comments, facilitating collaboration and thorough diagnosis.

- **Developer**
  The Developer class ensures the system's technical robustness. Developers maintain the machine learning models, handle error logging, validate data integrity, and implement security measures for data access. This role supports the ongoing reliability and security of the system.



3- Sequence Diagram

The sequence diagram illustrates a structured workflow where a Developer interacts with a System, Database, and Security Module to manage errors, update models, enforce security, and maintain data integrity. Below is a detailed breakdown of the process

1. System Error Handling

Objective: Monitor health system, diagnose issues, and resolve errors.

- **Request Logs:** The developer retrieves performance logs from the system to analyze potential issues.
- **Error Resolution:** After identifying problems (e.g., crashes or slowdowns), the developer implements fixes, and the system confirms successful resolution.

Why It Matters: Proactive error handling minimizes downtime and ensures smooth system operation.

---

2. Data and Model Updates

Objective: Keep machine learning models up-to-date with new data.

- **Upload Data/Model:** The developer submits new training data or an improved model version, which the database stores.
- **Training Status Check:** The developer verifies the training progress, ensuring the model updates correctly.

Why It Matters: Regular updates improve model accuracy and adapt to new trends.

---

3. Security Operations

Objective: Maintain system security and prevent vulnerabilities.

- **Update Security Settings:** The developer modifies security configurations (e.g., access controls or encryption).
- **Confirmation & Application:** The Security Module requests confirmation before enforcing changes, ensuring no unintended disruptions.
- **Database Sync:** Approved security updates are saved in the database for consistency.

Why It Matters: Prevents breaches and ensures compliance with security policies.

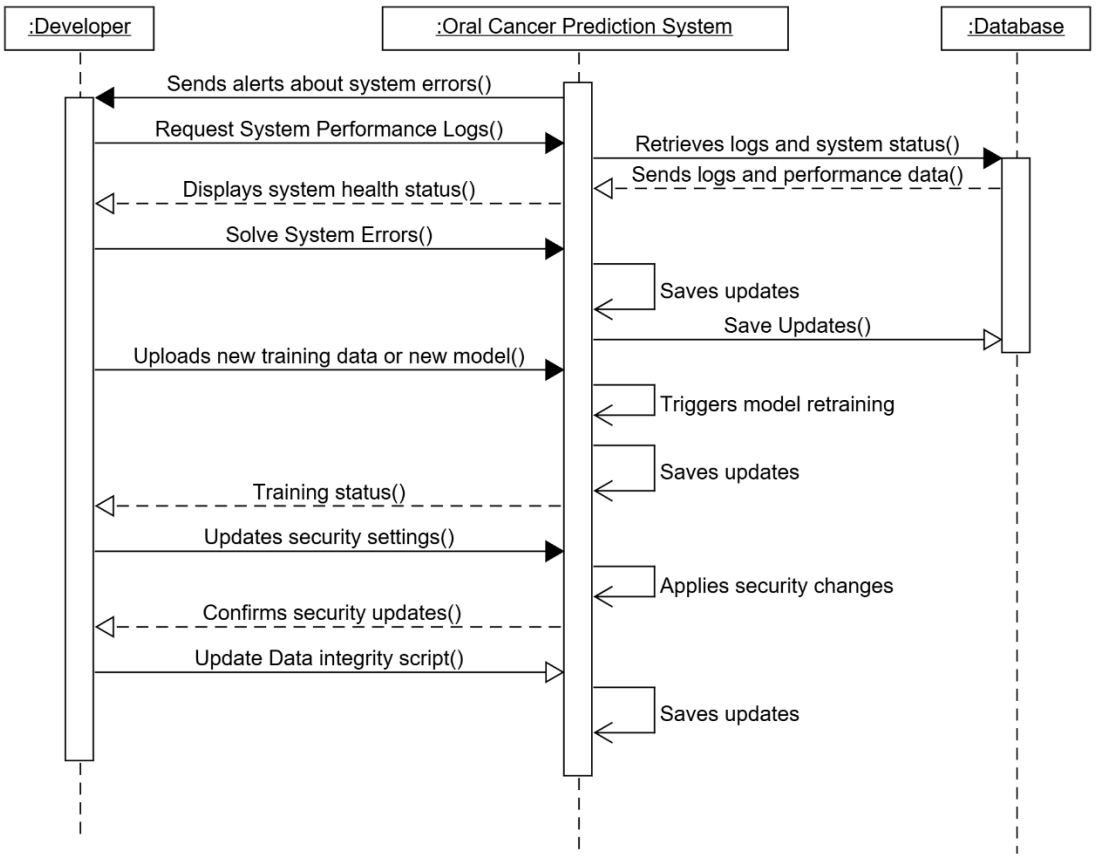
---

4. Database Interactions

Objective: Facilitate logging, retraining, and system updates.

- **Retrieve Logs/Status:** The developer pulls logs for audits or debugging.
- **Trigger Retraining:** When new data is available, the developer initiates model retraining via the database.

Why It Matters: Ensures traceability (logs) and automates model improvement (retraining).



And this for prediction :

This diagram outlines the process of an **Oral Cancer Prediction System** used by doctors, covering **user authentication, data processing, AI prediction, reporting, and model improvement**. Below is a step-by-step breakdown:

---

1. User Authentication

- **Login:** The doctor logs in with credentials (user, PW).
- **Verification:** The system checks the user's validity and grants access.
- **Confirmation:** A "Successfully" logged-in status and "OK" check response are returned.

Why It Matters: Ensures only authorized medical personnel access sensitive patient data.

2. Data Upload & Processing

- **Upload Medical Data:** The doctor submits patient data (linked to PatientID).

- **Storage & Cleaning:**
    - Raw data is stored ("Stored Successfully").
    - The system cleans the data (e.g., removing inconsistencies) and returns a sanitized dataset.
- **Prediction:** The AI model analyzes the data and returns an **oral cancer prediction**.
- **Result Storage:** The prediction is saved with the PatientID for future reference.

Why It Matters: Automated cleaning and prediction reduce manual effort and improve diagnostic speed.

3. Notification & Reporting

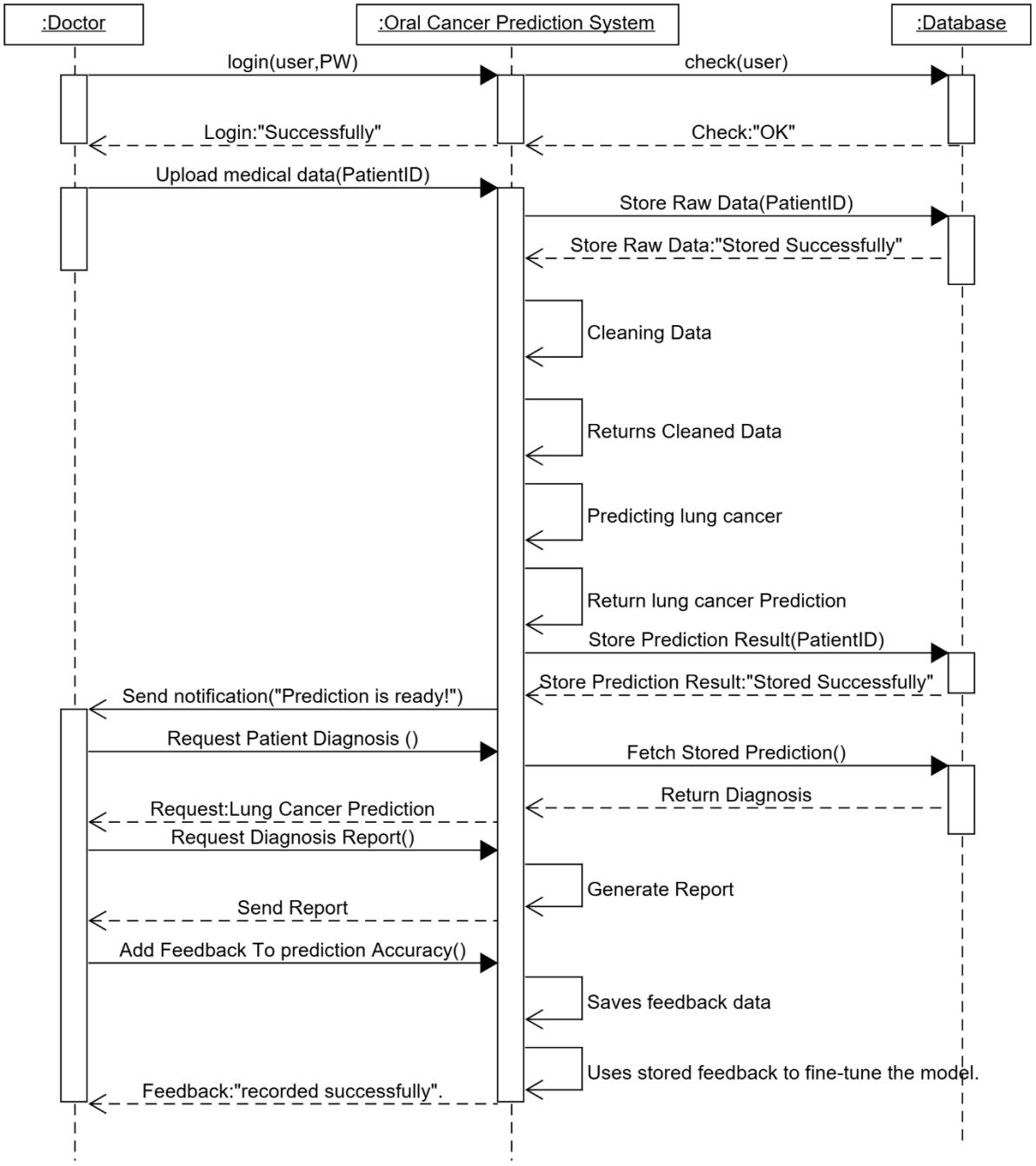
- **Alert:** The system notifies the doctor ("Prediction is ready!").
- **Report Generation:**
    - The doctor requests a diagnosis report.
    - The system fetches the prediction, generates a detailed report, and sends it.
- **Feedback Loop:**
    - The doctor provides feedback on prediction accuracy (e.g., correct/incorrect).
    - Feedback is recorded ("recorded successfully") and stored for model retraining.

Why It Matters: Reports aid clinical decisions, while feedback enhances the AI's accuracy over time.

4. Model Improvement

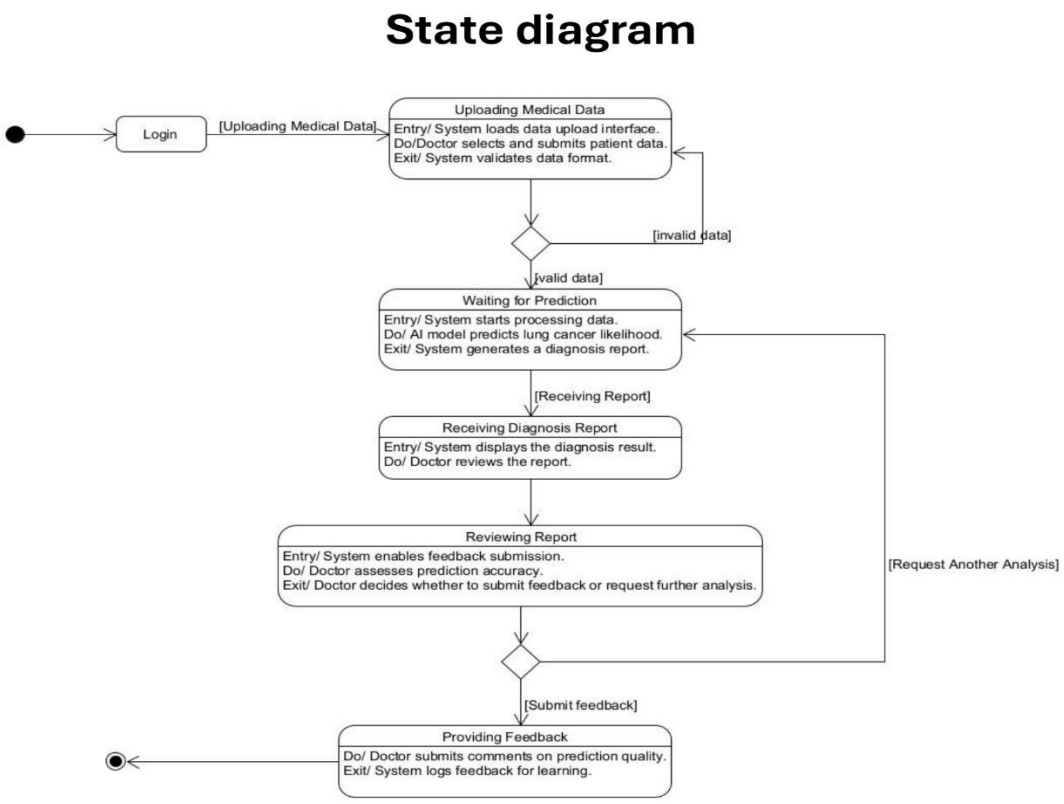
- **Feedback Utilization:** Stored feedback is used to **fine-tune the prediction model**.
- **Continuous Learning:** The system adapts to new patterns, improving future predictions.

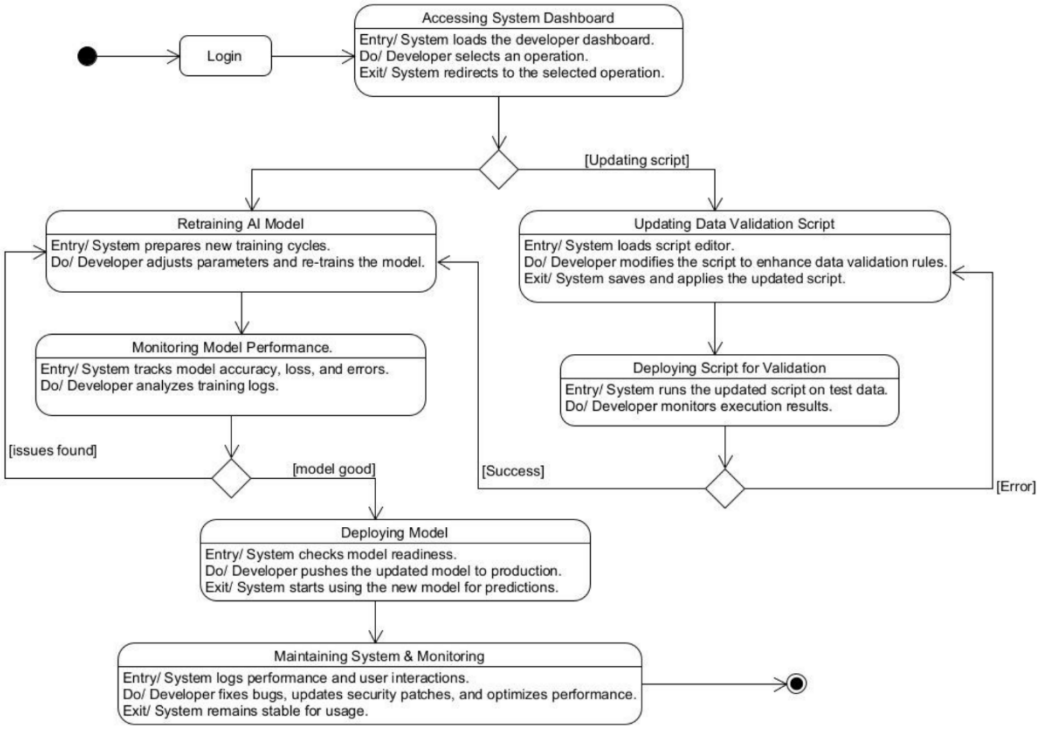
Why It Matters: Ensures the AI evolves with real-world data, reducing diagnostic errors.


:Doctor
:Oral Cancer Prediction System
:Database
login(user,PW)
check(user)
Login:"Successfully"
Check:"OK"
Upload medical data(PatientID)
Store Raw Data(PatientID)
Store Raw Data:"Stored Successfully"
Cleaning Data
Returns Cleaned Data
Predicting lung cancer
Return lung cancer Prediction
Store Prediction Result(PatientID)
Store Prediction Result:"Stored Successfully"
Send notification("Prediction is ready!")
Request Patient Diagnosis ()
Fetch Stored Prediction()
Return Diagnosis
Request:Lung Cancer Prediction
Request Diagnosis Report()
Generate Report
Send Report
Add Feedback To prediction Accuracy()
Saves feedback data
Uses stored feedback to fine-tune the model.
Feedback:"recorded successfully".

4. State Diagram

This state diagram describes the workflow of a medical AI system where doctors upload patient data, receive cancer predictions, and provide feedback. Below is a detailed breakdown of the states and transitions:




Login
Accessing System Dashboard
Entry/ System loads the developer dashboard.
Do/ Developer selects an operation.
Exit/ System redirects to the selected operation.
[Updating script]
Retraining AI Model
Entry/ System prepares new training cycles.
Do/ Developer adjusts parameters and re-trains the model.
Monitoring Model Performance.
Entry/ System tracks model accuracy, loss, and errors.
Do/ Developer analyzes training logs.
[issues found]
[model good]
Updating Data Validation Script
Entry/ System loads script editor.
Do/ Developer modifies the script to enhance data validation rules.
Exit/ System saves and applies the updated script.
Deploying Script for Validation
Entry/ System runs the updated script on test data.
Do/ Developer monitors execution results.
[Success]
[Error]
Deploying Model
Entry/ System checks model readiness.
Do/ Developer pushes the updated model to production.
Exit/ System starts using the new model for predictions.
Maintaining System & Monitoring
Entry/ System logs performance and user interactions.
Do/ Developer fixes bugs, updates security patches, and optimizes performance.
Exit/ System remains stable for usage.

5.Activity Diagram

An Activity Diagram is a behavioral diagram in Unified Modeling Language (UML) that represents the flow of activities, actions, and processes in a system. It is similar to a flowchart but provides more advanced features like parallel processing, swim lanes, and object flows. Activity diagrams are widely used in software engineering, business process modeling, and system design.

This **activity diagram** illustrates a **machine learning (ML)-assisted medical workflow** where a doctor inputs patient data to assess oral cancer risk. The diagram captures the **step-by-step process**, including **data validation, prediction generation, error handling, and treatment recommendations**.

1. Key Components of the Diagram

A. Input Phase

- **Doctor Inputs Patient Data**
  - Includes **age, smoking history, genetic factors, and symptoms**.
- **Validate Input Data**
  - Checks if data is complete and correctly formatted.
  - **Decision Node**:
    - **Yes** → Proceed to preprocessing.
    - **No** → Display error and prompt re-entry.

B. Preprocessing & Prediction

- **Preprocess Data**
  - Normalization, encoding (for ML model compatibility).
- **Load Trained ML Model**
  - A pre-trained model evaluates oral cancer risk.
- **Generate & Display Prediction**
  - Results are shown to the doctor.

C. Error Handling

- If data is invalid:
  - **Display Error Message** → Prompt re-entry.

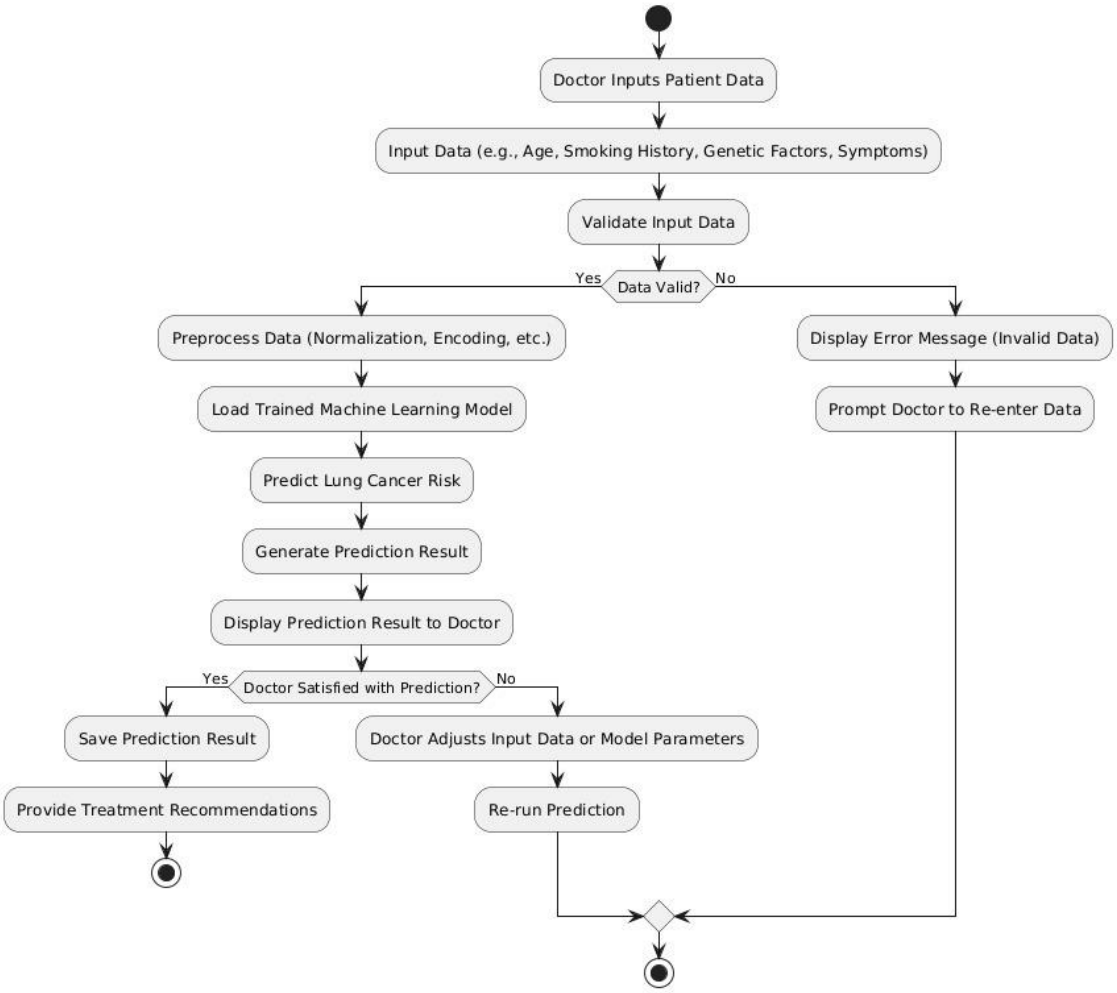
D. Feedback Loop

- **Doctor Reviews Prediction**

- **Satisfied?** → Save result and proceed.
- **Unsatisfied?** → Adjust data/model → Re-run prediction.

E. Final Output

- **Provide Treatment Recommendations**
  - Based on the validated prediction.



And for this :

1. Key Stages of the Workflow

A. Data Preparation Phase

1. **Collect Dataset**

- Input: Patient records, medical history (structured/unstructured data).

2. **Preprocess Data**
    - Steps: Cleaning (handling missing values), normalization, encoding (e.g., categorical → numerical).

B. Model Development Phase

3. **Split Data**
    - Training set (model learning) vs. testing set (evaluation).
4. **Train ML Model**
    - Algorithm selection (e.g., Random Forest, Neural Networks).
5. **Evaluate Performance**
    - Metrics: Accuracy, precision, recall, F1-score.

C. Decision Point: Model Acceptance

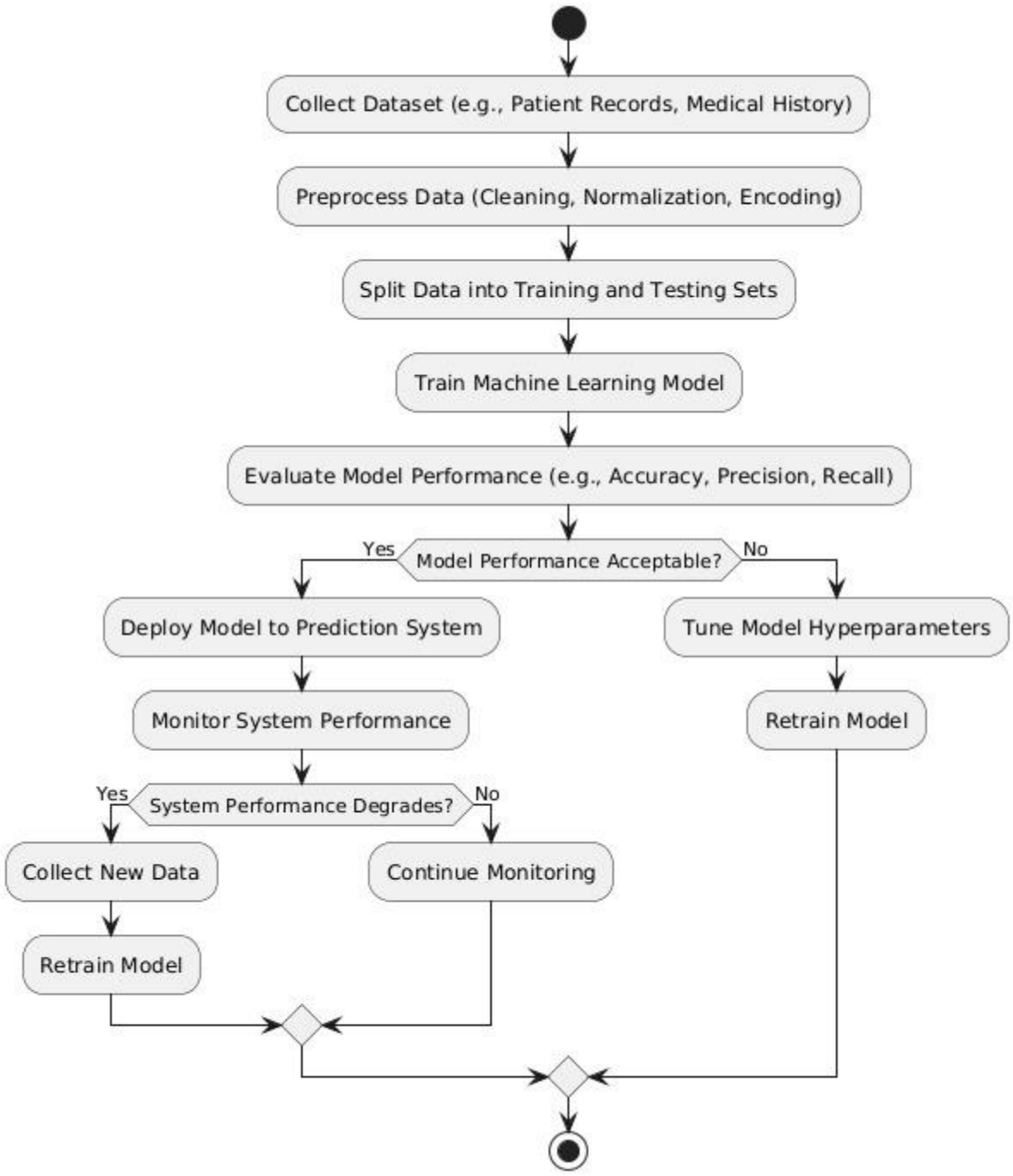
6. "**Model Performance Acceptable?**"
    - **Yes** → Proceed to deployment.
    - **No** → Tune hyperparameters (e.g., learning rate, epochs) and retrain.

D. Deployment & Monitoring Phase

7. **Deploy Model**
    - Integration into a prediction system (e.g., hospital diagnostic tool).
8. **Monitor System Performance**
    - Track real-world accuracy, latency, bias/drift.

E. Feedback Loop for Maintenance

9. "**System Performance Degrades?**"
    - **Yes** → Collect new data → Retrain model.
    - **No** → Continue monitoring.


Collect Dataset (e.g., Patient Records, Medical History)
Preprocess Data (Cleaning, Normalization, Encoding)
Split Data into Training and Testing Sets
Train Machine Learning Model
Evaluate Model Performance (e.g., Accuracy, Precision, Recall)
Model Performance Acceptable?
Yes
No
Deploy Model to Prediction System
Tune Model Hyperparameters
Monitor System Performance
Retrain Model
System Performance Degrades?
Yes
No
Collect New Data
Continue Monitoring
Retrain Model

Key Benefits of Each Diagram Type

1. Use Case Diagrams

- **Purpose**: Capture system requirements from user perspective
- **Value**:
    - Clearly define system boundaries and actor interactions
    - Identify all functional requirements early in development
    - Serve as communication bridge between stakeholders and developers
    - Help prevent scope creep by establishing clear system capabilities

2. Class Diagrams

- **Purpose**: Model system structure and relationships
- **Value**:
    - Blueprint for object-oriented implementation
    - Visualize inheritance, associations, and dependencies
    - Facilitate code generation and maintainability
    - Help identify opportunities for design patterns

3. Sequence Diagrams

- **Purpose**: Illustrate object interactions over time
- **Value**:
    - Detail the flow of messages between components
    - Identify timing issues and synchronization requirements
    - Validate use case realizations
    - Excellent for debugging complex interactions

4. State Diagrams

- **Purpose**: Model behavior of reactive systems
- **Value**:
    - Perfect for systems with complex state transitions

- Help implement state machines in code
- Identify all possible system states and transitions
- Critical for event-driven systems and protocol design

5. Activity Diagrams

- **Purpose**: Model workflows and business processes
- **Value**:
  - Visualize both sequential and parallel processes
  - Excellent for business process modeling
  - Help identify optimization opportunities in workflows
  - Bridge between business requirements and technical implementation

---

Conclusion

This documentation outlined the full development lifecycle of our Oral Cancer Diagnosis System — from understanding the problem domain and collecting data, to preprocessing, modeling, evaluation, and deployment. By leveraging machine learning algorithms, CNNs, and transfer learning, we achieved high accuracy in detecting oral cancer using both structured and image data. Our use of MLflow and Flask for deployment ensures the model is accessible, manageable, and ready for real-world use.

Through a combination of technical precision, medical insight, and practical design, this project stands as a powerful example of how artificial intelligence can support early cancer detection and ultimately improve patient outcomes. This work not only demonstrates our technical capabilities but also reflects our commitment to applying technology for meaningful healthcare impact.

THANKS