Issues encountered so far:

1. The first dataset we worked on was the lung disease dataset. It has 3 classes. We failed to train a model able to distinguish all 3 of them, despite attempts using numerous architectures and hyperparameters' sets. In the end, we've made a decision to change the task to melanoma classification.
2. Our CRP results with ResNet were unsatisfactory. It indicated that in a given layer, all channels are influenced by the input the same way, which seemed counterintuitive. We suspected there might be a bug in the library. After struggling with the problem and getting to know the source code, we stumbled upon a GitHub issue similar to ours. It turned out zennit-crp passes relevance through residual connections (which it shouldn't do). Now equipped with our knowledge of the frameworks' code we could tackle the problem by implementing a custom canonizer, which handles residual connections in the proper manner.
3. Artifacts found in the dataset - all photos made with the microscope belong to one class (melanoma). The problem was discovered by inspecting the explanations of microscope photos - the model focused on the dark ring only present in those samples. After noticing the issue, we tested the hypothesis that all microscope samples belong to one class using a python script implementing a simple heuristic and some manual checking. It turned out to be true. On the bright side, the number of those samples is not high enough to skew the model or the metrics.

Current progress:

1. Training pipeline created and working.
2. Two models (ResNet50, and custom ResNet50 with attention) trained and evaluated, achieving > 95% accuracy and high precision/recall on validation set.
3. Explanation using Grad-CAM and attention layer created.
4. Sample explanations with LRP and CRP created.
5. Issue with using ResNet with zennit-crp resolved (we posted our workaround in the zennit-crp github repo - https://github.com/rachtibat/zennit-crp/issues/16).

Next steps:

1. Train and explain custom EfficientNet with ReLU.
2. Evaluate models based on two metrics: Explanation sensitivity and Explanation Infidelity.
3. (Optional) Combine Attention-ResNet with CRP to create better explanations
4. (Optional, depending on whether our solution would make sense in the context of the original codebase) Create a Pull Request fixing the issue with ResNet in zennit-crp.
5. Create a report in the form of a Jupyter Notebook.