## Project Specification

# Computer Pointer Controller

Project Setup And Style

| Criteria | Meets Specifications |
|---|---|
| The code demonstrates the use of virtual environments. | The code uses a virtual environment for project isolation.<br><br>All required dependencies and their versions are listed in a requirements.txt file.<br><br>Tip: You should not paste the whole output of `pip freeze`. Only the required dependencies should be listed. |
| The code demonstrates error handling and logging. | When the program encounters an error, an exception is raised and a meaningful error message is displayed.<br><br>The program uses logging to track important events. |
| The project contains detailed documentation. | The project contains a README file with the following details:<br><br>• Short Introduction/Description<br>• Project Setup and Installation<br>• How to run a demo<br>• The command line options<br>• Explanation of the directory structure and overview of the files used in the project |

Inference Engine Pipeline

| Criteria | Meets Specifications |
|---|---|
| The project demonstrates the use of classes and functions. | The program organizes reusable blocks of code into functions.<br><br>The program encapsulates related methods and data into classes. |

| Criteria | Meets Specifications |
|---|---|
| The project demonstrates the ability to perform inference on video files or a webcam feed. | Based on user input, the project uses either a video file or a webcam feed to perform inference. |
| The project implements an inference data pipeline. | The project includes an inference pipeline in which:<br><br>• Input frames are fed to models for inference.<br>• Outputs from multiple models are fed consecutively to other models. |
| The project demonstrates the use of multiple model precisions (FP, INT). | The submission includes a write-up in the README. The write-up should contain:<br><br>• Benchmarking results for models of different precisions<br>• Discussion of the difference in the results among the models with different precisions (for instance, are some models more accurate than others?). |
| The project demonstrates an ability to show output of intermediate models for visualization. | The code allows the user to set a flag that can display the outputs of intermediate models.<br><br>The output is shown using a visualization of the output model (not just printed). |

Running the Program

| Criteria | Meets Specifications |
|---|---|
| The code demonstrates parsing of command-line arguments. | The code uses command-line arguments to change the behavior of the program. For instance, specifying model file, specifying hardware type, etc.<br><br>Where possible, default arguments are used for when the user does not specify the arguments.<br><br>A `--help` option should be present and should display information about the different configurations. |
| The code demonstrates the ability to run inference on multiple supported hardware options. | The program allows the user to select a hardware option on which to run the model (CPU, VPU etc). Inference then runs on the hardware that the user has chosen. |

## Suggestions to Make Your Project Stand Out!

1. Can you improve your inference speed without significant drop in performance by changing the precision of some of the models? In your README, include a short write-up that explains the procedure and the experiments you ran to find out the best combination of precision.

2. Write test cases and ensure your code covers all the edge cases.

3. Benchmark the running times of different parts of the preprocessing and inference pipeline and let the user specify a CLI argument if they want to see the benchmark timing. Use the `get_perf_counts` API to print the time it takes for each layer in the model.

4. Use Async Inference to allow multiple inference pipelines. Show how this affects performance and power as a short writeup in the README file.

5. There will be certain edge cases that will cause your system to not function properly. Examples of this include: lighting changes, multiple people in the same input frame, and so on. Make changes in your preprocessing and inference pipeline to solve some of these issues. Write a short write-up in the README about the problem it caused and your solution.

6. Add a toggle to the UI to shut off the camera feed and show stats only (as well as to toggle the camera feed back on). Show how this affects performance and power as a short write up in the README file.

7. Build an inference pipeline for both video file and webcam feed as input. Allow the user to select their input option in the command line arguments.