Here are my thoughts on the general strategies we can take for the competition:

**In general:**

Something that worked really well for the deepfake competition was this structure. Note it is a cycle and phases can get out of order, but in general we want to do it in the order below. The Action Phase and Tuning Stage can be done in parallel.

**Research Phase**:

Main goal is to get as much background information as possible to make smart decisions (and save us a lot of time later)

1. Do a lot of reading on what is currently the best model out there to transfer learn on for our specific use application.
2. Do a lot of reading on building robustness on the type of data we are looking at for augmentation
3. Do a lot of reading on techniques that may help us have better training results on our model with the type of data we are given
4. Figuring out your data format

**Action Phase:**

Here is where we do a major lift of the basic code, and try to make it as modular as possible and ensure it just works. Later we may want to make quick changes and having to rewrite a bunch of code will be painful. Using Stack Overflow and libraries are key to making this go quickly. If one library is not working try to move to another that will instead of trying to figure out some obscure bug with little documentation (I definitely did not do this)

1. Read Data Code
2. Split Data Code
3. Build Augmentation Code
4. Store Data Code
5. Train Basic Model Code
6. Test Basic Model Code
7. Build Modular plotting Code

COMMENT EVERYTHING!!

**Tuning Phase:**

This is where all the main code is built and everything is working how we expect, we go into training the models. We will most likely want to go local or on HPC

1. Getting HPC access (if necessary)
2. Training on Local Hardware (I can try to do this, I've had bad luck in the past)
3. Trying different hyperparameters based on research (there are libraries for this!!!)
4. Log all changes in another doc! Or just make more files with versions, we practically have endless storage space on the Drive.

Some things that went poorly to note:

* We waited way too long to get stuff running on local. I think getting this implemented early on will help a ton, instead of scrambling to the end.
* Having one person in charge of each action. People ended up rewriting a lot of code that someone else made. Also some of the code became incompatible.
* Lack of comments. I had to have people sit down with me to explain something if I wanted to work on it. Hell even make diagrams in draw.io real fast. It can be sloppy and quick is better than nothing!

Some things that went well to note:

* Working in parallel was huge. If there was a dependence on something, one person would write mock code for the other person to use so they could continue to work in parallel
  + One example is Person A wanted to start using real data to train the model, so Person B made super simple code to load it for them to stick it in before moving to augmentation.
* Documenting the changes we were making along the way. That way if something worked he saved it and if it broke we could see what changed. Think of 306 project backups.

**Competition Project**

**Phase 1:**

* We may be able to skip the Research Phase here. Getting something working and submitted may be our biggest priority since we may want to actually use our spring break lol
* One compromise is to use a basic RNN, since that is the structure we may most likely use on our final project.

**Phase 2:**

* This is where we need to walk through all the phases and be diligent about keeping up with it. Working in