🧾 🎯 **Project Title:** **MEDICAL AI DIAGNOSIS SYSTEM - FULL-STACK**  
📅 **Project Timeline:** October 2023 - August 2024  
🎥 YouTube Demo: <https://youtu.be/_0w1zhbn-rc>   
📦 GitHub Source Code: <https://github.com/IvanSicaja/2023.07.31_GitHub_Medical-AI-Diagnosis-System---Full-stack>   
---------------------------------------------------------------------------------------------------------------

🏷️ My Personal Profiles: ⬇︎  
🎥 Video Portfolio: To be added  
📦 GitHub Profile: <https://github.com/IvanSicaja>  
🔗 LinkedIn: <https://www.linkedin.com/in/ivan-si%C4%8Daja-832682222>  
🎥 YouTube: <https://www.youtube.com/@ivan_sicaja>  
 ----------------------------------------------------------------------------------------------------------------

### 📚🔍 Project description: ⬇︎⬇︎⬇︎

### 💡 App Purpose

The **Custom AI Medical Diagnosis App** offers a **user-friendly interface** for uploading **medical images** such as **CT, MRI, X-ray, PET, and Mammography scans**, depending on the **training dataset**. Focused on **brain cancer, skin cancer, and lung pneumonia diagnoses**, the app employs a **Convolutional Neural Network (CNN)** powered by **TensorFlow** and **Python**. After analysis, the **chatbot** provides a **text-based diagnosis** with corresponding **probability**, along with **medical advice**. Enhancing **user experience**, a friendly **doctor AI created avatar** interacts with users. Relevant **data** is saved in a **SQL database dynamically**. The app is designed with **Flask, Jinja2, Bootstrap, and JavaScript**, containerized with **Docker**, orchestrated with **Kubernetes**, and can be hosted on **AWS EKS** or any other platform which supports **Kubernetes**.

### 🧠 How It Works

PROJECT HIGHLIGHTS:

* **Data collection and preprocessing:** Since no access to the **official medical database**, a **custom database** should be created. Developed **Python script** that unifies the **width and the height of all images** in the database and unifies the **number of the image dimensions** (**number of the color channels, transparency, etc.**) to the **grayscale image**. Also developed the **Python script** which converts all **images** in a **.csv database** with **Pandas** where every row represents the **image instance normalized pixel value** together with the corresponding **class label**.
* **Build, train, and evaluate the neural network model:** **TensorFlow** was used to create a **convolutional neural network** from scratch, **train** and **save the CNN model**. A **validation accuracy score** was used for the **model validation**.
* **Building the web page frontend UI:** Because the idea of this project is to build a **full stack deployed app**, it was needed to develop the **entire web page**. **HTML, CSS, and Bootstrap framework** was used for the **frontend development**.
* **Building the web page backend:** As a **backend Python framework**, **Flask** is used. It allows us to get the **uploaded medical images** from the front end (**CT, MRI, X-ray, PET**). The **image** is converted with **Python backend function** to the corresponding **array form**, which was used for the **model training** with **TensorFlow and Keras**, and sent to the trained **convolutional neural network model** for **analysis**. The **result** that we got from the **neural network** is encoded to the corresponding **class** and sent back to the frontend like the **diagnosis text result** with corresponding **probability** to the **user** with the usage of **Jinja2**.
* **Deployment:** The **entire app** is **containerized** with the usage of **Docker Desktop** and the app's **Docker image** is uploaded to the **Docker Hub platform**. For the **deployment** is used **Google Kubernetes container orchestration tool**. The app is deployed locally with **Minikube** and tested for **performance**. The **domain** is bought on the **Namecheap platform** and the app can be deployed on any platform which supports **Kubernetes** such as: **Amazon Elastic Kubernetes Service (AWS->EKS), Microsoft Azure Kubernetes Service (AKS), Google Kubernetes Engine (GKE)**, etc.

**Data collection and preprocessing**  
Since no access to the official medical database, a custom database was created. Developed **Python scripts** that **unify image dimensions** (width, height, color channels, transparency) to **grayscale images**. Also created a script that **converts all images into a .CSV dataset** using **Pandas**, where each row represents a **normalized pixel value** with the corresponding **class label**.

**Used datasets**  
You can download the raw datasets from the following links:

* [**Brain Tumor Detection**](https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection)
* [**Melanoma Skin Cancer**](https://www.kaggle.com/datasets/hasnainjaved/melanoma-skin-cancer-dataset-of-10000-images)
* [**Chest X-ray Pneumonia**](https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia)

These datasets were downloaded and edited with **preprocessing scripts**:

* **unifyImageSizeFromFolderAndSave.py**
* **makeCSVDatasetFromImageFolders.py**

The **processed datasets** were saved as:

* **Br35H.csv**
* **melanoma\_cancer.csv**
* **chest\_xray.csv**

**Training process**  
Model training was performed using multiple **experimental CNN training scripts** with different architectures and optimization techniques, including:

* **1.0\_trainCNN.py**
* **1.1\_trainCNN\_CatchComplexPatterns.py**
* **1.2\_trainCNN\_CatchComplexPatterns\_DA\_EarlyStopping.py**
* **1.2\_trainCNN\_CatchComplexPatterns\_DA\_EarlyStopping\_128x128.py**
* **1.2\_trainCNN\_CatchComplexPatterns\_DA\_EarlyStopping\_MoreLayers\_128x128.py**
* **1.3\_trainCNN\_CatchComplexPatterns\_DA\_EarlyStopping\_ClassBalanced\_ReducedLearningRate.py**
* **1.4\_trainCNN\_MediumComplexity\_DA\_EarlyStopping\_Droput\_L2.py**
* **1.5\_trainCNN\_smallerNumberOfFilters.py**

**Database creation**  
For saving diagnostic results, the **database schema and tables** were created using:

* **create\_db.py**
* **db\_models.py**

**Application runtime**  
The **main script** which runs the entire app is:

* **main.py**

### ⚠️ Note

* This concept is for **local use only**, and significantly **improved training data** is essential for **real-world applications**.
* Please note that sharing project files is strictly prohibited. This project is intended solely for **educational** and **non-commercial** purposes.

### 🔧 Tech Stack

**Python, Convolutional neural network, Open CV – Computer vision, SQL, Pandas, Tensorflow, Keras, Scikit-learn, Git, GitHub, Docker Desktop, Docker Hub, Kubernetes, Minikube, Namecheap, HTML, CSS, Bootstrap, Javascript, Flask, Jinja2, Linux, AI Image generation tools as Adobe Firefly and Playground.com**

### 📣 Hashtags Section

**# #CustomAI #MedicalDiagnosis #FullStackApp #Python #CNN #ConvolutionalNeuralNetwork #TensorFlow #Keras #OpenCV #ComputerVision #SQL #Pandas #Flask #Jinja2 #Bootstrap #JavaScript #HTML #CSS #Docker #DockerHub #Kubernetes #Minikube #AWS #AWS\_EKS #AzureAKS #GKE #MedicalImaging #BrainCancer #SkinCancer #LungPneumonia #AIAvatar #Chatbot #DeepLearning #MachineLearning #DataPreprocessing #ImageProcessing #HealthcareAI #AIInHealthcare #TechStack #Deployment #Linux #AIImageGeneration #AdobeFirefly #PlaygroundAI #Git #GitHub #FullStackDevelopment**