- All of the assignments should be submitted with properly managed code-base with Readme.md and a visualization of code working using any illustration tool of your choice, final illustration should be in PDF format
- Proper code comments and functional design of code structure is must.
- Questions marked as (*) are meant to be **tough** questions, pseudo codes/providing approaches are highly encouraged.

1. Natural Language Processing and Synthesis

Question 1:

Implement a text-to-speech (TTS) system that can read out project updates from a given text file. Use a TTS library such as Google's TTS API, Amazon Polly, or open-source alternatives like Mozilla's TTS. Try to use a local TTS model if possible.

Requirements:
- Use Python for implementation.
- Support for multiple languages is a plus.
- Provide a brief demo video showcasing the TTS system.

Question 2*:

Develop a basic chatbot that can understand and respond to queries about IIT Bombay's humanoid project. The chatbot should be able to answer questions about the project's objectives, current progress, and team members. Implement this using a natural language processing library like NLTK, spaCy, or Hugging Face's Transformers. (Explore LangChain, RAG, HuggingFace)

Requirements:
- Use Python for implementation.
- Incorporate a language model to understand and generate responses after training on data provided.
- Provide a README file explaining the setup and usage.

Note: Data required to train the model will be provided in PDF/Word Format.

2. Object Detection with PyTorch/TensorFlow/YOLO

Question 1:

Create an object detection model using PyTorch or TensorFlow to identify and classify common objects in a laboratory environment (e.g., computers, tools, books). Use a pre-trained model like YOLOv8/9/10 or Faster R-CNN and fine-tune it on a custom dataset.

Requirements:
- Use PyTorch or TensorFlow for implementation.
- Annotate a small dataset (at least 50 images) for training.
- Evaluate the model's performance and provide accuracy metrics with the challenges faced and solutions.

Question 2:

Implement a YOLO-based object detection system for identifying components of a humanoid robot. Train the model to recognize parts like arms, legs, sensors, and cameras.

Requirements:
- Provide a dataset of labelled images of robot components.
- Use YOLOv8/9/10 for training and inference.
- Present a short report on model performance and accuracy.

3. Experimentation with Vision Transformers

Question 1*:

Explore and implement a Vision Transformer (ViT) model for image classification. Use a publicly available dataset (e.g., CIFAR-10, ImageNet) to train the model and compare its performance with a CNN-based model.

Requirements:
- Finetune both ViT and CNN models and compare results. (You can use any publicly available model for this task.)
- Provide visualizations of the training process and performance metrics.

Question 2*:
Implement a ViT-based image segmentation model to segment parts of a humanoid robot from images. Use a pre-trained ViT model and fine-tune it on a relevant dataset.
Requirements:
- Annotate a dataset with segmentation masks for training.
- Use transfer learning to adapt a ViT model for segmentation tasks.
- Document the process and results in a concise report.

4. Stereo Imaging and Depth Analysis

Question 1:
Develop an algorithm to calculate depth information from stereo images captured by the robot's stereo camera. Implement a depth map visualization and analyse the performance in various lighting conditions.

Requirements:
- Use OpenCV or a similar library for image processing.
- Test the algorithm on different stereo image pairs.
- Provide a detailed report with depth map examples and analysis.

Note: Use any publicly available stereo image dataset with calibrated readings for reference.

Question 2*:
Create a system that uses stereo imaging to detect obstacles and calculate their distances from the robot. Implement a real-time demo showing obstacle detection and distance measurement.

Requirements:
- Use Python for implementation with OpenCV or similar libraries.
- Demonstrate real-time processing capabilities.
- Discuss the system's performance and potential improvements.

Note: Capture a video of a busy corridor in insti and use it as input video.