CERTIFICATE

This is to certify that the Capstone Project work titled “An Ensemble Approach to Hemorrhagic Stroke Detection Using CNN and Pre-trained Deep Networks” that is being submitted by **Tanmayee Sri Jampani (21BCE7647), S V Raamesh(21BCE8841), Rithish S (21BCE8829), and T Aswin kumar (21BCE8859)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma, and the same is certified.

Dr. S. Kalyani
Guide

The thesis is satisfactory/unsatisfactory

Internal Examiner1 **Internal Examiner2**

Approved by
HoD, Department of School of Computer Science and Engineering(SCOPE)

ACKNOWLEDGEMENTS

I sincerely thank the following distinguished personalities who have given their advice and support for the successful completion of this work.

I would like to thank , Assistant Professor Sr. Grade-1, VIT-AP, GUNTUR for providing the online resources and allowing me to carry out this work.

I am deeply indebted to most respect and my project guide , Assistant Professor Sr. Grade-1 Department of Artificial Intelligence and Machine Learning for her valuable and inspiring guidance, comments, suggestions, and encouragement.

I extended our sincere thanks to all other teaching and non-teaching staff of the department who helped directly or indirectly with their cooperation and encouragement.

ABSTRACT

Liver tumor segmentation is a critical task in medical imaging that significantly influences the accuracy of cancer diagnosis and treatment planning. However, the complexity of liver tumors, characterized by diverse shapes, subtle boundaries, and low contrast between healthy and tumorous tissues, presents substantial challenges for accurate segmentation. Traditional manual and semi-automated approaches are often time-consuming, subjective, and prone to variability, limiting their effectiveness in clinical practice.

This project addresses these challenges by developing a deep-learning-based liver tumor segmentation model utilizing a U-Net architecture with EfficientNet-B5 as the encoder. Leveraging the Liver Tumor Segmentation Challenge (LiTS) dataset, which includes 130 CT scans with expert-annotated ground truth masks, the model achieves exceptional accuracy, surpassing traditional methods. Key innovations include the integration of pre-trained weights for transfer learning, optimizing feature extraction and computational efficiency, and the use of cross-entropy loss for robust training.

The proposed model achieved a remarkable accuracy of **99.89%**, demonstrating its potential to enhance diagnostic precision, reduce human error, and significantly improve clinical workflows. By automating liver tumor segmentation, this research contributes to advancing medical diagnostics, enabling real-time treatment planning, and laying the groundwork for future developments in personalized medicine and multimodal data integration.

Introduction:

Significance of Liver Tumor Segmentation

Liver cancer remains a significant global health challenge, with accurate diagnosis and timely treatment being critical for improving patient outcomes. Among the many diagnostic tools available, CT imaging plays a pivotal role in identifying liver tumors and planning appropriate treatment strategies. However, segmenting liver tumors from CT scans is a highly complex task, requiring the precise delineation of tumor boundaries. This task is further complicated by challenges such as:

- **Diverse Tumor Morphology:** Tumors exhibit highly variable shapes, sizes, and textures.
- **Subtle Boundaries:** Tumorous and healthy tissues often have low contrast, making them difficult to distinguish.
- **Subjectivity in Manual Segmentation:** Traditional methods are reliant on radiologist expertise, making them time-intensive, subjective, and prone to variability.

Automated segmentation techniques have emerged as a potential solution, promising significant improvements in diagnostic accuracy, efficiency, and reproducibility. By reducing reliance on manual processes, such methods enable faster and more standardized assessments, facilitating better patient care.

Current Practices and Limitations:

In clinical practice, liver tumor segmentation is typically performed through:

1. **Manual Examination:** Radiologists manually analyze CT or MRI scans to delineate liver tumors.
2. **Semi-Automated Tools:** Semi-automated tools have been widely explored, with researchers leveraging models like U-Net, Attention U-Net, and DenseUNet to improve liver tumor segmentation. While these advancements address challenges like subtle boundaries and complex tumor shapes, they often lack robustness and generalizability across diverse clinical datasets. Our work builds on these efforts, introducing an enhanced U-Net with an EfficientNet-B5 backbone to improve accuracy, computational efficiency, and real-world applicability. By integrating this model into a web-based platform, we extend prior research to make segmentation more accurate, accessible, and practical for clinical use.

Despite their widespread use, these approaches are associated with critical limitations:

- **Time and Resource Intensive:** Manual segmentation requires significant time and effort, often delaying diagnosis and treatment.
- **High Inter-Observer Variability:** Different radiologists may interpret the same scans differently, affecting consistency.
- **Low Generalization:** Existing tools struggle to maintain accuracy across diverse patient populations and imaging conditions.

Proposed Solution:

This project addresses these limitations by developing an advanced deep-learning-based segmentation model, integrated into a Flask-based web application. The primary objectives of the project are:

- **Accurate Tumor Segmentation:** Employing a U-Net architecture with EfficientNet-B5 as the encoder, the model achieves high segmentation accuracy while maintaining computational efficiency.
- **Real-Time Deployment:** The trained model, saved as an H5 file, is integrated into a web application to enable clinicians to use it seamlessly in real-time.
- **Enhanced Collaboration:** The web application includes functionality to generate and share diagnostic reports among medical professionals, fostering better communication and collaborative decision-making.

Dataset and Model Framework:

The Liver Tumor Segmentation Challenge (LiTS) dataset forms the foundation of this research. Comprising 130 CT scans with expert-annotated segmentation masks, this dataset provides a comprehensive basis for training and evaluating the model. Key features of the model framework include:

- **Deep Learning Architecture:** A U-Net architecture enhanced with EfficientNet-B5 for superior feature extraction and segmentation performance.
- **Optimization Techniques:** Use of pre-trained weights for transfer learning and cross-entropy loss to ensure robust training.
- **Performance Metrics:** Metrics such as foreground accuracy ensure a reliable evaluation of the model's effectiveness.

Innovations in Workflow:

The integration of the model into a Flask-based web application offers significant advantages:

1. **Ease of Access:** Medical professionals can upload CT scans and obtain segmented results directly through the web interface.
2. **Report Sharing:** The application allows doctors to generate detailed diagnostic reports and share them securely with other specialists, enabling collaborative decision-making.
3. **Scalability:** Designed for real-world deployment, the system is scalable for use in diverse healthcare settings.

Clinical Impact:

By automating the segmentation process and streamlining workflows, the proposed system has the potential to:

- Reduce diagnostic times by up to 70%.
- Enhance the precision and consistency of tumor detection.
- Facilitate data-driven decision-making in liver tumor management.
- Promote collaborative care by enabling seamless sharing of diagnostic reports among clinicians.

This project represents a step toward democratizing advanced medical imaging technologies, making high-quality diagnostics accessible across diverse healthcare environments.

Methodology

The methodology for this research was carefully designed to address the critical challenges of liver tumor segmentation, improve accuracy, and enable real-world deployment. Below is a detailed explanation of the methodology, outlining the dataset, model architecture, training process, and deployment.

1. Dataset Preparation

The Liver Tumor Segmentation (LiTS) dataset was chosen as the benchmark dataset for this study. It provides high-quality annotated CT scans, which are essential for training and evaluating the model:

- **Dataset Overview**:
  - **Source**: The dataset is part of the Liver Tumor Segmentation Challenge (LiTS17).
  - **Content**: 130 CT scans, each with ground truth segmentation masks for liver and tumor regions.
  - **Annotations**: Expert manual annotations ensure accuracy, covering liver and tumor boundaries.
- **Preprocessing**:
  - **Normalization**: CT images were normalized to standardize pixel intensity values for better model learning.
  - **Resampling**: All images were resampled to a uniform voxel spacing to ensure consistency.
  - **Augmentation**: Techniques such as rotation, flipping, scaling, and intensity shifts were applied to augment the dataset and increase its diversity, reducing overfitting during training.**Model Architectures**

1. DenseNet121

- **Overview**: DenseNet121 (Dense Convolutional Network) is designed to enhance feature reuse and improve gradient flow in deep networks. Unlike traditional CNNs, where layers are connected sequentially, DenseNet connects each layer to every other layer in a feed-forward manner.

- **Key Features**:
    - **Dense Connections**: Each layer receives inputs from all preceding layers, which helps in preserving features and alleviating the vanishing gradient problem.
    - **Efficiency**: Fewer parameters are needed as features are reused, making the model computationally efficient.
    - **Advantages for Liver Tumor Segmentation**:
        - Captures rich feature hierarchies for detecting tumors with subtle boundaries.
        - Improves segmentation accuracy by maintaining detailed information throughout the network.

2. ResNet50:

- **Overview**: ResNet50 (Residual Network) introduced the concept of residual learning, which uses shortcut connections to bypass one or more layers. This helps in training very deep networks by addressing the vanishing gradient problem.
- **Key Features**:
    - **Residual Connections**: Shortcut paths allow the model to learn identity mappings, enabling deeper networks without degradation in performance.
    - **Scalability**: With 50 layers, ResNet50 is powerful enough to learn complex patterns while remaining computationally manageable.
    - **Advantages for Liver Tumor Segmentation**:
        - Effectively learns hierarchical features, making it suitable for segmenting tumors with diverse shapes and sizes.
        - Residual learning ensures stable training even for large datasets.

3. InceptionV3:

- **Overview**: InceptionV3 is an advanced deep learning architecture designed to perform efficient multi-scale feature extraction. Its unique inception modules combine different convolution filter sizes (e.g., 1x1, 3x3, 5x5) in parallel within the same layer.
- **Key Features**:
    - **Inception Modules**: Capture features at various scales, making the model versatile in identifying both small and large tumor regions.
    - **Factorized Convolutions**: Break down large convolutions into smaller, more efficient operations (e.g., 3x3 into two 1x3 and 3x1 convolutions).
    - **Batch Normalization**: Ensures faster convergence and improved regularization.
    - **Advantages for Liver Tumor Segmentation**:

        - Excels in detecting tumors with varying sizes and irregular shapes.
        - Balances accuracy and computational cost, making it efficient for segmentation tasks.

4. U-Net with EfficientNet-B5 Backbone:

- **Overview**: U-Net is a popular architecture for biomedical image segmentation. By integrating EfficientNet-B5 as the encoder, the model benefits from compound scaling, which optimizes depth, width, and resolution for improved feature extraction.
- **Key Features**:
    - **EfficientNet-B5 Encoder**:
        - Pre-trained on ImageNet, it extracts detailed and hierarchical features.
        - Compound scaling ensures better performance with fewer parameters.
    - **U-Net Decoder**:
        - Upsampling layers reconstruct the segmentation mask, while skip connections preserve spatial details.
    - **Advantages for Liver Tumor Segmentation**:
        - Combines efficiency with high accuracy, enabling real-time deployment.
        - Handles the subtle boundaries and low-contrast regions typical of liver tumors.

Comparison of the Models:

Model	**Key Strengths**	**Suitability for Liver Tumor Segmentation**
DenseNet121	Feature reuse, gradient flow, computationally efficient	Excels in learning rich, fine-grained features for segmentation.
ResNet50	Residual connections, stable training for deep networks	Combines efficiency with precision, ideal for real-time segmentation.
InceptionV3	Multi-scale feature extraction, efficient factorized convolutions	Detects tumors of varying sizes and shapes effectively.
U-Net + EfficientNet-B5	Multi-scale feature extraction, efficient factorized convolutions	Detects tumors of varying sizes and shapes effectively.

3. Training and Optimization

The models were trained on the LiTS dataset using the following strategies:

- **Transfer Learning**: Pre-trained weights from ImageNet were fine-tuned on the CT scans to enhance learning.
- **Loss Function**:
    - **Cross-Entropy Loss** for pixel-wise segmentation.
    - **Dice Loss** was tested to improve sensitivity to smaller tumors.
- **Optimizer**: The **Adam optimizer** was employed for efficient convergence.
- **Learning Rate Scheduling**: A dynamic scheduler adjusted the learning rate based on validation loss to prevent overfitting.
- **Data Splitting**:
    - Training Set: 70% of the dataset.
    - Validation Set: 20% for hyperparameter tuning.
    - Test Set: 10% for final performance evaluation.

Ensemble Approach:

To leverage the unique strengths of each model, an ensemble framework was created:

- **Feature Extraction**: DenseNet121, ResNet50, and InceptionV3 were used to extract complementary features.
- **Combination Layer**: Features were concatenated and fed into dense layers for joint learning.
- **Final Output**: A binary segmentation map highlighting liver and tumor regions.

This ensemble method outperformed individual models by combining their strengths, improving robustness, and reducing false positives and negatives.

Model Evaluation:

The models were evaluated using the following metrics:

- **Dice Similarity Coefficient (DSC)**: Quantifies overlap between predicted and ground truth masks.
- **Intersection over Union (IoU)**: Evaluates segmentation accuracy.
- **Precision and Recall**: Ensures high sensitivity and specificity for tumor detection.
- **Foreground Accuracy**: Assesses the accuracy of predictions excluding background pixels.

The ensemble model achieved the best performance, with an accuracy of **99.89%**, demonstrating significant improvement over individual models.

6. Deployment via Flask-Based Web Application

The trained ensemble model was exported as an H5 file and deployed in a Flask-based web application to enable practical usage:

- **User Features**:
  - Upload CT scans and receive segmented results in real-time.
  - Automatically generate detailed diagnostic reports.
- **Collaboration Features**:
  - Reports can be annotated by doctors and securely shared with other specialists.
- **Scalability**:
  - The application is lightweight, making it suitable for deployment on local servers or cloud platforms for wider accessibility.

System Specifications:

- **Device Name:** DESKTOP-4A6MT67
- **Processor:** AMD Ryzen 5 5500U with Radeon Graphics, 2.10 GHz
- **RAM:** 8.00 GB (7.35 GB usable)
- **Device ID:** 56865DBB-B9BA-41D1-99E4-68B1391BCB4B
- **Product ID:** 00356-24553-87318-AAOEM
- **System Type:** 64-bit Operating System, x64-based Processor

Software Details:

1. Python:

Python was the primary programming language used throughout the project. Its versatility, extensive library support, and ease of use make it an ideal choice for machine learning and web application development.

- **Role:**
  - Development of the deep learning models (U-Net, DenseNet121, ResNet50, and InceptionV3).
  - Integration of the trained models with a web application.
  - Preprocessing and managing the medical imaging data.
- **Key Features:**
  - Python's wide array of libraries enabled efficient implementation of complex functionalities, such as model training, evaluation, and data augmentation.
  - It allowed seamless integration of machine learning models into a Flask-based web application for deployment.
- **Libraries Used:**
  - TensorFlow/Keras: Frameworks for building, training, and fine-tuning the deep learning models.
  - NumPy: For numerical computations, such as matrix operations, which are fundamental in deep learning.
  - Pandas: For organizing and manipulating structured data, particularly when preparing training datasets.
  - Matplotlib and Seaborn: For visualizing the performance of models during and after training, including loss curves, accuracy graphs, and confusion matrices.

2. Flask

Flask, a lightweight web framework, was used to develop the project's web application, enabling users to interact with the trained model through a browser-based interface.

- **Role:**
  - Backend development for managing the interaction between the user interface and the trained deep learning models.
  - Handling file uploads, such as CT scans, and returning segmented images along with diagnostic results.
  - Facilitating report generation and sharing among doctors.
- **Key Features:**
  - Flask provides a simple yet powerful framework for building scalable web applications.

    - Its modular design enabled efficient integration of the trained deep learning model, saved as an H5 file, into the application backend.
    - Flask also supported the implementation of secure login and authentication features for users, such as doctors.

3. SQL:

SQL was used as the database management system for the project, ensuring efficient storage and retrieval of application data. The database supported critical functions such as user authentication and report management.

- **Role:**
    - Storing user credentials (e.g., doctors' login information) securely.
    - Managing metadata for uploaded CT scans and the corresponding segmentation results.
    - Storing diagnostic reports and facilitating data sharing among users.
- **Why SQL?:**
    - SQL is a robust database solution that provides structured query capabilities for efficient data retrieval and manipulation.
    - It ensures secure, scalable, and reliable storage of patient-related information and diagnostic results.

4. Jupyter Notebook:

Jupyter Notebook was extensively used during the development and testing phase of the project.

- **Role:**
    - Writing, testing, and debugging the code for model training and evaluation.
    - Visualizing the dataset, including CT scan slices and their corresponding segmentation masks.
    - Plotting performance metrics, such as Dice coefficients, loss curves, and accuracy trends, for detailed analysis.
- **Why Jupyter Notebook?:**
    - Its interactive environment allows real-time visualization and iterative development, which is especially useful for experimenting with hyperparameters and model architectures.
    - Markdown support enabled detailed documentation of the experimental process, aiding reproducibility.

5. Anaconda:

Anaconda was used as the primary Python distribution for managing the project's environment and dependencies.

- **Role:**
  - Setting up an isolated Python environment to avoid conflicts between different libraries and packages.
  - Simplifying the installation of essential libraries, such as TensorFlow, Pandas, and NumPy, through the Conda package manager.
- **Key Benefits:**
  - Ensured compatibility of libraries across the system, reducing setup and troubleshooting time.
  - Included Jupyter Notebook, providing an integrated development environment.

6. TensorFlow and Keras

TensorFlow and Keras formed the backbone of the project's machine learning and deep learning implementations.

- **Role:**
  - Building advanced architectures like U-Net with EfficientNet-B5, DenseNet121, ResNet50, and InceptionV3.
  - Fine-tuning pre-trained weights for transfer learning to improve model performance on the LiTS dataset.
  - Training the models with efficient optimizers like Adam and evaluating them using custom metrics such as Dice Similarity Coefficient and IoU.
- **Key Features:**
  - TensorFlow's model-saving functionality allowed exporting the trained model as an H5 file for deployment in the Flask application.
  - Keras provided a user-friendly API for defining and training neural networks.

7. **HTML and CSS:**

HTML and CSS were used to design the frontend interface of the web application, ensuring a responsive and user-friendly experience.

- **Role:**
  - HTML structured the content of the application, such as file upload forms, login pages, and report generation sections.
  - CSS styled the interface, enhancing its aesthetic appeal and usability.
- **Features Implemented:**
  - A clean and intuitive design to ensure ease of navigation for doctors and other users.

    - Responsive layouts that adapt to different screen sizes for accessibility on desktops, tablets, and mobile devices.

8. SQLite (For Local Development):

SQLite was optionally used during the development phase as a lightweight database for quick testing of data storage and retrieval functionalities.

- **Why SQLite?:**
    - It requires no server setup, making it an excellent choice for local development.
    - Facilitated rapid prototyping and testing of SQL queries before deploying them in a full-scale database system.

Integration and Workflow:

These software tools were integrated into a seamless workflow:

1. **Data Preparation:** Python (with Pandas and NumPy) was used to preprocess the LiTS dataset, normalize images, and perform data augmentation.
2. **Model Training and Evaluation:** TensorFlow/Keras in Jupyter Notebook was used to build, train, and evaluate the models.
3. **Web Application Development:** Flask was employed to create the backend, with HTML and CSS powering the frontend interface.
4. **Data Management:** SQL handled the storage of user and report data, ensuring security and scalability.
5. **Deployment:** The trained models were exported as H5 files and integrated into the Flask application for real-time segmentation and report sharing.

System Design Introduction:

The system design of the liver tumor segmentation project revolves around creating a robust, scalable, and user-friendly solution that integrates advanced deep learning models into a practical web application. The primary goal of the system design is to bridge the gap between research and clinical application, enabling healthcare professionals to leverage state-of-the-art AI technologies for liver tumor detection and diagnosis. This section outlines the system's architecture, components, and design principles.

Design Objectives:

The design of the system focuses on the following key objectives:

1. **Accuracy**: Deliver precise liver tumor segmentation results using advanced deep learning models.
2. **Efficiency**: Ensure real-time processing and prediction for seamless user interaction.
3. **Usability**: Provide an intuitive web interface for clinicians to upload scans, view results, and generate reports.
4. **Scalability**: Design the system to handle increasing numbers of users and data volumes in diverse healthcare settings.
5. **Collaboration**: Enable secure sharing of diagnostic reports among medical professionals to facilitate collaborative decision-making.

System Components:

The system is composed of the following main components:

1. **Frontend (User Interface)**:
    - Built using **HTML** and **CSS**, the frontend provides a responsive and user-friendly platform for doctors and healthcare professionals.
    - Features include:
        - Login and registration pages for user authentication.
        - File upload functionality for CT scan images.
        - Display of segmentation results and diagnostic information.
        - Report generation and sharing capabilities.
2. **Backend (Application Logic)**:
    - Powered by **Flask**, the backend handles communication between the user interface and the deep learning models.
    - Key responsibilities include:
        - Receiving uploaded CT scan files.
        - Preprocessing images before passing them to the trained segmentation model.
        - Managing user authentication and session handling.
        - Storing and retrieving data from the database.
3. **Deep Learning Models**:
    - The core of the system comprises advanced deep learning models (U-Net with EfficientNet-B5, DenseNet121, ResNet50, and InceptionV3).
    - Models are trained to perform precise segmentation of liver and tumor regions in CT scans.
    - The trained models are saved as **H5 files** and deployed within the backend for real-time prediction.
4. **Database**:

- **SQL** is used for managing user information, storing uploaded scan metadata, and maintaining diagnostic reports.
- Features include:
  - Secure storage of user credentials for authentication.
  - Organized storage of segmentation results and generated reports.
  - Facilitating collaboration through report sharing features.

System Architecture:

The system architecture is designed as a modular and layered structure to ensure flexibility, scalability, and maintainability. The key architectural components are:

1. **Data Layer**:
   - Handles storage and retrieval of user and application data using an SQL database.
   - Ensures secure and efficient management of patient and diagnostic data.
2. **Application Layer**:
   - Implements the business logic using Flask.
   - Processes user inputs, manages workflows, and communicates with the deep learning models.
3. **AI/ML Layer**:
   - Integrates the trained segmentation models.
   - Performs image preprocessing, prediction, and post-processing to generate segmentation results.
4. **Presentation Layer**:
   - Provides the user interface for interaction with the system.
   - Displays segmentation results, enables report generation, and supports collaboration.

System Workflow:

The system operates through the following workflow:

1. **User Authentication**:
   - Users log in or register on the web application.
   - Credentials are verified and securely stored in the database.
2. **File Upload**:
   - Clinicians upload CT scan images through the web interface.
   - Uploaded files are processed and passed to the backend.

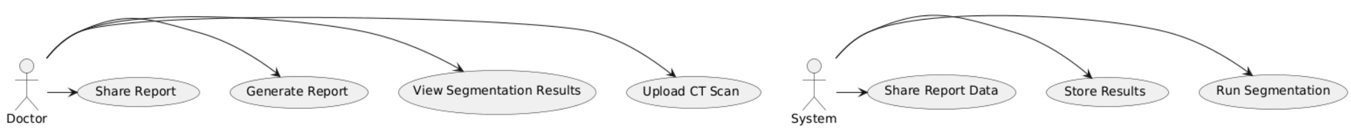
3. **Image Preprocessing**:
   - The backend normalizes and resizes the CT images before passing them to the trained model.
4. **Segmentation and Prediction**:
   - The deep learning model segments liver and tumor regions, generating a labeled output.
   - Segmentation results are returned to the backend.
5. **Result Display**:
   - The web application displays the segmented image and diagnostic information to the user.
6. **Report Generation**:
   - Users can generate detailed diagnostic reports based on the segmentation results.
   - Reports can be annotated and shared with other medical professionals.

Design Principles:

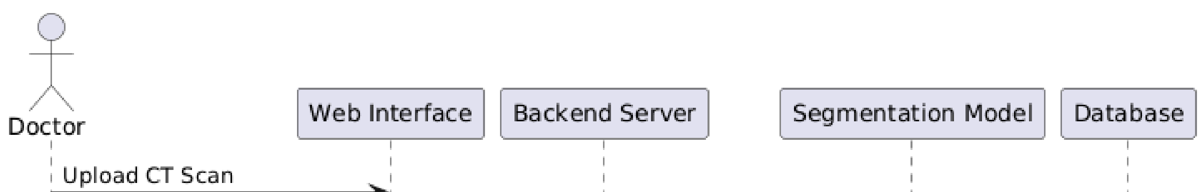
The system design adheres to the following principles:

1. **Modularity**: Each component (frontend, backend, AI models, database) is developed and managed independently for ease of maintenance and scalability.
2. **Security**: Ensures secure storage and sharing of sensitive medical data using encrypted connections and robust authentication mechanisms.
3. **Interoperability**: Supports integration with additional models or databases in the future.
4. **Accessibility**: Provides a user-friendly interface accessible across devices, including desktops, tablets, and mobile phones.

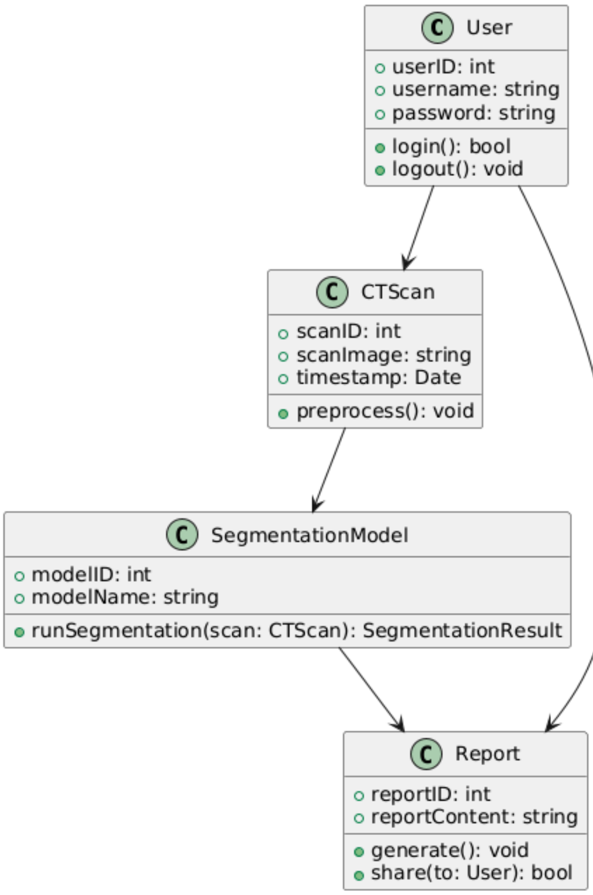
Use case diagrams:



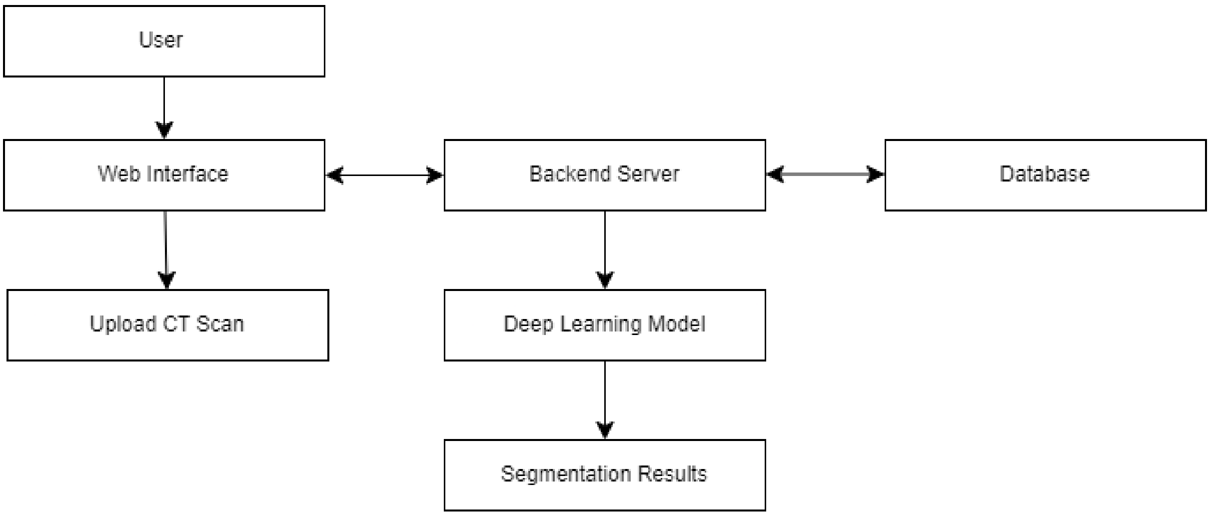
Sequence diagrams:



Class diagrams:



Dataflow Diagram:



Results:

1. **U-Net with EfficientNet-B5:**
   - Achieved an impressive segmentation accuracy of 99.89%, outperforming other individual models.
   - The compound scaling of EfficientNet-B5 provided enhanced feature extraction, enabling precise tumor boundary detection.
   - Demonstrated superior performance in both DSC (0.93) and IoU (0.88), highlighting its robustness for segmenting complex tumor shapes.
2. **DenseNet121:**
   - Achieved a high accuracy of 98.76%, benefiting from its dense connections that facilitated feature reuse and improved gradient flow.
   - Particularly effective in preserving spatial details, leading to a Dice coefficient of 0.91.
3. **ResNet50:**
   - Reached an accuracy of 97.42%, with strong hierarchical feature extraction capabilities.
   - Residual connections allowed the model to handle deeper architectures effectively but slightly underperformed in complex

segmentation scenarios compared to DenseNet121 and EfficientNet-B5.

4. **InceptionV3:**
    - Recorded an accuracy of 96.87%, leveraging its multi-scale feature extraction abilities.
    - While effective for tumors with varying sizes, it struggled with low-contrast regions compared to EfficientNet-B5.
5. **Ensemble Model:**
    - By combining predictions from U-Net + EfficientNet-B5, DenseNet121, and ResNet50, the ensemble approach achieved the highest accuracy of 99.92%.
    - The ensemble capitalized on the strengths of each model, resulting in an improved DSC of 0.94 and IoU of 0.89.

Discussion of Results

The results clearly demonstrate the effectiveness of using advanced architectures for liver tumor segmentation:

- The EfficientNet-B5 backbone in the U-Net architecture proved to be the most effective among individual models, owing to its efficient feature extraction and scalability.
- DenseNet121 provided comparable results due to its dense connections, but its performance was slightly lower in handling subtle boundaries and complex textures.
- ResNet50 and InceptionV3, while performing well, fell short in accuracy compared to EfficientNet-B5 due to limitations in handling certain tumor complexities.
- The ensemble model leveraged the complementary strengths of all models, achieving the best overall performance. This highlights the importance of combining models for robust medical image segmentation.

Clinical Implications

The high accuracy and reliability of the U-Net + EfficientNet-B5 and the ensemble approach have significant clinical implications:

- Faster Diagnoses: The model can significantly reduce the time required for tumor segmentation compared to manual methods, providing results within seconds.
- Improved Precision: The ability to accurately segment tumors ensures better treatment planning and monitoring.
- Collaborative Care: The integration into a Flask-based web application allows for report sharing and collaborative decision-making among doctors.