Project Title: Federated Learning of Medical Image Reconstruction

Names and Email Addresses of Team Members:

- Joshua Sheldon (jsheldon2022@my.fit.edu)
- Izzy MacDonald (imacdonald2022@my.fit.edu)
- Tanuj Kancharla (tkancharla2022@my.fit.edu)
- Yash Jani (yjani2023@my.fit.edu)

Faculty Advisor & Client: Debasis Mitra (dmitra@fit.edu)

Meeting(s) with the Client for Developing this Plan:

- August 20, 2025 @ 11 AM

Goal and Motivation: "Single photon emission computed tomography (SPECT) is a nuclear imaging technique using gamma rays" (University of Utah). A SPECT scan is begun by injecting a patient with a gamma ray emitting pharmaceutical (a tracer). The patient then lies down on a table in a scanning room equipped with a gamma camera, which uses a collimator instead of a lens and creates images by detecting radioactivity instead of light. These images are monochromatic images, where brightness in any given pixel of the image is determined by the tracer detection count at that position on the collimator's surface.

The gamma camera is rotated around the patient, capturing projections of a portion of the patient's body at different angles, creating a unique type of image called a sinogram, "where the horizontal axis represents the count location on the detector [gamma camera], and the vertical axis corresponds to the angular position of the detector" (Philippe P. Bruyant, PhD).

These sinograms are then reconstructed, traditionally using analytic and iterative algorithms, to create a medical image easily interpretable by medical professionals. However, these algorithms are slow. The "Tomographic Medical Image Reconstruction using Deep Learning" project attempts to develop a machine learning model that can perform the exact reconstruction much quicker than traditional methods. The project aims to train this model on synthetically generated data.

However, the accuracy of the project's model is limited by the lack of variety in the training data and the exclusive use of synthetic data. This project aims to improve the accuracy of this existing machine learning model by addressing both concerns.

Approach: We intend to improve the accuracy of the existing machine learning model through two approaches. Additionally, two parties are involved in the implementation of our approach:

- Learning Managers - Individuals, likely involved with this project or the client's lab, who are working to develop this machine learning model through traditional and federated learning.
- Medical Professionals - Individuals who are conducting SPECT scans and are willing to use gathered data to contribute to the machine learning model.

Our first approach for improving accuracy is modifying the data synthesis pipeline to make synthetic data more realistic. The previous data synthesis pipeline had many issues, including a

low number of artificial human body images, incorrect physics for the simulated SPECT scans, incorrect data dimensionality, and the use of augmentation strategies that destroyed the quality of our synthetic data. Additionally, the current dataset does not include images from patients with infractions or ailments in the heart and other organs.

Last spring, we introduced the ability to generate synthetic data representing patients with heart infractions. Over the summer, we amended these issues with the data synthesis pipeline. Unfortunately, this means we're starting from scratch with our training data. However, we're mitigating this by (1) introducing a new augmentation step which adds noise to the sinogram (input data) and (2) using GATE 10 as our physics simulator, reducing the time it takes to create synthetic sinograms (by far the lengthiest step of data synthesis) by up to 4x, depending on the machine. Based on our current calculations, we hope to generate roughly ~1,000 samples every two weeks.

We are modifying the machine learning pipeline from the previous project to train a derivative of the original model with this new, highly realistic synthetic data. This machine learning pipeline is both improved and used by the learning managers.

Machine Learning Pipeline

- Learning managers can use the data generation pipeline to generate synthetic, realistic human body images and simulated sinograms of that body.
- Learning managers can specify for the data generation pipeline to include anomalies like heart infractions in the generated data.
- Learning managers can supply generated data to train a machine learning model that reconstructs medical images from sinograms.
- Learning managers can apply this machine learning model to reconstruct medical images from sinograms.

Our second approach to improving accuracy is developing an application set that enables the model to be trained with federated learning. Federated learning is a machine learning technique that allows a model to be trained using data from third-party contributors without those contributors ever having to share the raw data.

This technique is realized through two different applications: an orchestrator application, which facilitates the learning, and a contributor application, which allows the use of relevant data to contribute to the learning.

The orchestrator application connects with a set of contributor applications, and distributes an initial machine learning model to them. The contributor applications use local data to train this model, and then send their models back to the orchestrator application. The orchestrator application averages the parameters of all the models, creating a new initial machine learning model, which is again distributed to the contributor applications, restarting the cycle.

In our project, the learning managers would run and manage the orchestrator application. The medical professionals would run and use the contributor applications to train the machine learning model using medical data from their institutions.

This would allow medical institutions to contribute to the model with real data without violating patient privacy regulations. We hope introducing real training data will raise the model's accuracy to a level unattainable with synthetic data.

Orchestrator Application

- Learning managers can interact with the federated learning orchestrator application through a user interface.
- Learning managers can supply an initial model for federated learning or randomly initialize one if no training has been done.
- Learning managers can define trusted contributors to the federated learning.
- Learning managers can see which contributors have new data for training.
- Learning managers can start a round of training and select which contributors will participate (likely educated by which contributors have new data for training).
- Learning managers can send an initial model to all contributors in the training round.
- Learning managers can receive trained models from contributors.
- Learning managers can combine the contributor’s models to create a new initial model.
- Learning managers can rename and delete models produced by federated learning rounds.

Contributor Application

- Medical professionals can interact with the federated learning contributor application through a user interface.
- Medical professionals can connect to a federated learning orchestrator application.
- Medical professionals can submit training data to the application (which notifies the orchestrator application).
- Medical professionals can agree to participate in training rounds initiated by the orchestrator application.
- Medical professionals can receive initial models from the orchestrator application.
- Medical professionals can allow the contributor application to train the initial model distributed by the orchestrator application using submitted training data.
- Medical professionals can allow the contributor application to send their trained model back to the orchestrator application.

Algorithms and Tools:

- The data synthesis pipeline is primarily composed of Python and Bash scripts. We use the XCAT Phantom Program, XCAT+ (by BiCLab), GATE, and a myriad of Python libraries.
- The machine learning model is implemented using PyTorch.
- The federated learning applications are composed of Python backends and React frontends. We use the Flower framework for our federated learning operations and Flask for interapplication and frontend-backend communications.

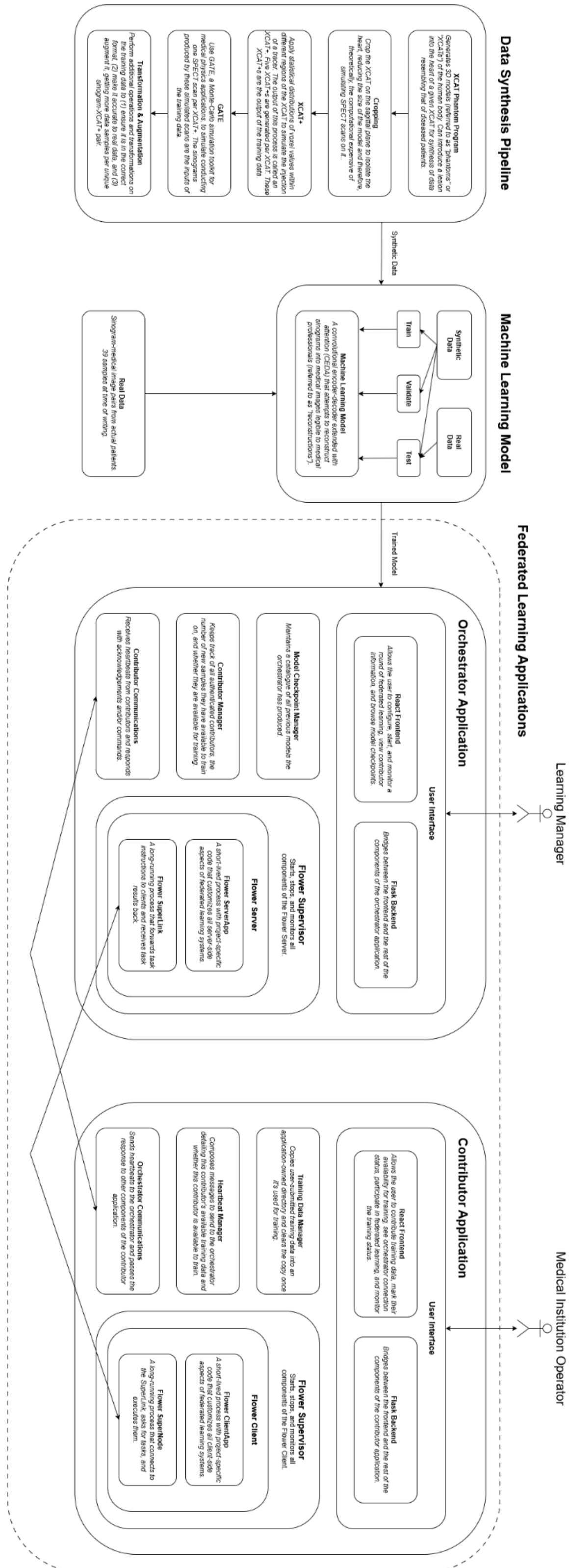
Novel Features: To the best of our knowledge, training a machine learning model to reconstruct medical images from SPECT scans, using synthetic data **with artificially introduced heart infractions**, has never been done before.

The second branch of our approach, applying federated learning in a medical context, is not novel, having been realized by projects such as MELLODDY. This aspect of our approach is largely focused on using proven techniques in a new problem space.

Technical Challenges: Our primary technical challenges this semester revolve around integration:

- Finding a way to have our frontends respond when there's changes to the return value of an endpoint in the backend.
- Having our federated learning applications successfully launch, observe, and shutdown completely separate applications from the Flower framework in a manner that is elegant and fault tolerant.
- Integrating the highly disparate components of our system that we're working on (machine learning model, federated learning applications, Flower framework code, etc.) into one reliable system.

Design:


Data Synthesis Pipeline
XCAT Phantom Program
Cropping
XCAT+
GATE
Transformation & Augmentation
Synthetic Data
Machine Learning Model
Synthetic Data
Real Data
Train
Validate
Test
Machine Learning Model
Real Data
Trained Model
Federated Learning Applications
Learning Manager
Medical Institution Operator
Orchestrator Application
User Interface
React Frontend
Flask Backend
Model Checkpoint Manager
Contributor Manager
Contributor Communications
Flower Supervisor
Flower Server
Flower ServerApp
Flower SuperLink
Contributor Application
User Interface
React Frontend
Flask Backend
Training Data Manager
Heartbeat Manager
Orchestrator Communications
Flower Supervisor
Flower Client
Flower ClientApp
Flower SuperNode

Evaluation: The main metrics of our project are reliability (factoring in number of participants) and machine learning model performance.

- We should reliably (>=90% success rate on any given round) be able to perform federated learning with our applications.
- That reliability should hold when performing federated learning with one orchestrator and four contributors.
- The accuracy of the machine learning model produced by our applications should be greater than the accuracy of a machine learning model trained on any one contributor's data.

Progress Summary:

Module/Feature	**Completion %**	**To-Do**
Data Synthesis Pipeline	90%	Check if GATE 10 can be installed on AI.Panther, and if so, deploy the data synthesis pipeline on AI.Panther.
Machine Learning Model	80%	Tweak the model to work with new data dimensionality, continually fine-tune for higher accuracy.
(FL Apps) Daemons	80%	Fix secure communications, integrate Flower framework.
(FL Apps) Frontends	40%	Integrate orchestrator frontend and backend, create contributor frontend.
(FL Apps) Flower Framework Code	0%	Implement.

Milestone 4 (Sep. 29):

- Begin implementing and testing Flower framework code.
- Implement, test, and demo contributor application UI.
- Implement, test, and demo new ML model dimensionality.
- Implement, test, and demo the connection between the orchestrator application's frontend and backend.
- Implement, test, and demo secure communications between the orchestrator and contributor applications.
- Implement, test, and demo data synthesis using AI.Panther.

Milestone 5 (Oct. 27):

- Finish the implementation and testing of, then demo the Flower framework code.
- Implement, test, and demo the integration between the Flower framework code and the federated learning applications.
- Implement, test, and demo improvements to machine learning model accuracy.

- Implement, test, and demo the connection between the contributor application's frontend and backend.
- Create poster for Senior Design Showcase.

Milestone 6 (Nov. 24):

- Perform a full system test and demo.
- Compare full system performance to single-client performance and write report containing results and analysis.
- Create user/developer manual.
- Create demo video.

Task Matrix for Milestone 4:

Task	**Joshua**	**Izzy**	**Tanuj**	**Yash**
Begin writing Flower framework code.	0%	0%	0%	100%
Develop a UI for the contributor application.	0%	100%	0%	0%
Update the ML model to reflect data dimensionality.	0%	0%	100%	0%
Connect orchestrator application frontend and backend.	50%	50%	0%	0%
Fix FL daemon secure communications.	100%	0%	0%	0%
Attempt to leverage AI.Panther for data synthesis.	100%	0%	0%	0%

Description of Planned Tasks for Milestone 4:

- **Begin writing Flower framework code:** We are using the Flower framework for our federated learning operations. To use the Flower framework, you have to write code to instruct its applications (the ServerApp for the orchestrator and the ClientApp for the contributor) how to conduct the federated learning process (ex. Which model to use, how to load data, what percentage of client should participate, etc.)
- **Develop a UI for the contributor application:** In the first semester, we successfully developed a UI for the orchestrator application. Now, we need to do the same for the contributor application. This means writing the React code to outline the structure of the page and the basic elements, using dummy data. This does not mean making the frontend functional for interacting with the application.
- **Update the ML model to reflect data dimensionality:** Over the summer, we made many modifications to the data synthesis pipeline. This resulted in our model's training data having different dimensions, to more accurately reflect real data while keeping physics simulation run times low. The ML model was developed by the "Tomographic Medical Image Reconstruction using Deep Learning" team, and uses input and output dimensionality, and therefore a model structure, which reflects the dimensionality of the data they generated. We need to update the model to reflect the dimensionality of the data we are generating.

- **Connect orchestrator application frontend and backend:** We must update the orchestrator application frontend to use the API exposed by the daemon, to populate the frontend with real data and let it be used to interface with the application. Additionally, we must modify the application so that when the backend is launched, the frontend launches as well.
- **Fix FL daemon secure communications:** We run into issues when trying to use certificate-key based authentication for communication between the orchestrator and its contributors. We must fix this issue to enable secure communication. Additionally, we must add the ability for the orchestrator to authenticate contributors based on their certificates.
- **Attempt to leverage AI.Panther for data synthesis:** During semester 1, we developed a script to automatically install GATE and all of its required prerequisites on a Linux machine without sudo access. However, while this script previously worked on AI.Panther, it now seems to be broken. We must (1) update the script to install GATE 10 instead of GATE 9 (we switched versions over the summer) and (2) fix the script on AI.Panther.

Approval from Faculty Advisor:

Signature Date

_____ _____