Milestone 2 Evaluation — Covid-19 Detection

1. Project Title, Team Members

Project Title: Covid-19 Detection

- Drew Quashie (dquashie2024@my.fit.edu)
- Amanda Ogbonna (kogbonna2025@my.fit.edu)
- Loleyi Oluwatomisin (ooluwatomisi2023@my.fit.edu)
- Richard Alonso Garcia (ralonsogarci2023@my.fit.edu)

2. Faculty Advisor

Name: Dr. Zahra Nematzadeh **Email:** znematzadeh@fit.edu

3. Client

Client is Faculty Advisor; Dr. Zahra Nematzadeh (znematzadeh@fit.edu) — Faculty Advisor, Florida Tech

4. Progress of current Milestone (progress matrix)

Task	**Completion %**	**Drew Quashie**	**Amanda Ogbonna**	**Loleyi Oluwatomisin**	**Richard Alonso Garcia**	**To do**
Train & Fine-Tune CNN Models & Transfer Learning Architectures	85%	x	x	x		Retune cascade thresholds on the validation set; improve stage-1 sick recall
Initial Model Evaluation & Performance Reports	85%	x		x		Re-run cascade eval at retuned thresholds; feed operating points into confidence-gate calibration
Develop FastAPI Backend Endpoints	90%	x	x			Wire real model artifacts into gate and inference stubs
Build React Web UI Components for Prediction Results and Risk Feedback	70%				x	Results screen, inconclusive variant, integration with /evaluate

5. Discussion (at least a paragraph) of each accomplished task (and obstacles) for the current Milestone

Train & Fine-Tune CNN Models & Transfer Learning Architectures

The first approach was a single 3-class CNN trained directly on the three labels (healthy, symptomatic, COVID-19), with joint MFCC + log-Mel features, and three input variants tested (waveform only, spectrogram only, both). Every variant produced the same result: the model collapsed to the majority class. The validation F1 sat at 0.2459 for all 60 epochs in each run while accuracy stayed at 58.43%, which is exactly the healthy-class share of the dataset (31,522 of 42,364 recordings, 74.4%). Training loss decreased while validation loss did not, so the runs were memorizing the training set rather than learning anything transferable.

Selecting augmentation methods also turned out to be harder than expected because of how unusual cough audio is as a signal: standard audio augmentations (time stretch, pitch shift, noise) can distort the acoustic features that distinguish a sick cough from a healthy one, so aggressive augmentation pushed the model toward the majority-class answer instead of helping it generalize.

The collapse had a second cause we found and fixed: the original random split let recordings from the same speaker appear in train, validation, and test sets. The literature warned us about exactly this; Buttow et al. (2021, npj Digital Medicine) showed AUC dropping from 0.90 to 0.71 when a cough model moves from a random sample-level split to a participant-independent one. We rebuilt the split to be recording-wise (implemented 2026-10-23), so no speaker crosses between sets.

The larger pivot was architectural. The direct 3-class setup asks one model to learn the COVID-vs-symptomatic boundary, which is the one boundary the literature says is not learnable from cough audio: Orlandic et al. (2023, Computer Methods and Programs in Biomedicine) re-reviewed 581 "symptomatic" CoughVid recordings with physicians and found only 32 were actually COVID-positive, meaning the symptomatic class is defined by self-reported symptoms rather than acoustics. We restructured into a two-model cascade: model 1 is binary, (COVID + symptomatic) vs healthy, with the large healthy majority stabilizing training; model 2 is binary, COVID vs symptomatic, run only on recordings model 1 flags as sick. Their combined outputs keep the 3-class output the app requires. Training uses a class-weighted loss for imbalance (the method Chang et al. 2022 used on CoughVid), and the recording-wise split applies to both models. Both cascade stages are now fully trained for the default CNN, the dual-branch variant, and ResNet50 transfer learning (see the evaluation section below for the results).

Initial Model Evaluation & Performance Reports

Evaluation is currently concentrated on the first stage of the cascade. The early results confirm the pivot did its job: the model no longer leans entirely on the largest class. Where the direct 3-class runs sat frozen at 58.43% accuracy and 0.2459 F1, the (COVID+symptomatic)-vs-healthy binary model reaches an F1 above 0.51 with validation AUC around 0.56 and PR-AUC around 0.29, and the metrics move epoch to epoch instead of flatlining. Absolute numbers are still modest, but that matches the literature baseline: participant-independent splits on CoughVid-style data land near AUC 0.70 at best, and the 90%+ figures published with random splits are leakage artifacts. This is why we report per-model AUC and PR-AUC alongside F1.

The full training and evaluation runs are now complete across three architectures (the default custom CNN, the dual-branch waveform + spectrogram variant, and ResNet50 transfer learning), each trained on both cascade stages with the class-weighted loss and early stopping on validation loss. Stage 1 (sick vs healthy) converges to essentially the same place in all three: final validation AUC 0.55–0.57 and validation accuracy 52–68% that oscillates with the class prior, confirming the stage-1 models are still mostly learning the healthy/sick prior rather than strong acoustic signal. ResNet50 is the clearest failure mode of the three: it overfits almost immediately (train loss falls from 0.13 to 0.04 within 20 epochs while validation loss climbs from 0.13 to 0.46), and early stopping caught it at epoch 23.

Stage 2 (COVID vs symptomatic), evaluated standalone on the held-out test split, carries the most signal, and the architecture comparison now has a clear winner. The branched variant reaches test ROC-AUC 0.637 (PR-AUC 0.26, COVID recall 0.61 at its operating threshold), beating the default CNN (0.599 / 0.22 / recall 0.47) and ResNet50 (0.584 / 0.21 / recall 0.50). The branched stage-2 confusion matrix at a 0.5 threshold catches 150 of 186 COVID recordings (81% recall), at the cost of flagging 611 of 954 symptomatic ones as COVID — the error is asymmetric in the clinically safer direction, since COVID is the class you do not want to miss. This is a small separation above chance on a task the literature says is barely learnable from cough audio, and it validates the transfer learning comparison: the pretrained backbone underperforms the purpose-built dual-branch CNN on this data.

The assembled 3-class cascade is still bottlenecked by stage 1, and the new per-class confusion matrices quantify exactly by how much. On the combined test set, healthy recall is 0.84–0.87 across all three architectures, but COVID recall lands at only 0.03 (default), 0.07 (branched), and 0.03 (ResNet50): stage 1 still routes most COVID recordings to "healthy" before stage 2 ever runs, so the stage-2 improvement cannot express itself in the end-to-end numbers. The cascade evaluation also produced its full per-class report set — confusion matrices, ROC/PR test curves, training-history plots, and CSVs for the training summary, stage-2 standalone metrics, and assembled-cascade recalls — written to the figures directory and notebook root, so the performance reports feeding the confidence-gate calibration now exist in full.

The plan going forward follows directly from this decomposition: retune the stage-1 sick threshold (and stage-2 COVID threshold) on the validation set rather than the defaults, since trading some healthy recall for stage-1 sick recall is the single change with the most leverage on end-to-end COVID recall; report the branched stage-2 standalone result as the headline with the cascade reported as stage-1-recall-limited; and lead the reports with PR-AUC and per-class recall instead of accuracy, which is close to meaningless at this class balance.

The obstacle that dominated evaluation was the leakage problem described above: before the recording-wise split, our test numbers were not measuring anything meaningful, since the model could recognize speakers rather than cough characteristics. The second evaluation obstacle is the COVID-vs-symptomatic confusion itself, which stage 2 of the cascade now owns in isolation; its PR-AUC of ~0.26 shows how little headroom exists above the class prior even for the best architecture. The remaining work for this milestone is retuning the cascade thresholds on the validation set and re-running the assembled evaluation at the new operating points, then handing the calibrated thresholds to the confidence gate in the backend.

Develop FastAPI Backend Endpoints

The backend is a FastAPI app with two routers. Auth: `POST /signup`, `POST /login` (bearer token), and `GET /me`. Evaluation: `POST /evaluate` takes a multipart audio upload from an authenticated user, resamples it to 16 kHz mono WAV, runs the XGBoost cough gate first (rejection is a 422), then the PyTorch model, and stores the result in SQLite; `GET /evaluations`, `GET /evaluations/{id}`, and `GET /export` serve the stored history back. Models load lazily from environment-overridable paths, so the API layer never hardcodes model locations. A framework-free smoke test (`test_smoke.py`) verifies the whole set against a live server: signup, login, wrong-password rejection, upload to `/evaluate`, history listing, export, and auth enforcement. All smoke tests pass.

Two pieces are deliberately stubbed: the XGBoost gate scoring and the PyTorch inference call return placeholder values because the model artifacts are not yet exported from the training pipeline; the HTTP contract, validation, gate ordering, and persistence are real and slot the models in behind the same function signatures. One deviation from the design document: results are currently stored server-side per user account, while the design targets stateless inference with browser-side storage (SNFR2); this was the intermediate step that made the endpoints testable end-to-end, and it gets resolved when persistence moves client-side in Milestone 3.

Build React Web UI Components for Prediction Results and Risk Feedback

The dashboard was completed as a single React page composed of independent card components that read from one shared record schema. Each result record displays a risk level, a confidence score, the three per cough votes, per class signal strengths, and recording quality measurements. The page uses a twelve column bento grid containing a latest result hero card,

a confidence trend chart with selectable time ranges, a vote breakdown, a recording quality summary, a filterable and sortable history list, an activity summary, and a data controls card for CSV and JSON export and deletion. Selecting a history row opens a detail dialog, and the risk badge, advice text, and status tags are driven by a single risk level lookup so the wording stays consistent across the page. The recorder page was restyled with the dashboard's design new gradient frame. Its three cough indicator and live dB meter were placed on the same gradient hero card. The two pages are connected through hash routing. Each "Start cough test" entry point on the dashboard routes to the recorder, and the recorder routes back through the sidebar, a back button, and a "View dashboard" button shown after a completed capture.

One obstacle that we had developing the dashboard was the recorder's logic when it toggles its Stop and Try again buttons with the HTML hidden attribute. The new button styles set an explicit display value, which overrides that attribute, so hidden buttons would have rendered in every phase. This was solved with a style rule that forces [hidden] elements to remain hidden. Some lifecycle defects were also corrected during development such as the capture logic's click handlers were never removed, which allowed duplicate microphone sessions under react's development double mount, and the microphone stream could remain open if the user left the page during a test or while the permission prompt was pending. The capture pipeline now returns a cleanup function that removes its listeners and releases the audio stream.

6. Discussion (at least a paragraph) of contribution of each team member to the current Milestone

Drew Quashie

I revamped the training pipeline for the two-model cascade approach: model 1 on (COVID + symptomatic) vs healthy and model 2 on COVID vs symptomatic, with Amanda and Loleyi helping me create the new subsets that split the dataset for the two binary cascade tasks. I also managed the experiment run for the augmentations, configuring the per-class augmentation probabilities (augmenting COVID-19 samples 90% of the time, symptomatic 70%, and healthy only 40%, so the minority classes get the most exposure) and running the waveform-vs-spectrogram input comparison (wave only, spec only, and both branches) across full 60-epoch training runs. On top of that I started making the code for the mass training runs across both cascade models and their input variants, and I redid the base convolution block with the adaptive version.

Amanda Ogbonna

For this milestone, I worked on expanding the research portion of our project based on feedback from our faculty advisor. I went back through the papers we collected and added more detailed information about which algorithms performed the best, the limitations of each study, dataset sizes, whether the datasets were balanced or imbalanced, and what techniques were used to handle class imbalance. I also added additional research related to deep learning and audio classification to help us better understand which architectures and techniques could be useful

for our own model. Through this research, I identified limitations with our original three-class classification approach and recommended changing it to a two-class (binary) formulation. I helped write the code that restructures the dataset into the two binary tasks: the stage-1 (COVID + symptomatic) vs healthy grouping and the stage-2 COVID vs symptomatic subset.

In addition to the research, I worked on implementing our ResNet50 transfer learning model using PyTorch. I implemented the model architecture using pretrained ResNet50 weights and added the option to freeze the backbone layers during training. This helped prepare the model for future training and comparison with our other CNN architectures. I also began working on our initial FastAPI backend by working on the SQLite database layer, including the users and evaluations tables. I developed some initial code for running model predictions through the backend. One challenge for me was becoming more familiar with the model architecture and understanding how the components of our system would connect.

Loleyi Oluwatomisin

My work this milestone centered on the dataset restructuring for the two-model cascade. They helped create the new dataset subsets that split the data into the two binary tasks the cascade needs: the (COVID + symptomatic) vs healthy grouping for stage 1 and the COVID vs symptomatic subset for stage 2. This included the binary label-grouping utilities that remap the 3-class labels (healthy / COVID-19 / symptomatic) into each binary task's targets: the one-vs-group mapping for stage 1's merged sick class and the per-label filtering that builds stage 2's standalone dataset from only the sick recordings. They also supported the augmentation experiment runs.

Richard Alonso Garcia

My main contributions for this milestone centered on the React UI components for prediction results and risk feedback. I completed the full dashboard, which was unfinished before this milestone. It includes a latest result card with a confidence score, a color coded risk badge (low, medium, high), and plain language advice for each risk level. It also has a vote breakdown for the three coughs, a detail view with signal strength and recording quality, and a confidence trend chart for tracking results over time. I connected the dashboard to the recorder so a full test session flows from starting a test to viewing the result. I also fixed bugs in the recorder code that kept it from running reliably, including duplicate microphone sessions and a microphone that could stay on after leaving the page. The /evaluate integration is the next step, and the dashboard is structured so live results can replace the current placeholder data.

7. Plan for the next Milestone (task matrix) — Milestone 3 (November 23rd)

Task	**Drew Quashie**	**Amanda Ogbonna**	**Loleyi Oluwatomisin**	**Richard Alonso Garcia**
Evaluate and integrate additional data sources to grow the training set	x	x	x	x
Improve preprocessing and feature extraction for the cascade models	x	x	x	x
Rebalance the training data and revisit the class-weighting strategy	x	x	x	x
Refine stage 1 of the cascade: thresholds, architecture, and sick-class recall	x	x	x	x
Retrain and re-evaluate the full cascade at the improved operating point	x	x	x	x

8. Discussion (at least a paragraph) of each planned task for the next Milestone

Evaluate and integrate additional data sources to grow the training set

Stage 1's central weakness is that the models learn the class prior more than the signal, and the most direct fix is more and better sick-class data. This task surveys the additional cough datasets available to us (CoughVid subsets we have not yet merged, Coswara, and other public COVID cough corpora), checks each for licensing, recording quality, and label overlap with our healthy/symptomatic/COVID classes, and defines the merge criteria so new data does not contradict the recording-wise split (speaker identity must remain unique across train/validation/test after any merge). If a usable source is identified, the dataset-building pipeline is extended to ingest it and the class counts are re-measured. This task is deliberately scoped as an evaluation first: we commit to the survey and the merge criteria, and only commit to full integration once a source clears the quality bar.

Improve preprocessing and feature extraction for the cascade models

The branched stage-2 win (ROC-AUC 0.637 vs 0.599 for the default CNN) suggests the input representation matters, so this milestone revisits what goes into the models rather than only the models themselves. Work includes tuning the spectrogram parameters (n-FFT, hop, mel bands) against stage-1 validation AUC instead of leaving them at defaults, examining per-recording quality filtering (clipping, silence, background noise) so noisy recordings stop injecting label noise, and evaluating whether additional feature channels (e.g. delta features or per-window energy) give the stage-1 model anything the current MFCC + log-Mel pair does not. Every preprocessing change is validated with the same recording-wise split and the same metrics (ROC-AUC and PR-AUC, not accuracy) so improvements are attributable.

Rebalance the training data and revisit the class-weighting strategy

Healthy dominates the data roughly 3-to-1 over each sick class, and stage 1 exploits that prior. This task experiments with the rebalancing levers systematically: adjusting the per-class weighting in the loss, downsampling healthy or upsampling sick recordings, and per-class augmentation probabilities (the current 90/70/40% scheme) tuned against stage-1 sick recall on validation rather than overall accuracy. The success criterion is explicit: raise stage-1 sick recall above the current ~0.5 without letting healthy precision collapse, since end-to-end COVID recall is currently capped by stage 1 routing most COVID recordings to healthy before stage 2 ever runs.

Refine stage 1 of the cascade: thresholds, architecture, and sick-class recall

This task owns stage 1 end to end. First, thresholds: the sick/COVID operating points used in the assembled evaluation were defaults, not validation-tuned, so we sweep the sick threshold on the validation set and quantify the healthy-recall-for-sick-recall tradeoff. Second, architecture: transfer what stage 2 taught us (the dual-branch input representation helps) back into stage 1, and test whether deeper or wider variants change the 0.56 AUC ceiling. Third, recall: any change is judged by stage-1 sick recall and AUC on the validation split, because that single number gates how much of stage 2's signal can ever reach the final output.

Retrain and re-evaluate the full cascade at the improved operating point

Once the data, preprocessing, rebalancing, and stage-1 changes land, the full two-model cascade is retrained end to end and re-evaluated with the same report set built this milestone: stage-1 and stage-2 standalone ROC-AUC and PR-AUC, per-class confusion matrices and recalls at the new validation-tuned thresholds, and the assembled 3-class results. The comparison target is this milestone's baseline (branched stage 2 at 0.637 ROC-AUC, assembled COVID recall 0.03–0.07), and the goal is to move assembled COVID recall materially above that floor by raising stage-1 sick recall. The numbers from this run become the paper's evaluation section and the operating points the confidence gate uses.

11. Date(s) of meeting(s) with Faculty Advisor during the current milestone

1. Thursday October 1st 11:15am — Olin Engineering Complex room 353
2. Thursday October 8th 11:45am — Olin Engineering Complex room 353
3. Thursday October 15th 12:00pm — Olin Engineering Complex room 353
4. Thursday October 22th 12:00pm — Olin Engineering Complex room 353

12. Faculty Advisor feedback on each task for the current Milestone

Train & Fine-Tune CNN Models & Transfer Learning Architectures

Initial Model Evaluation & Performance Reports

Develop FastAPI Backend Endpoints

Build React Web UI Components for Prediction Results and Risk Feedback