System Description

- *Purpose of System*
  Mediviz is an application that allows medical professionals to identify areas of the brain that are at high risk for Posterior Fossa Syndrome, a behavior deficit caused by the presence of lesions.

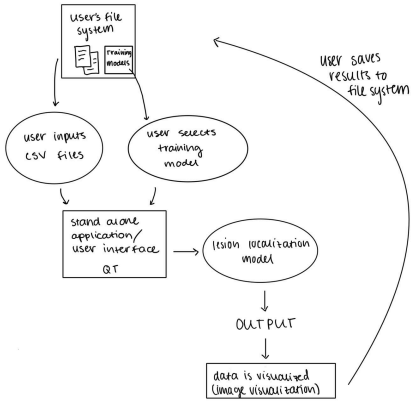
- *Basic overview of functionality of the system*
  Given a patient's brain scan and diagnosis, the system aims to identify correlations between patient diagnoses and the location of their lesions. The system prompts users to upload patient brain scans and select the region of interest. Areas of high risk are identified and visualized on the patient's brain scan. The resulting image can be saved directly to the user's file system, allowing for streamlined accessibility.

Architectural Review

- *Describe design/architecture of system*
  The system first processes the data to ensure that it is in the correct format. After ensuring correct data input, it is inserted into a machine learning model for parameter optimization and lesion behavior analysis. The results from the machine learning algorithm are then inserted into an image visualization tool that provides various details on the results.

- *List of major system components*
  - Data Preprocessing Tool
  - Image Processor
  - Lesion Localization Model
  - Image Visualization Tool

- *System diagram*



Four Epics

Epic 01-Store & Access Data

1. As a user of MediViz,
   I want to store and access my previous data
   so that I can decrease future computation time.

Epic 02-Customizable Algorithm for Better Control

2. As a user of MediViz,
   I want a customizable interface (parameters, targeted region, etc)
   so that I have more control over the algorithm's target

Epic 03-Visualize Program Output

3. As a user of MediViz,
   I want a visualization of the program output
   so that I can analyze areas of high risk and lesion placement.

Epic 04 - Systematic Occlusion

4. As a user of MediViz,
   I want to have the option to utilize systematic occlusion
   so that I can isolate particular voxels for the algorithm to focus on/ignore.

Non-functional Requirements

- The machine learning algorithm should produce output in 10 minutes or less.
- The program should support file sizes up to 1 MB
- The system should be intuitive and easy to use

Technologies and Frameworks

- Python
- SkiLearn (SVM)
- Pandas
- PlotLib

Minimum Viable Product (MVP)

- Basic user interface/upload or generate point for user data
- Machine learning algorithm that provides weights
- Output of voxel weights as a matrix
- Visualization of machine learning algorithm accuracy

The user is prompted to select a file path containing patient data in the form of excel files. The ML algorithm runs and applies weights to each voxel (region) in the matrix. The app then provides a table showing the precision and accuracy of the ML algorithm for each classification (whether the patient has the diagnosis or not).

Preliminary Roadmap

Sprint 0 Feb 6th):

- Solidify project planning
- Research packages needed

Sprint 1 (Feb 20th):

- Solidify brain behavior data format
- Implement generation of brain behavior data
- Begin creation of user interface
    - Create an upload point for users to upload data from their own file system

Sprint 2 (Mar. 6th):

- Implement running of ML algorithm
- Implement storage of ML model into the file system
- Achieve minimum viable product
- Meet with Dr.McAfee

Sprint 3 (Mar. 20th):

- Implement selection of saved models
- Implement running of saved models

Sprint 4 (April 3rd):

- Implement results visualization dashboard
- Implement results dashboard export into a pdf
- Meet with Dr.McAfee

Sprint 5 (April 15th):

- Implement systematic occlusion

Sprint 6 (April 29th):

- Review finalized product with Dr.McAfee (and invite him to the presentation ofc! 🙂)