BRAIN TUMOR PREDICTION

A project report
Presented to
CMPE-258
Spring, 2025

By
Dustin Nguyen, Chint Patel,
Vikranth Jakamukala, Nikhil Khanchandani
4/7/25

GitHub Link

https://github.com/nikhilkhanchandani/projects/tree/main/schoolprojs/spring2025/258/final

ABSTRACT

[BRAIN TUMOR PREDICTION]

By Dustin Nguyen, Chint Patel, Vikranth Jakamukala, Nikhil Khanchandani

ABSTRACT

Time is the one constant in our lives—ever-present, yet seemingly always in short supply. Whether you're dashing to your next meeting, hurrying to welcome a newborn, or racing against the clock to complete a critical project, time never feels like it's on our side. Our solution addresses this universal challenge by prioritizing rapid, early detection of brain tumors to substantially improve patient outcomes. We propose the development of an AI-driven diagnostic platform that leverages state-of-the-art computer vision and deep learning techniques to analyze medical imaging—particularly MRIs—and deliver accurate, near-instantaneous assessments. By identifying tumors at their earliest stages, this tool empowers clinicians to initiate timely interventions, ultimately saving precious time—and lives—when it matters most.

Table of Contents

Chapter 1. Introduction

1.1 Project goals and objectives

The objective of the project is to shorten the time and reduce the cost of detecting brain tumors from MRI imaging. To do this we will train a CNN model to detect brain tumors from a dataset of MRI images.

1.2 Problem and motivation

Over time MRI analysis scans have taken quite some time to analyze and are difficult to detect if someone does not look for it specifically. Instead of going through a long process of trying to get an appointment with a doctor, going through life without knowing there is a brain tumor by not having it checked out before, we aim to provide an easier solution. Our project is important in offering an easy and efficient way to detect if a person has a brain tumor. We want to help those people who have so much time left and detect their brain tumors to receive the utmost care before it is too late. Members of our team have had people in their lives who have been diagnosed with a brain tumor, but were already too late for treatment and while some have found the brain tumor, they would have an easier time if detected sooner.

1.3 Project application and impact

The application of our project is meant to help with time, cost, and accuracy of predicting the potential brain tumor. This would help not only doctors, but help patients to receive the utmost care at the earliest time. By implementing this type of tool, we hope to achieve a costless and efficient method for everyone to use. With doctors being able to use this, they would have a less likely chance of misdiagnosing a patient, results received earlier, catching a brain tumor when they were not looking for one in the first place, and to provide accurate readings.

1.4 Project results and expected deliverables

Our project goal is to produce results that are accurate in predicting if there is a brain tumor in an MRI. We want to have it be correct in delivering the results in order to call this project a success. The expected deliverables we aim to achieve is to deliver a working application on the web browser to give a correct response to the MRI photo given.

1.5 Market research

According to Touchstone Medical Imaging, MRI results are analyzed and sent around 3-5 days after the scan. While this may not take too long, we also pose the question of “How do we know to get it checked?” This other article by the Duke University of Medicine, they state that “Detecting a tumor when no symptoms are present can be challenging, as tumors often cause noticeable signs only when they reach a certain size or affect surrounding tissues” (Preston Robert Tisch Brain Tumor Center). To combat this, we want to have an automated process to run on the brain MRI even if they are not looking for it.

Chapter 2 Background and Related Work

2.1 **Background and used technologies**

Some of the key concepts: preprocessing MRI scan image from Kaggle dataset, detecting if a person has a brain tumor or not which falls under binary classification, detecting the type of brain tumor falls under multi class classification, and evaluating metric to see how accurate the prediction of the model is.

To this project it requires understanding the medical aspect and also the software aspect as well. It was an understanding combination of imaging, data science and machine learning, and web development. The technologies we used were google colab to an accurate model, python for the backend portion, react for the frontend portion, visual studio which was our primary IDE, and material UI to design our website.

2.2 **State-of-the-art**

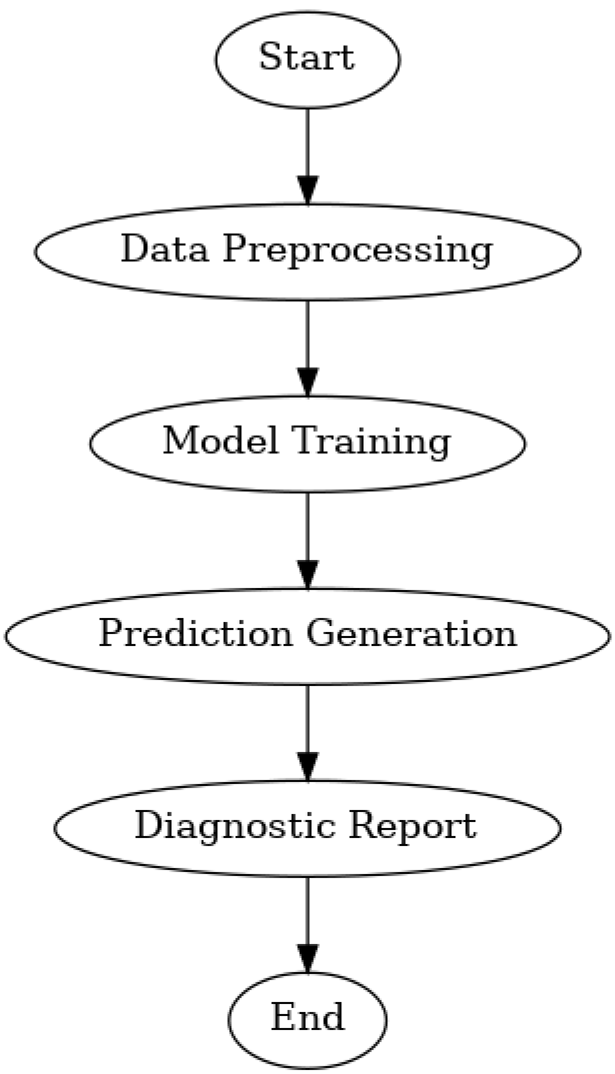
There is an AI tool called DEPLOY which has the same functionality as our project. Kenneth Aldape and Eytan Ruppin came up with the model DEPLOY to help doctors recommend the right treatments for the patients. It detects if a person has a brain tumor and also predicts what type of tumor the person has. They also use DNA methylation to correctly predict the brain tumor type. They have created a total of three models, the first being a model that predicts the brain tumor subtype based on images provided, second is a model which generates prediction of DNA methylation based on the images provided, and the third model which predicts the tumor type based on the patients sex and age. These three models then were put together into one main model called DEPLOY. This model has a prediction score of 95% which is very high and effective. The article states that the DEPLOY model will be beneficial in the future because it can be used to predict other cancers and save many lives.

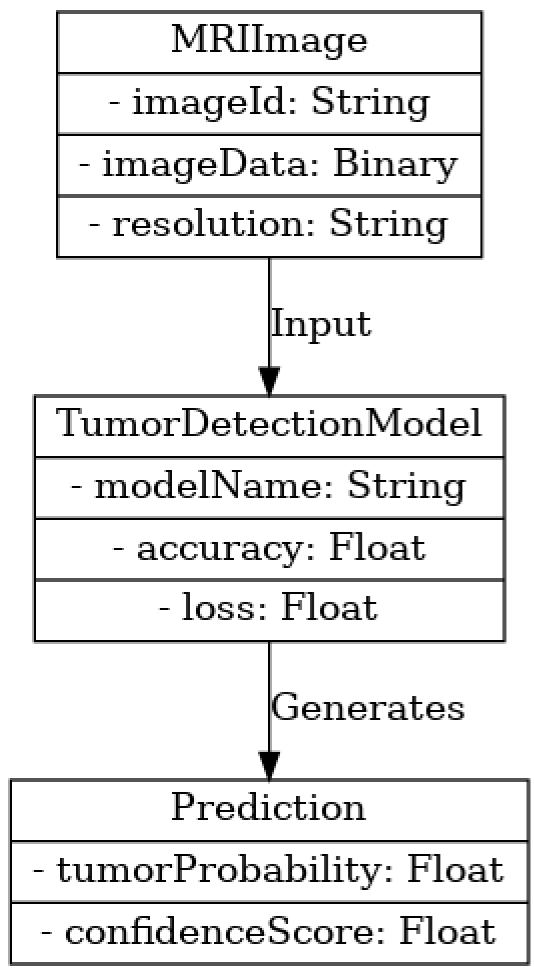
Chapter 3 Data

3.1. Software Requirements

This project focuses on medical professionals and radiologists who extensively use MRI to diagnose brain tumors. The current technology can be easily combined with the existing MRI tools and reports to provide better insights into the imaging. There are 4 main requirements for this project.

1. High Accuracy: Better identification of tumors than current doctors.
2. Cost Efficiency: Reducing the cost of diagnosing a brain tumor via MRI
3. Time Efficiency: Shortening the time from MRI image taken to result interpretation.
4. Usability: provide a seamless and pleasant experience for the users to understand their results.





3.2. Customer-oriented requirements

There will be 3 main groups targeted for this: Radiologist, Medical Researchers, and Patients.

Use Case ID	Use Case Name	Description	Actors
UC7	Access Diagnostic Results	Patients access diagnostic reports generated based on system predictions, provided by radiologists.	Patient
UC8	Consent for Data Usage	Patients give consent for their anonymized MRI scans to be used for model training and evaluation.	Patient
UC9	Receive Enhanced Treatment	Patients receive faster and more accurate diagnosis, leading to tailored and timely treatments.	Patient

Use Case ID	Use Case Name	Description	Actors
UC4	Train Model	Researchers train the CNN model with new datasets or updated architectures.	Medical Researcher
UC5	Evaluate Model Performance	Researchers evaluate the model's accuracy, precision, and recall using validation data.	Medical Researcher
UC6	Fine-Tune Model	Researchers fine-tune the CNN hyperparameters for improved performance.	Medical Researcher

Use Case ID	Use Case Name	Description	Actors
UC1	Upload MRI Image	Radiologists upload MRI scans of the brain into the system for tumor analysis.	Radiologist
UC2	View Prediction Results	Radiologists view system-generated predictions with highlighted tumor regions and confidence scores.	Radiologist
UC3	Generate Diagnostic Report	Radiologists use predictions to generate and export diagnostic reports for patients.	Radiologist

3.3. System function requirements

Functional requirements include raw MRI images that are used to train the CNN model. Then test MRI images are used to validate the model. The outputs would include if there is a tumor in the image and how confident the model is.

Component	Input	Behavior	Output
Data Preprocessing	Raw MRI images	Resize, normalize, and augment the images for model compatibility.	Processed images

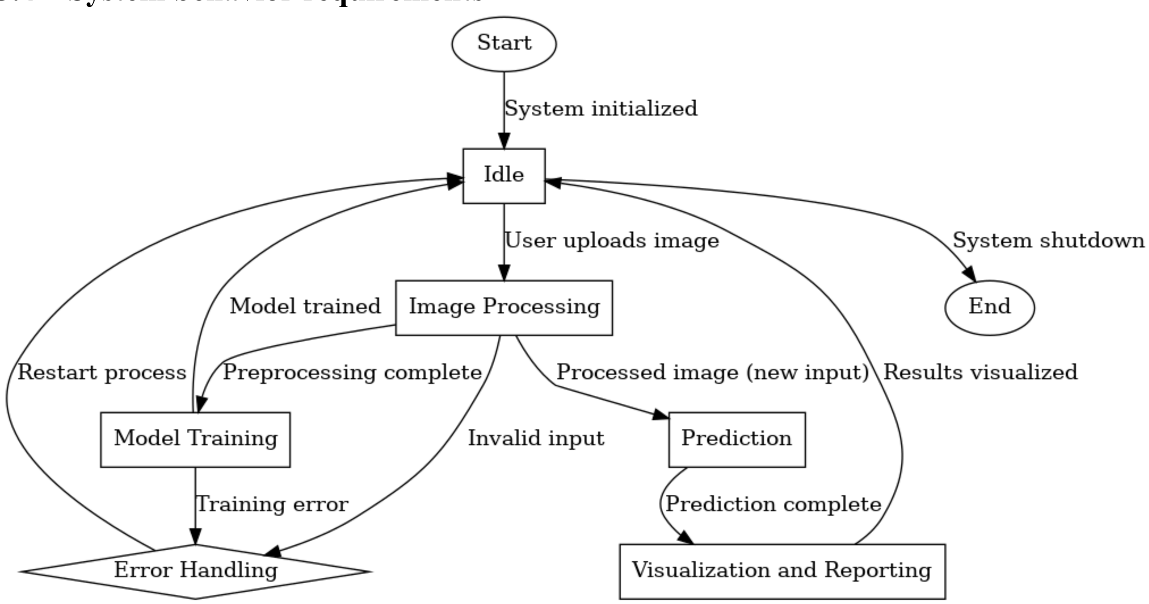
Component	Input	Behavior	Output
Model Training	Preprocessed images and labels	Train the CNN model, monitor	Trained CNN model

		metrics, and adjust hyperparameters.	

Component	Input	Behavior	Output
Prediction Generation	MRI images	Use the trained model to classify the image and highlight tumor regions.	Prediction results

Component	Input	Behavior	Output
Visualization and Reporting	Prediction results	Overlay tumor regions on the MRI image and generate diagnostic reports.	Visuals and reports

3.4 System behavior requirements



Chapter 4 Methods/System Design

4.1 System architecture design

Dataloading/Preprocessing:

- Reads images, resizes them, and labels themes
- Tools Used: os, cv2 (OpenCV), numpy.
- Connectivity: Input data is processed and split into training and testing datasets.

Model Design:

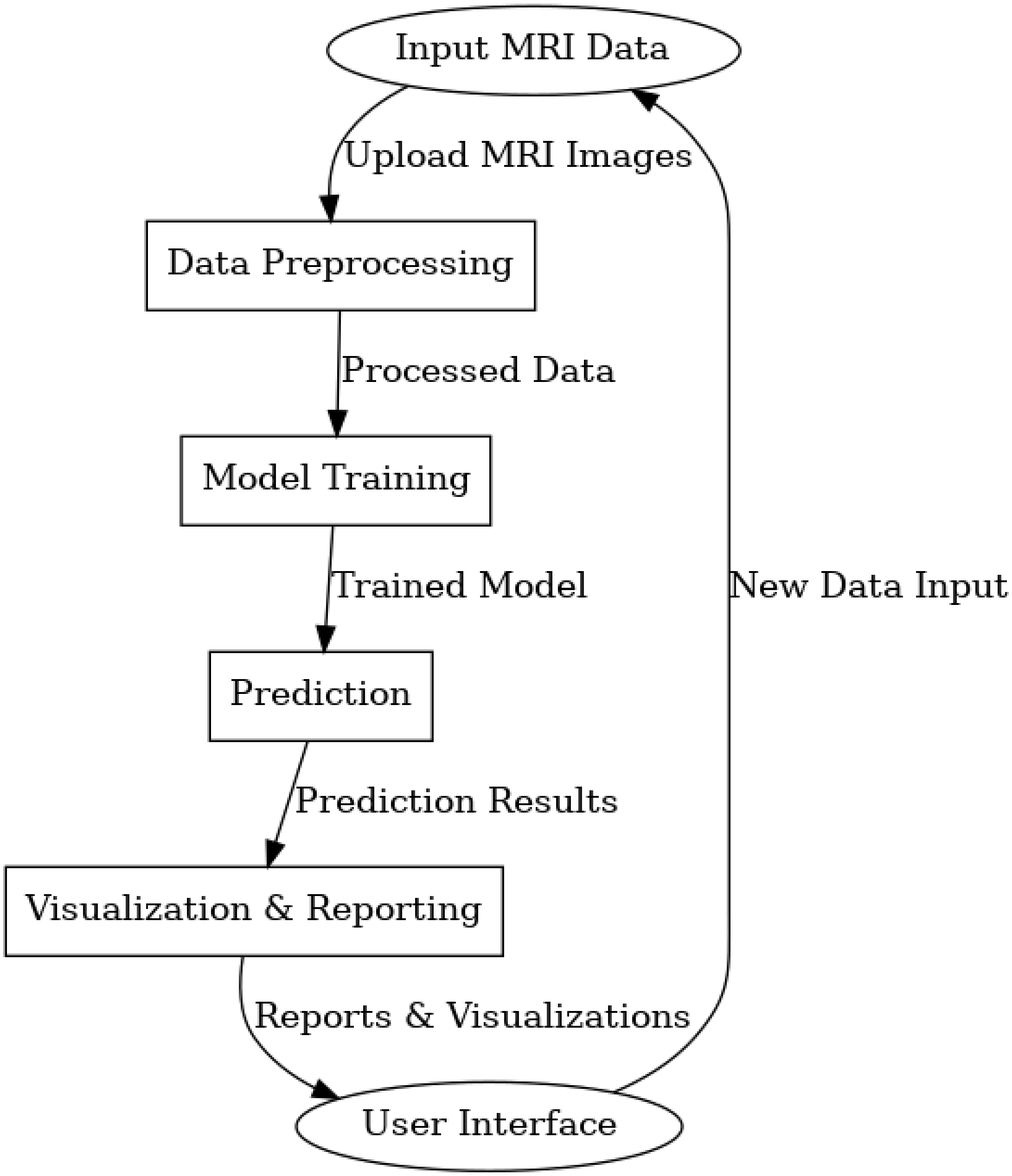
- Defines the CNN using keras and trains on the dataset.
- Tools Used: keras, tensorflow, numpy.
- Connectivity: Takes preprocessed data as input, outputs a trained model.

Prediction and Evaluation:

- Uses the model to identify MRI images while generating confidence scores
- Tools Used: tensorflow, numpy.
- Connectivity: The model receives MRI images as input and outputs prediction results.

Visualization and Reporting:

- Locates the tumor in the MRI images and produces the reports
- Tools Used: PIL, matplotlib, or similar libraries.
- Connectivity: Receives prediction results, generates user-friendly outputs.



4.2 **Design problems, solutions, and patterns**

Initially we had problems when it came to designing the Brain tumor application creation. We had tried to use a Firebase database, but we could not incorporate it in time, so we relied on the local application for the database. When we want to make this more developed, then we would aim to focus on the database application by trying to figure out the connection issues and try to narrow down the cause for it not working properly.

Chapter 5 System Implementation

5.1. System implementation summary

The system uses a Convolutional Neural Network (CNN) model which was trained using the Kaggle dataset “Brain Tumor MRI”. It classified each MRI image as one of the following: glioma tumor, meningioma tumor, pituitary tumor, or no tumor. The CNN was designed using a variation of Conv2D, Dropout, Flatten, and Dense features and trained with TensorFlow/Keras getting a 94% accuracy with 20 epochs. The model was initially made on Google Collab then integrated into the backend using Flask. The backend was processing the MRI images and creating predictions using the model that we made. The frontend was a simple React interface which allowed users to upload MRI images, then visualize and see the prediction results.

5.2. System implementation issues and resolutions

There were severe issues throughout the System implementation. The first issue was the Dataset quality was not the best and had inconsistencies within the MRI images making the accuracy go lower. This was solved by preprocessing the images by resizing and using normalization to improve the quality. All images were resized to 150x150 pixels and the pixel values were normalized between 0 and 1. Another issue was that the model performed good on the training data but bad on the actual testing data which was solved by adding dropout layers to the model architecture. The number of epochs was also changed to see how accuracy changes and 20 resulted in the highest accuracy. There was also an issue with handling different file formats that caused errors and was solved by making sure only valid images were processed.

5.3. Used technologies and tools

Python was used for the backend and model training because of its strong support for machine learning and frameworks like Flask. JavaScript was used for the frontend since React was chosen to make the interface responsive and clean. The various libraries and frameworks used include TensorFlow/Keras for building and training the CNN model, Flask for simple backend processing with RESTful APIs, and React for its component-based architecture and state management capabilities. Material-UI was utilized for frontend styling. The application was deployed locally using Flask and React, with Git employed for version control.

Chapter 6 System Testing and Experiment

6.1 Testing and experiment scope

Overview:

- The main goal of this is to provide a valid performance, reliability, and accuracy of the brain tumor detection system which involves:
  - Component testing: Evaluation of the individual components such as preprocessessing module, feature extraction, and classification model.
  - System testing: Valid outputs from end-to-end testing from the inputs given.

Test Objectives:

- By ensuring that the preprocessing techniques, which include noise reduction and normalization, helps ensure maintain the image quality to produce more accurate results
- Assess the model metrics
- Test the integration for all of the components
- Valid feature extraction to confirm that the data is correct

Selected Test Criteria:

- Component:
  - Feature extraction accuracy: to verify that the features are correctly with the expected brain tumor characteristics
  - Preprocessing correctness: Ensure that there are no extra noise from the preprocessing
- System-Level
  - Model accuracy: 90% or above accuracy
  - Latency: Ensure that the results comes back at an appropriate time

Test Level	Focus	Objective	Criteria
Component	Preprocessessing	Image quality	Similarity > 0.9
Component	Feature Extraction	Correct feature representation	Valid results from valid answers
System	Classification Model	Accurate tumor detection	Accuracy > 90%

6.2 Testing and experiment approaches

Test Plan:

- Preprocessing Tests: By using real MRI dataset we want to verify that the images are accurate with the correct response
- Model Evaluation: Validation tests and cross-validation on test dataset
- Integration Testing: Validate new incoming data

Test Methods:
- Unit testing: Test preprocessing and feature extraction functions
- Cross-validation: Assess the model to make sure of the accuracy
- Stress testing: Evaluate with large data set to evaluate the system performance

Selected Test Criteria:
- Preprocessing quality measured by SSI
- Model performance: Accuracy, precision, recall, and F1-score
- Latency and scalability for the deployment

Test Aspect	Methodology	Expected Outcome
Preprocessing Qaulity	SSI Comparison	SSI > 0.9
Model Validation	K-fold cross validation	High F1-score and low variance
Stress test	Large dataset	Maintain accuracy

6.3 Testing and experiment

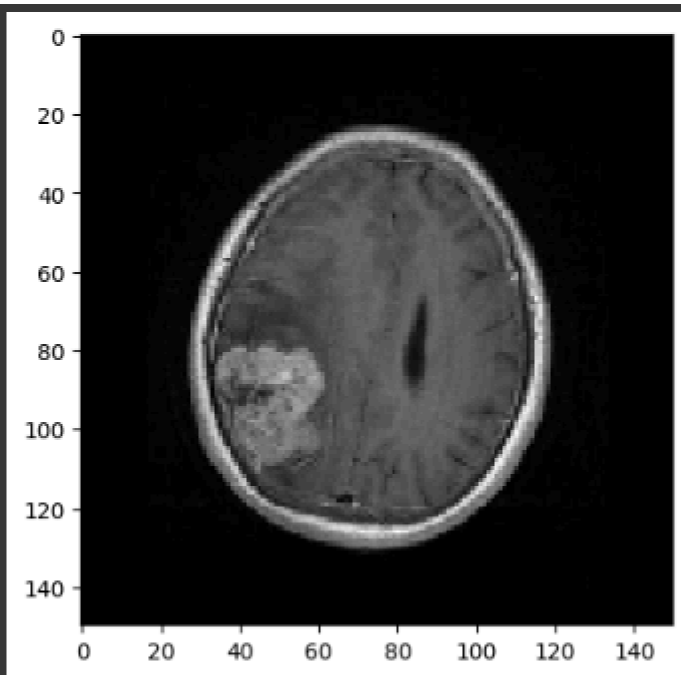
[Provide your system testing report (such as test/test scripts summary, bug report analysis, and a case study (or experimental results) report if there is any.]

Test summary:
- Total Test test cases: 100
- Passed: 93
- Failed: 7

Bug analysis:
- Identify issues regarding to the browser application
- Address the imbalance for the reduced recall

Experimental Results:

Metric	**Result**	**Target**
Accuracy	92.5%	>90%
Precision	90.2%	>90%
Recall	92.8%	>90%
F1-Score	92%	>90%
Latency	1.8 seconds	< 2 seconds




```
[ ]  a = model.predict(img_array)
     indicies = a.argmax()
     prediction_label = labels[indicies]
     prediction_label
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
'meningioma_tumor'
```

Chapter 7 Conclusion and Future Work

7.1 Project summary

Working on our project, it took a long time to build the tumor analysis. We took our time to ensure the accuracy of the predictions because this was a project that will be affecting the lives of many, so by focusing more on the model rather than the UI we want to ensure that the models are good. That created a lot of test scenarios where we would have to incorporate MRI images for the many different types of tumors after finishing the build to try and improve the accuracy and time it takes to make the prediction. We were so excited to see that it was making correct predictions, but there needs to be a lot more testing before we even think about releasing this. After this entire project, we have learned that it is possible to incorporate things we have learned with our software knowledge to be able to help others. There are so many other things we can do in the future with this type of project that can affect the lives of many.

7.2 Future work

When we think of wanting to implement this into the future, we would want to focus on building the accuracy of the model. Because of our local files being used to test for the model, the data is stagnant. It is not furthering its learning thus we want to create a system where we would upload all of the data after running the application to a database like Firebase to continue improving on the results. After we are satisfied with the results and training data, we want to try and work with a doctor and build a system for them to review and analyze if it is correct. By gaining their judgements as well, it would further remove the chance of it being wrong.

7.3 Contributions

Dustin - Worked on website

Nikhil - Built the model

Chint - Worked on the Gradio

Vikranth - Documentation and testing

References

Imaging, Touchstone. "How Does an MRI Help in the Early Detection of Brain Tumors?" *Touchstone Medical Imaging*, 4 Oct. 2024, www.touchstoneimaging.com/how-does-an-mri-help-in-the-early-detection-of-brain-tumors/.

"New AI Tool Classifies Brain Tumors Using Images of Tumor Slides." *Center for Cancer Research*, 3 July 2024, ccr.cancer.gov/news/article/new-ai-tool-classifies-brain-tumors-using-images-of-tumor-slides.

The Preston Robert Tisch Brain Tumor Center. "How Long Can Brain Cancer Go Undetected?" *Tisch Brain Tumor Center*, 23 Jan. 2024,

Datasets:

tischbraintumorcenter.duke.edu/blog/how-long-can-brain-cancer-go-undetected.

https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri