Software Requirements Specification (SRS) and Design Document

1. Introduction

1.1 Introduction

Cancer is a group of diseases characterized by the uncontrolled growth and spread of abnormal cells. If the spread is not controlled, it can result in death. Cancer begins when the genetic material (DNA) of a cell becomes damaged or changed, producing mutations that affect normal cell growth and division. When this happens, cells do not die when they should, and new cells form when the body does not need them. These extra cells can form a mass of tissue called a tumor. Cancer can occur anywhere in the body, and it consists of more than 100 different types. It can affect organs, bones, muscles, or blood cells. Depending on the type and stage of the cancer, treatments include surgery, radiation, chemotherapy, and increasingly, targeted therapies that use the body's own immune system to fight the disease.

Cancer and its Global Impact

Cancer is a leading cause of death worldwide, with millions of new cases diagnosed annually. It is a complex disease characterized by uncontrolled growth of abnormal cells in the body. Cancerous cells can invade surrounding tissues and spread to distant parts of the body through the blood and lymph systems, a process known as metastasis. Early detection of cancer is crucial for effective treatment, significantly improving patient survival rates and quality of life. Importance of Early Detection Early detection of cancer plays a vital role in reducing cancer-related mortality. By identifying cancer cells at an early stage, healthcare professionals can intervene before the disease progresses and spreads. Techniques for early detection range from imaging tests (like CT and MRI scans) to tissue biopsies and blood tests. Microscopic analysis of cells plays a key role in early diagnosis, particularly in cancers like leukemia and melanoma, where visual assessment of cell morphology is necessary

Challenges in Cancer Cell Detection

Detecting cancer cells is a challenging task, especially at early stages when the cancer cells are too few or small to be easily identified. Cancer cells display a high level of variation in shape, size, and structure, making it difficult to distinguish them from normal cells.

They also exhibit unique properties, such as uncontrolled growth and darker, enlarged nuclei. These characteristics complicate detection, requiring advanced techniques for accurate identification. Image-Based Cancer Detection Recent advancements in image-based cancer detection methods, supported by computer vision and artificial intelligence, have shown promising results. Medical imaging techniques such as MRI, CT scans, and digital microscopy are combined with machine learning algorithms to automate the process of detecting cancerous cells in medical images.

Segmentation in Medical Imaging

Segmentation is a crucial step in medical image analysis, involving the separation of cancerous regions from healthy tissue in images. In cancer cell detection, segmentation is used to outline the precise boundaries of tumor cells, helping to quantify the size and spread of the cancer. This process is essential for treatment planning, allowing for accurate tumor targeting in therapies such as radiation and surgery. Various techniques, including deep learning models, are used for this purpose, such as Convolutional Neural Networks (CNNs), which have been widely adopted due to their high accuracy in image classification tasks.

Proposed Solution: Cancer Cell Detection and Segmentation

This project focuses on developing a system for cancer cell detection and segmentation using deep learning techniques. By utilizing pre-trained models like Inception and ResNet, the project aims to detect cancer cells in microscopic images and segment them effectively. The model will be trained on publicly available datasets, including UCSB Bio-Segmentation and Bio GPS datasets, and will undergo various experiments to ensure high accuracy and precision.

1.2 Problem Description

Early-stage cancer detection is critical for providing timely treatment and reducing the risk of death due to cancer. Detecting cancer in its later stages often leads to increased suffering and a higher chance of mortality. Cancer cells exhibit significant variations in shape and size, with characteristics such as a larger and darker nucleus compared to normal cells. These cells also grow uncontrollably due to the loss of contact inhibition, a behavior that differentiates them from normal cells, which stop growing when they reach the required body limits. Several tests, such as the Blood Protein Test, Complete Blood Count (CBC), and Tumor Marker Test, are currently used to detect the presence of cancer cells. However, detecting cancer cells in microscopic images presents an additional challenge. It requires segmenting the image into multiple regions and filtering out those that represent cancer cells. Cancer cells also exhibit a pro-survival characteristic that makes them difficult to differentiate from normal cells. Furthermore, early-stage detection is challenging due to the small size of the cancer cells, making it hard to detect them using traditional methods. Thus, developing a system that can accurately detect and segment cancer cells in microscopic images is crucial for improving early diagnosis, which can significantly enhance treatment outcomes and survival rates.

1.3 Problem Statement

Design and develop a Cancer Cell Detection and Segmentation System that uses deep learning techniques to accurately detect cancer cells in microscopic images and segment them for further analysis. This system aims to enhance the early detection of cancer, improving diagnosis accuracy and aiding in treatment planning.

1.4 Objectives

1. To develop a Convolutional Neural Network (CNN) based model for detecting cancer cells from microscopic images.
2. To implement segmentation techniques to accurately isolate cancer cells in the images for further analysis.
3. To improve the detection performance by evaluating accuracy, recall, and mean average precision (MAP) of the proposed model.

1.5 Scope

a) Software Product Identification:

The software product to be developed is the Cancer Cell Detection and Segmentation System. This system will leverage deep learning techniques to identify and segment cancer cells in microscopic images, focusing on improving early cancer detection and assisting medical professionals in making informed treatment decisions.

b) Product Capabilities:

What the software will do:

- Allow users to upload microscopic images of tissue samples.
- Detect cancer cells in the images using pre-trained deep learning models such as ResNet and Inception.
- Segment cancerous regions from healthy tissues in the images.
- Provide performance metrics such as accuracy, recall, and precision for the detection and segmentation tasks.
- Display results in a user-friendly format, allowing pathologists and medical professionals to analyze cancerous areas.

What the software will not do:

- Perform real-time image capture or integration with medical imaging devices (e.g., MRI, CT scanners).
- Diagnose cancer or recommend treatments—it is strictly for detection and segmentation purposes.
- Handle 3D imaging modalities, as the focus is on 2D microscopic images.
- Replace manual expert analysis but will instead assist professionals by providing supplementary information.

c) Application and Benefits:

Applications

The applications of cancer cell detection and segmentation are vast. From improving diagnostic accuracy to aiding in research for new cancer treatments, this technology has the potential to transform the healthcare industry. It can be used in:

- Early Diagnosis: Helping pathologists detect cancer in its early stages.
- Treatment Planning: Assisting doctors in deciding the best course of action for treatment by analyzing the tumor's growth and spread.
- Clinical Research: Providing researchers with detailed data on cancer cell behavior, aiding in the development of new therapeutic strategies.
- Personalized Medicine: Enabling doctors to tailor treatments based on the specific characteristics of a patient's cancer cells.

Benefits:

- The system will help reduce the workload of medical experts by automating the detection and segmentation process.
- It will improve accuracy and consistency in detecting cancer cells, minimizing human error.
- The tool can facilitate faster diagnosis, enabling timely interventions and treatment planning for patients.

Objectives and Goals:

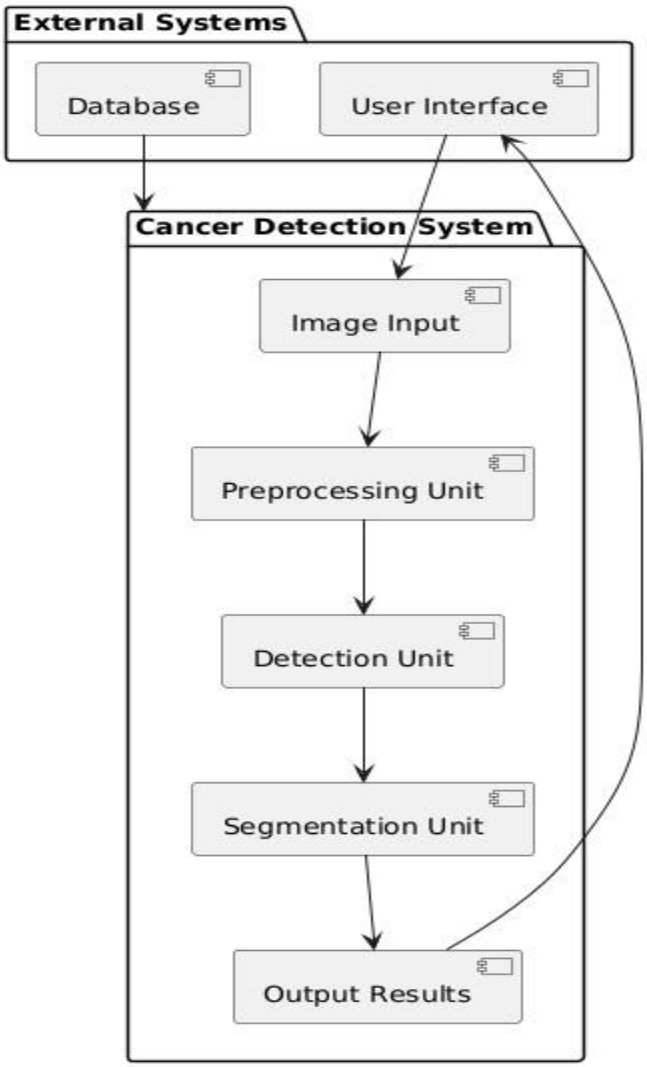
- The goal of this project is to develop a highly accurate, efficient, and easy-to-use system for cancer cell detection and segmentation.
- It aims to provide healthcare professionals with a tool that accelerates cancer cell identification, contributing to better patient outcomes by aiding in early detection and precise treatment planning.

2. Overall description

2.1 Product perspective

The Cancer Cell Detection and Segmentation System is a specialized application aimed at assisting healthcare professionals and researchers in detecting and segmenting cancer cells from microscopic images using deep learning techniques. It functions as a stand-alone desktop application that processes pre-captured medical images to identify and segment cancerous regions. Below are the subsections describing the key components and requirements of the system.

2.1.1 A block diagram showing the major components of the larger system, interconnections, and external interfaces.



2.1.2. Hardware Requirements

To run the **Cancer Cell Detection and Segmentation System** efficiently, the following hardware components are required:

- **Processor**: Intel Core i5 10th generation or higher (i7/i9 preferred for faster performance)
- **RAM**: Minimum 8 GB (16 GB or more recommended for smooth handling of large images and deep learning computations)
- **GPU**: NVIDIA GPU with at least 4 GB VRAM (e.g., NVIDIA GTX 1650 or better). A powerful GPU is essential for accelerating the deep learning tasks such as detection and segmentation.
- **Storage**: At least 100 GB of free storage for dataset storage and application installation. A solid-state drive (SSD) is recommended for faster read/write operations.
- **Display**: Full HD display (1920x1080 resolution) or higher for better visualization of medical images.

2.1.3. Software Requirements

The system requires the following software tools and libraries for its development and execution:

- **Operating System**: Windows 10 or later, or any Linux distribution (e.g., Ubuntu 20.04 or later).
- **Programming Language**:
    - **Python 3.x**: The application is built using Python, which is ideal for machine learning, deep learning, and image processing tasks.
- **Development Environment**:
    - **Anaconda**: Anaconda is used to manage the Python environment and dependencies. It simplifies package management and deployment.
- **Libraries**:
    - **TensorFlow**: For building, training, and deploying the deep learning models used in cancer detection and segmentation.
    - **Keras**: High-level neural networks API, running on top of TensorFlow, used to simplify the development of CNN architectures.

- **OpenCV**: Used for image processing tasks like resizing, normalization, and visualization of segmented cancer cells.
- **PyQt5**: For building the graphical user interface (GUI), allowing users to upload images and view the analysis results.

- **Other Dependencies**:
  - **NumPy, Pandas**: For numerical and data analysis tasks.
  - **Matplotlib/Seaborn**: For plotting graphs, performance metrics, and visualizing results.
- **Datasets**:
  - Access to large, publicly available cancer image datasets such as UCSB BioSegmentation, Bio GPS, and TCGA datasets to train and test the system.

2.2 Product functions

The **Cancer Cell Detection and Segmentation System** is designed to assist healthcare professionals in accurately detecting and segmenting cancer cells in microscopic images. Below are the major functions the software will perform:

1. **Image Upload**:
   - Users can upload microscopic images of tissue samples in supported formats (e.g., JPEG, PNG, TIFF). The system will provide an intuitive interface for selecting and uploading files. 2. **Image Preprocessing**:
   - The system will preprocess the uploaded images to enhance their quality for better detection outcomes. This includes:
     - **Resizing**: Adjusting images to standard dimensions suitable for model input.
     - **Normalization**: Scaling pixel values to a specific range to improve model performance.
     - **Enhancement**: Applying filters and techniques to reduce noise and enhance features of interest in the images.

3. **Cancer Detection**:
    - The software will utilize pre-trained Convolutional Neural Network (CNN) models, such as ResNet or Inception, to automatically detect cancer cells in the preprocessed images. This involves:
        - Analyzing the image to identify potential cancerous regions based on learned patterns.
        - Generating probability scores indicating the likelihood of cancer presence in various image areas.
4. **Segmentation**:
    - After detecting potential cancer cells, the system will segment these regions from the surrounding healthy tissue. This function involves:
        - Utilizing techniques like Fully Convolutional Networks (FCNs) to delineate the boundaries of cancerous regions accurately.
        - Producing a segmented image that highlights the identified cancer cells for further analysis.
5. **Performance Metrics Calculation**:
    - The system will calculate and display key performance metrics, including:
        - **Accuracy**: The proportion of true results (both true positives and true negatives) among the total number of cases examined.
        - **Recall**: The ratio of true positive results to the total number of actual positives, indicating the system's ability to identify cancer cells.
        - **Precision**: The ratio of true positive results to the total predicted positives, assessing the accuracy of the detected cancer cells.
6. **Result Visualization**:
    - The software will provide visual outputs, displaying the original images with highlighted detected cancerous regions and segmented areas. This will enable users to review the analysis effectively and make informed decisions. ○ Users can interact with the visualizations, allowing for zooming in and examining specific areas of interest in the images.

3. Specific requirements

This section describes the specific requirements of the **Cancer Cell Detection and Segmentation System**. It includes detailed descriptions of inputs (stimuli), outputs (responses), and functions performed by the system.

3.1 Functional Requirements

1. **Image Upload**
   - **Input**:
     - Users upload microscopic images of tissue samples.
     - Supported formats: JPEG, PNG, TIFF.
   - **Function**:
     - The system shall validate the file format and size before uploading.
     - If the file format is unsupported or exceeds the size limit, the system shall display an error message.
2. **Image Preprocessing**
   - **Input**:
     - The uploaded image is passed to the preprocessing module.
   - **Functions**:
     - **Resizing**: The system shall resize the image to a predefined dimension (e.g., 256x256 pixels).
     - **Normalization**: The system shall normalize pixel values to a range of [0, 1].
     - **Enhancement**: The system shall apply techniques (e.g., Gaussian filtering) to enhance image quality.
   - **Output**:
     - The preprocessed image shall be ready for analysis.

3. **Cancer Detection**
   - **Input**:
     - The preprocessed image is sent to the cancer detection module.
   - **Function**:
     - The system shall utilize a pre-trained CNN model (e.g., ResNet, Inception) to analyze the image and detect potential cancerous regions.
   - **Output**:
     - The system shall generate a probability score map indicating the likelihood of cancer presence in various regions of the image.
     - Highlighted regions identified as cancerous shall be marked for further processing.
4. **Segmentation**
   - **Input**:
     - The output from the cancer detection module (probability score map) is sent to the segmentation module. ○ **Function**:
     - The system shall apply Fully Convolutional Networks (FCNs) or similar segmentation techniques to isolate cancerous regions from healthy tissue.
   - **Output**:
     - The system shall produce a segmented image that clearly delineates cancerous regions.
     - The boundaries of the detected cancer cells shall be marked.
5. **Performance**

   Metrics Calculation
   - **Input**:
     - The system shall utilize the ground truth data (if available) to calculate performance metrics.

- **Function**:
  - The system shall compute metrics such as:
    - **Accuracy**: Total correct predictions divided by total predictions.
    - **Recall**: True positives divided by total actual positives.
    - **Precision**: True positives divided by total predicted positives.
- **Output**:
  - The calculated metrics shall be displayed on the user interface for user review.

6. **Result Visualization**
   - **Input**:
     - The original image, preprocessed image, detection results, and segmentation results.
   - **Function**:
     - The system shall overlay detection and segmentation results on the original image for visualization.
     - The interface shall allow users to zoom in/out and pan across the images.
   - **Output**:
     - The system shall display the processed images along with highlighted cancerous regions and segmentation overlays.
7. **Exporting Results**
   - **Input**:
     - User requests to export results.
   - **Function**:
     - The system shall allow users to choose the format for exporting (e.g., PDF, CSV).

- The exported document shall include the original images, segmented results, and performance metrics.

- **Output**:
  - The system shall generate and save the results in the selected format at the user-specified location.

3.2 Non-Functional Requirements

1. **Performance Requirements**:
   - The system shall process each image within a maximum time limit of 10 seconds on a standard NVIDIA GPU.
   - The detection accuracy shall be at least 90% based on benchmark datasets.
2. **Usability Requirements**:
   - The GUI shall be user-friendly and designed for medical professionals with varying levels of technical expertise.
   - User instructions shall be provided within the interface to guide users through the process.
3. **Reliability Requirements**:
   - The system shall ensure consistent performance with minimal downtime. ○ Error handling mechanisms shall be implemented to manage and report errors effectively.
4. **Security Requirements**:
   - The system shall implement measures to protect patient data and ensure compliance with regulations (e.g., HIPAA, GDPR).
   - Access to the application shall be restricted to authorized users only.

3.3 External interfaces

This section describes the external interfaces of the Cancer Cell Detection and Segmentation System, detailing the inputs into the system and the outputs generated by the system.

Input Interfaces

1. Image Input Interface ○ Description: The primary interface for users to upload microscopic images of tissue samples.
    - Supported Formats: JPEG, PNG, TIFF.
    - Validation:
        - The system shall validate that the uploaded files meet format and size requirements (e.g., maximum file size of 10 MB).
        - If an unsupported format or oversized file is uploaded, the system shall display an appropriate error message (e.g., "Unsupported file format" or "File size exceeds the limit").
2. User Command Interface ○ Description: The graphical user interface (GUI) allows users to interact with the system, providing commands to upload images, initiate processing, and view results. ○ User Actions:
    - Users can click buttons to upload images, start detection and segmentation processes, and request results.
    - The system shall capture user inputs, such as command clicks and selections from dropdown menus. Output Interfaces

1. Processing Status Output ○ Description: Real-time feedback provided to users regarding the status of the image processing.
    - Content: The interface shall display messages such as "Uploading image...", "Processing image...", and "Processing completed" to keep users informed.
2. Image Visualization Output ○ Description: The processed images will be displayed to users, showing:
        - The original image alongside the preprocessed image.
        - Highlighted regions indicating detected cancer cells.
        - Segmented images with clearly delineated cancerous regions.

- Features:
  - Users shall be able to zoom in and pan across the images for detailed inspection.
  - The output display shall include options to toggle between different visualization modes (e.g., original, detected, segmented).

3. Performance Metrics Output ○ Description: The system will provide calculated performance metrics to assess detection and segmentation accuracy.
   - Metrics Included:
     - Accuracy: The proportion of correctly identified cancerous and noncancerous areas.
     - Recall: The ratio of true positives to actual positives.
     - Precision: The ratio of true positives to predicted positives.
   - Format: The metrics shall be displayed in a clear, tabular format alongside graphical representations (if applicable).

4. Exported Results Output ○ Description: Users can export the analysis results in various formats for reporting and documentation purposes. ○ Available Formats: PDF, CSV.
   - Content:
     - Exported files shall include:
     - The original image.
     - The segmented image with marked cancerous regions.
     - Calculated performance metrics.
   - User Interaction: Users will be prompted to select the desired format and specify a location to save the exported file.
5. Error Messages Output ○ Description: The system will generate error messages for various issues encountered during operation.

- Types of Errors:
  - Input errors (e.g., unsupported file format, file size limits).
  - Processing errors (e.g., model loading failures, runtime errors).
- Display: Errors shall be displayed in a user-friendly manner, including suggestions for resolution when applicable (e.g., "Please upload a valid image format").

3.4 Performance requirements

This section specifies both static and dynamic numerical requirements placed on the software, including system performance under normal and peak workload conditions.

6.1 Static Numerical Requirements

a) **Number of Terminals Supported**:

- The system shall support a single user terminal at a time for image upload, processing, and result visualization.

b) **Number of Simultaneous Users Supported**:

- The system shall support **up to 10 simultaneous users** in a multi-user environment (e.g., if deployed on a server or cloud platform), each processing separate images independently.
- The performance of the system shall not degrade significantly when handling up to 10 users.

c) **Amount and Type of Information to Be Handled**:

- **Image Size**: The system shall handle images of sizes up to **10 MB**.
- **Data Type**: The system shall primarily handle 2D grayscale or colored microscopic images in formats such as JPEG, PNG, and TIFF.
- **Output Data**: For each processed image, the system shall generate:
  - A segmented image with cancerous regions.
  - Performance metrics (accuracy, recall, precision).

6.2 Dynamic Numerical Requirements

a) **Number of Transactions/Tasks Processed**:

- **Normal Workload**: Under normal conditions (e.g., a single user processing one image at a time), the system shall process and analyze an image in under **10 seconds** on a machine equipped with a **standard NVIDIA GPU**.
- **Peak Workload**: Under peak conditions (e.g., multiple users processing images simultaneously), the system shall maintain processing times under **30 seconds** for each image.

b) **Amount of Data Processed**:

- The system shall be able to process at least **100 images per hour** when operating at full capacity (in a server-based or cloud-based deployment).
- During peak workload, the system shall process at least **50 images per hour**. c) **Response Time**:
- The system shall provide feedback within **1 second** after the user uploads an image, displaying a "Processing" status message.
- The system shall deliver the final output (detection, segmentation, and performance metrics) within **10 seconds** for each image during normal conditions.

d) **Memory Usage**:

- The system shall use a maximum of **8 GB of RAM** during the image processing phase.
- The system shall automatically free memory once processing is completed, ensuring that additional images can be uploaded and analyzed without memory leaks.

6.3 Load and Stress Conditions

- The system shall be tested under high-load conditions (e.g., with multiple large images being processed simultaneously) to ensure stability.
- During stress testing, the system shall maintain a failure rate of **less than 1%** for image processing tasks even under heavy workload.

6.4 Human Interaction Requirements

- The system shall ensure that the **GUI responds within 2 seconds** for all user actions (e.g., uploading images, navigating through results).
- The system shall be optimized to minimize lag in displaying processed images and results, even when zooming or panning through large image files.

3.5 Design constraints

This section outlines the design constraints imposed by various factors, including standards, company policies, and hardware limitations, that will impact the development of the **Cancer Cell Detection and Segmentation System**.

Standards Compliance

1. **Regulatory Compliance**:
    - The system shall comply with relevant healthcare regulations, such as:
        - **HIPAA** (Health Insurance Portability and Accountability Act): To ensure the privacy and security of patient data during image handling and processing.
        - **GDPR** (General Data Protection Regulation): If applicable, to protect user data and ensure informed consent for data processing.
2. **Data Handling Standards**:
    - The system shall adhere to standards for medical data handling and imaging, such as:
        - **DICOM** (Digital Imaging and Communications in Medicine): While primarily focused on image transport, the system shall ensure compatibility for input/output with medical imaging systems.

Company Policies

3. **Security Policies**:
    - The system shall implement security measures consistent with company policies, including:
        - User authentication and role-based access control to ensure that only authorized personnel can access sensitive patient data and system functionalities.
4. **Development Policies**:
    - The system shall be developed following company software development lifecycle policies, including:
        - Code reviews and quality assurance testing at each development stage to maintain high-quality software standards.
5. **Documentation Standards**:
    - The system shall be accompanied by comprehensive documentation, including user manuals and technical documentation, in line with company standards for software projects.

Hardware Limitations

6. **Computational Requirements**:
    - The system shall be designed to run efficiently on hardware configurations with:
        - At least an **Intel Core i5 10th generation processor** or equivalent.
        - A dedicated **NVIDIA GPU** with a minimum of **4 GB VRAM** for deep learning computations.
        - **8 GB RAM** as a minimum requirement, with 16 GB recommended for optimal performance.

7.4 Software Limitations

8. **Operating System Compatibility**:
    - The system shall be designed to run on:
        - **Windows 10 or higher** or any **Linux distribution** (e.g., Ubuntu 20.04 or later).
    - Compatibility with specific libraries (TensorFlow, Keras, OpenCV) must be ensured, considering their version dependencies.
9. **Library and Framework Limitations**:
    - The system shall rely on well-documented libraries that are widely supported and updated. Any library used must have community support for troubleshooting and enhancement.

7.5 User Interaction Constraints

10. **User Experience (UX) Requirements**:
    - The user interface shall be designed for usability by medical professionals with varying levels of technical expertise. ○ The design shall adhere to accessibility standards to ensure it is usable by individuals with disabilities (e.g., screen reader compatibility).

3.6 Software system attributes

This section specifies the key attributes of the **Cancer Cell Detection and Segmentation System** that are essential for its operation, maintenance, and overall quality. Each attribute is defined to ensure that its achievement can be objectively verified.

9.1 Reliability

To establish the required reliability of the software system at the time of delivery, the following factors shall be considered:

- **Fault Tolerance**: The system shall be designed to continue operating correctly in the event of a software failure. Critical functionalities should have fallback mechanisms in place to ensure seamless user experience.

- **Error Handling**: The system shall implement robust error handling to catch and manage exceptions without crashing, providing meaningful feedback to users when issues arise.
- **Testing**: The software shall undergo rigorous testing, including unit tests, integration tests, and user acceptance tests, to validate that it meets reliability standards. The system shall achieve a minimum reliability rate of **99.5%** during testing.
- **Monitoring**: The system shall include monitoring tools to log performance and failure events for ongoing reliability assessment.

9.2 Availability

To guarantee a defined availability level for the entire system, the following factors shall be implemented:

- **Uptime Requirement**: The system shall maintain an uptime of **99%**, ensuring that it is accessible to users for processing images and retrieving results.
- **Checkpoint Mechanism**: The system shall implement a checkpoint mechanism to save the state of processing tasks at regular intervals, allowing recovery from failures without data loss.
- **Recovery Procedures**: The system shall include automated recovery procedures to restore functionality in case of failure, minimizing downtime.
- **Scheduled Maintenance**: The system shall have a predefined schedule for maintenance activities, communicated to users in advance to minimize disruption.

9.3 Security

To protect the software from accidental or malicious access, use, modification, destruction, or disclosure, the following security measures shall be adopted:

- **User Authentication**: The system shall implement user authentication mechanisms (e.g., username and password, role-based access control) to restrict access to authorized personnel only.
- **Data Encryption**: Sensitive data (e.g., patient images, personal user information) shall be encrypted both at rest and in transit to protect against unauthorized access and data breaches.
- **Access Controls**: The system shall enforce access controls to limit data visibility and operations based on user roles (e.g., administrators, pathologists).

- **Audit Logging**: The system shall maintain audit logs of all user activities, including image uploads, processing requests, and exports, for accountability and monitoring.
- **Vulnerability Management**: Regular security assessments and updates shall be performed to address potential vulnerabilities and ensure compliance with security standards.

9.4 Maintainability

To enhance the ease of maintenance of the software, the following attributes shall be established:

- **Modular Design**: The system shall be developed using a modular architecture to facilitate easier updates, bug fixes, and enhancements without affecting the entire system.
- **Code Documentation**: Comprehensive documentation of the codebase shall be provided to support developers in understanding and maintaining the system.
- **Version Control**: The system shall utilize version control practices (e.g., Git) to track changes and manage different versions of the software, allowing for easier collaboration and rollback if necessary.
- **Automated Testing**: Automated testing scripts shall be implemented to simplify regression testing and ensure that new changes do not introduce defects.

9.5 Portability

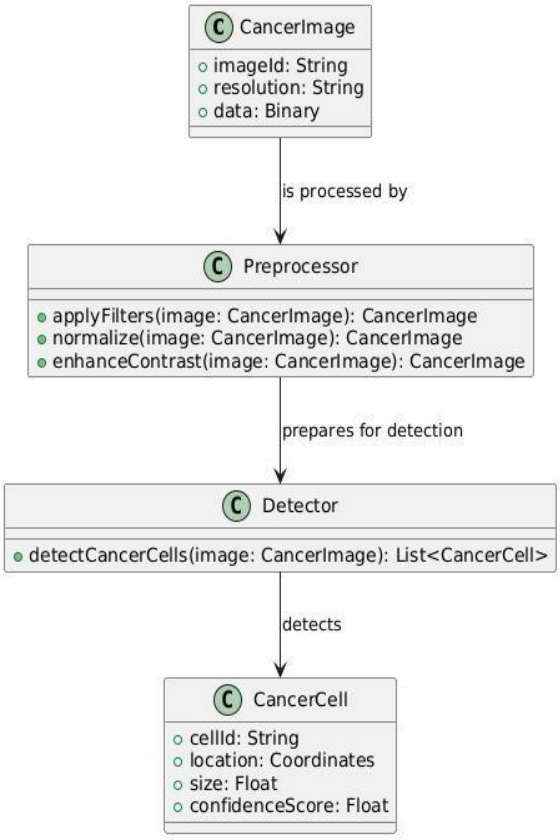
To facilitate the ease of porting the software to other hosts or platforms, the following attributes shall be specified:

- **Platform Independence**: The system shall be designed to be platform-independent, allowing it to run on different operating systems (e.g., Windows, Linux) without requiring significant modifications.
- **Dependency Management**: The system shall use dependency management tools (e.g., Anaconda for Python libraries) to simplify the installation and configuration of required libraries across different environments.
- **Containerization**: The system shall consider using containerization technologies (e.g., Docker) to encapsulate the application and its dependencies, making it easier to deploy on various platforms.
- **Configuration Files**: The system shall utilize configuration files for environment-specific settings, enabling easy modifications when porting to a different host.

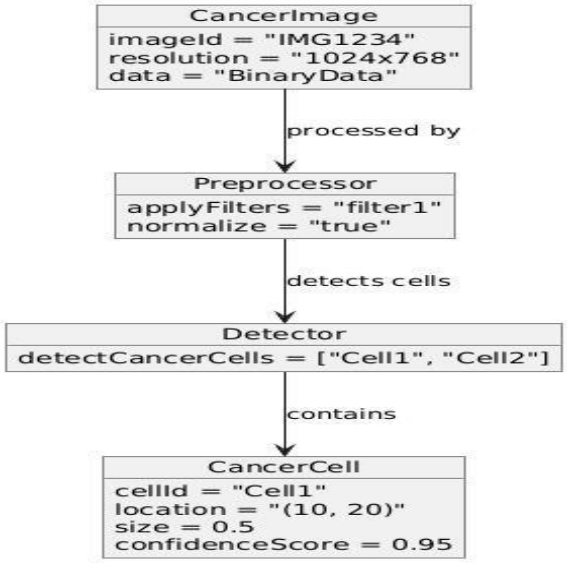
4. Software Design Document

4.1. Structural Design

a. Class Diagrams

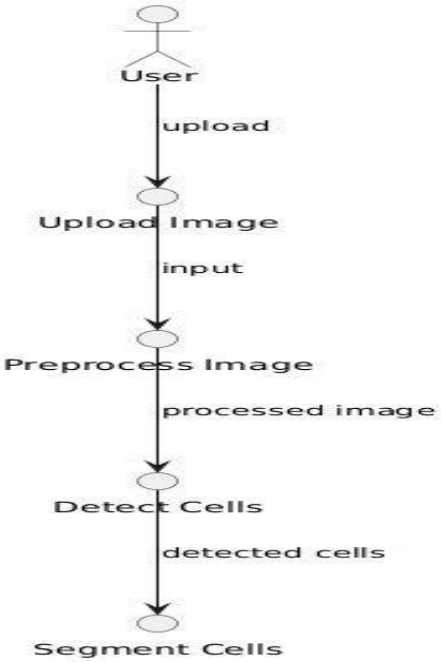


b. Object Diagrams



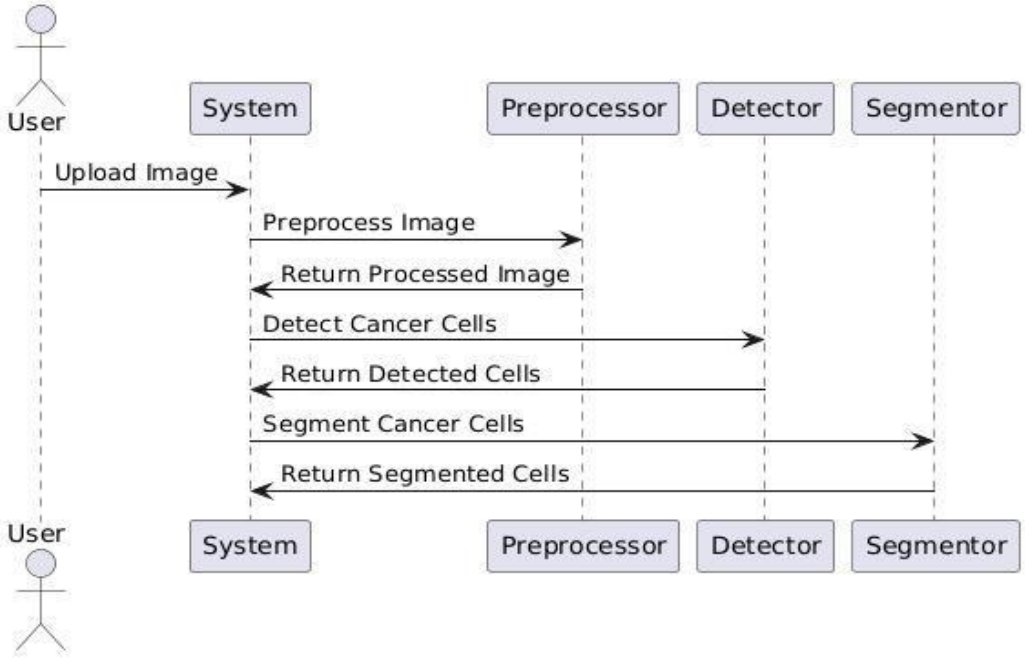
4.2. Data Design

a. Data-flow Diagram

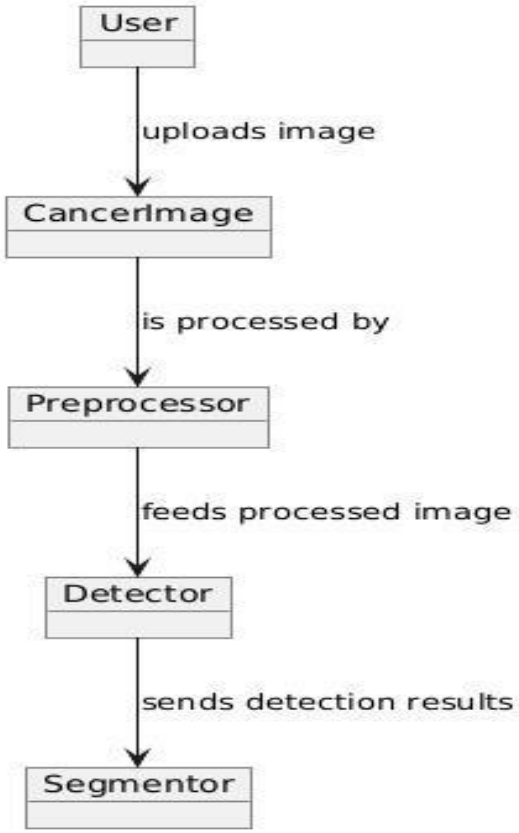


4.3. Behavioral Design

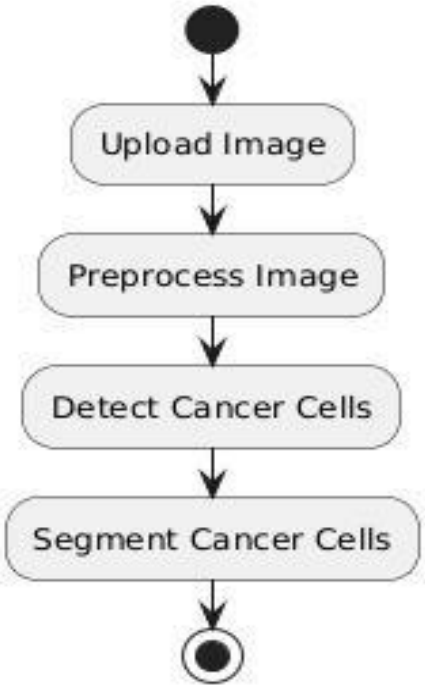
a. Sequence Diagram



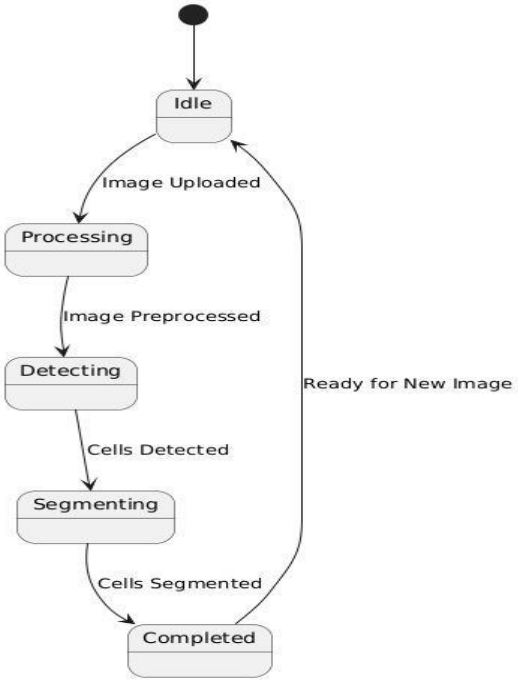
b. Collaboration diagram



c. Activity Diagram

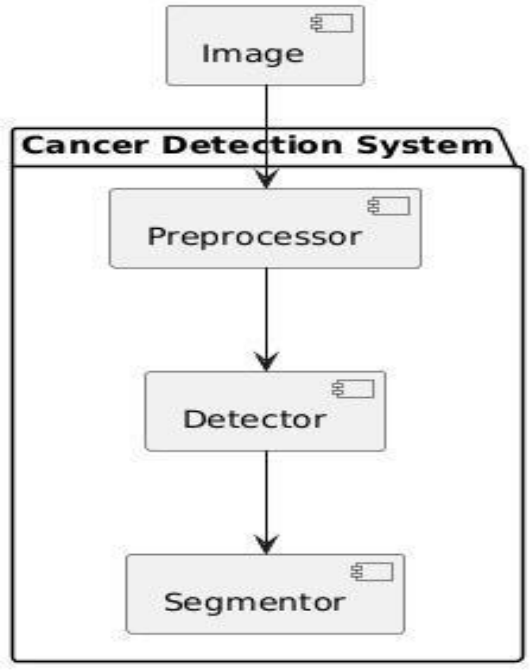


d. State-chart diagram



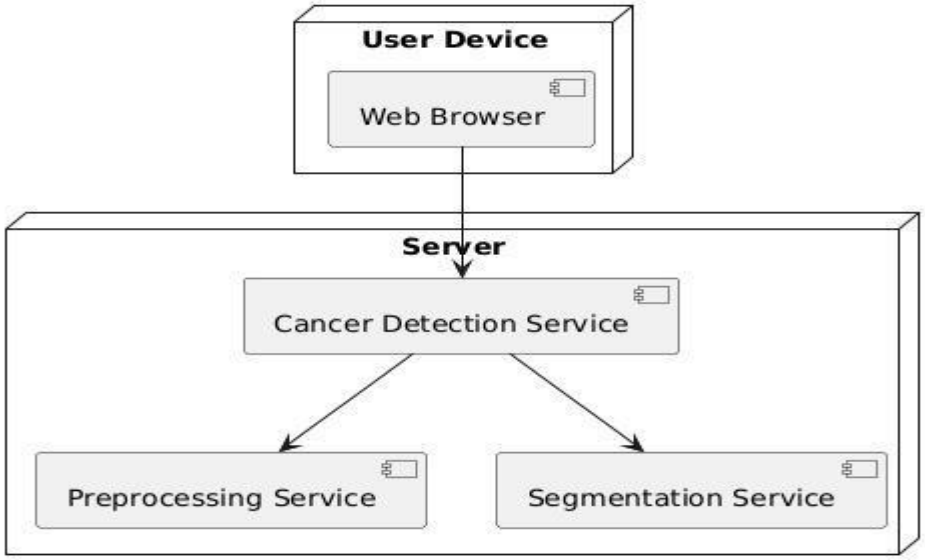
4.3. Implementation Design

a. Component Diagram



4.4. Environment Design

a. Deployment Diagram



5. Algorithms for Functional Requirements

This section describes the algorithms associated with the functional requirements of the **Cancer Cell Detection and Segmentation System**. Each algorithm is designed to fulfill specific functionalities as detailed in the functional requirements section.

10.1 Image Upload and Validation

Algorithm: ValidateImageUpload

1. **Input**: File path of the uploaded image.
2. **Output**: Validation status (success/failure).
3. **Steps**:
    - Check the file extension (JPEG, PNG, TIFF). ○ If the extension is not valid, return "Unsupported file format". ○ Check the file size. ○ If the file size exceeds 10 MB, return "File size exceeds the limit".
    - If both checks pass, return "Validation successful".

10.2 Image Preprocessing Algorithm:

PreprocessImage

1. **Input**: Original image.
2. **Output**: Preprocessed image.
3. **Steps**:
    - Resize the image to 256x256 pixels.
    - Normalize pixel values to the range [0, 1].
    - Apply Gaussian filtering for enhancement.
    - Return the preprocessed image.

10.3 Cancer Detection Algorithm:

DetectCancerCells

1. **Input**: Preprocessed image.
2. **Output**: Probability score map and highlighted regions.

3. **Steps**:
    - Load the pre-trained CNN model (e.g., ResNet or Inception).
    - Pass the preprocessed image through the model.
    - Generate a probability score map indicating the likelihood of cancerous regions.
    - Identify regions with scores above a defined threshold (e.g., 0.5) and mark them.
    - Return the probability score map and highlighted regions.

10.4 Segmentation

Algorithm: Segment Cancer Cells

1. **Input**: Probability score map.
2. **Output**: Segmented image.
3. **Steps**:
    - Load the segmentation model (e.g., FCN).
    - Pass the probability score map through the segmentation model.
    - Apply thresholding to distinguish cancerous regions from healthy tissue.
    - Create a binary mask of the segmented cancerous regions.
    - Return the segmented image with marked boundaries.

10.5 Performance Metrics Calculation

Algorithm: Calculate Performance Metrics

1. **Input**: Ground truth data, detection results, and segmentation results.
2. **Output**: Accuracy, recall, precision.
3. **Steps**:
    - Initialize counters for true positives (TP), false positives (FP), and false negatives (FN).
    - Compare detection results with ground truth to update TP, FP, and FN.
    - Calculate:
        - **Accuracy** = (TP + TN) / (TP + TN + FP + FN)

- **Recall** = TP / (TP + FN)
- **Precision** = TP / (TP + FP) ○ Return calculated metrics.

10.6 Result Visualization

Algorithm: Visualize Results

1. **Input**: Original image, detection results, segmented image, performance metrics.
2. **Output**: Visual display of results.
3. **Steps**:
    - Display the original image. ○ Overlay detection results (highlighted cancerous regions) on the original image.
    - Display the segmented image with boundaries clearly marked.

10.7 Exporting Results

Algorithm: ExportResults

1. **Input**: User-selected format (PDF/CSV), original image, segmented image, performance metrics.
2. **Output**: Exported file.
3. **Steps**:
    - Depending on the selected format:
        - If PDF:
        - Create a PDF document containing the original image, segmented image, and metrics.
        - Save the PDF to the specified location.
        - If CSV:
        - Create a CSV file containing performance metrics.
        - Save the CSV to the specified location.
    - Return the success status of the export operation.

6. References

1. **Arooj S., Atta-ur-Rahman, Zubair M., Khan MF, Alissa K, Khan MA, Mosavi A**, "Breast cancer detection and classification empowered with transfer learning," Frontiers in Public Health, Vol. 10, pp. 1-18, July 2022(SpringerLink).
2. **Dwivedi A.K**., "Artificial neural network model for effective cancer classification using microarray gene expression data," Neural Computing and Applications, Vol. 29, No. 12, pp. 1545–1554, 2018(SpringerLink).
3. **Das P.K., Meher S.,** "An efficient deep Convolutional Neural Network based detection and classification of Acute Lymphoblastic Leukemia," Expert Systems with Applications, Vol. 183, pp. 115311, 2021(SpringerLink).
4. **Mahmood F.,** "Deep adversarial training for multi-organ nuclei segmentation in histopathology images," IEEE Transactions on Medical Imaging, Vol. 39, pp. 3257- 3267, 2019(MDPI).