## Step-by-step procedure to run this contactless heart rate monitoring project:

### Prerequisites:

1. **Google Colab Environment:** This project is designed to run in a Google Colaboratory environment. You'll need a Google account to access Colab.
2. **Google Drive Access:** The notebook assumes your video file (`vid.avi`) and ground truth file (`ground_truth.txt`) are located in a specific directory in your Google Drive: `/content/drive/MyDrive/hr_ds/subject3/`. Ensure these files exist in this location or modify the `VIDEO_FILE` and `GT_FILE` constants accordingly.
3. **Internet Connection:** You'll need an active internet connection to download and install the required libraries and to mount Google Drive.

### Steps to Run:

1. **Open the Colab Notebook:** Open the provided Google Colab notebook link in your web browser.
2. **Connect to a Runtime:** In the Colab menu, go to "Runtime" and select "Connect to a hosted runtime". This will allocate computational resources for running the code.
3. **Mount Google Drive:** The notebook includes the following code to mount your Google Drive:
4. Python

```
from google.colab import drive
drive.mount('/content/drive')
```

5. 
6. Run this cell. You will be prompted to authorize Colab to access your Google Drive. Follow the on-screen instructions.
7. **Install Required Libraries:** The notebook starts by uninstalling any existing `mediapipe` and then installing a specific version:
8. Python

```
!pip uninstall mediapipe -y
!pip install mediapipe==0.10.10 --no-cache-dir
```

9. 
10. Run this cell to install the necessary version of the `mediapipe` library. You'll also see other `pip install` commands for libraries like `graphviz`, `pydot`, and `gradio` later in the notebook; run these cells as you encounter them.

11. **Define File Paths and Constants:** Ensure the `VIDEO_FILE` and `GT_FILE` constants at the beginning of the notebook correctly point to the location of your video and ground truth files in your Google Drive. If they are in a different location, update these paths. Also, verify the `IMG_HEIGHT`, `IMG_WIDTH`, and `IMG_CHANNELS` constants are appropriate for your project.
12. **Run Data Loading and Preprocessing Cells:** Execute the cells under the "Data Loading and Preprocessing" section. This will:
    - Load the video frames.
    - Detect and crop faces using Mediapipe.
    - Preprocess the cropped face images.
    - Load the ground truth heart rate values.
    - Split the data into training and testing sets.
    - Display a few sample extracted faces.
13. **Run Model Definition and Training Cells:** Execute the cells for each of the defined models (LeNet, Custom CNN, GoogLeNet, and UNet) and their respective training sections. This will:
    - Define the architecture of each neural network model.
    - Compile each model with an optimizer, loss function, and metrics.
    - Train each model on the training data.
    - Save the trained models to `.keras` files.
14. **Run Model Evaluation Cells:** Execute the evaluation sections for each model. This will:
    - Load the trained models.
    - Evaluate their performance on the test data using metrics like MSE and MAE.
    - Print the evaluation results.
    - Visualize the model architectures.
15. **Run Ensemble Model Cells:** Execute the cells for creating and evaluating the ensemble model. This will:
    - Load the individual trained models.
    - Create an ensemble model by averaging the predictions of the individual models.
    - Evaluate the ensemble model on the test data.
    - Print the ensemble model's performance.
    - Visualize the ensemble model architecture.
16. **Run Live Video Processing and Heart Rate Estimation (Optional):** Execute the cells under the "Live Video Processing and Heart Rate Estimation" section. **Note:** This section uses a simplified signal processing approach with a synthetic PPG signal for demonstration. For actual live processing and PPG extraction from video, more advanced techniques would be needed. This section will:

    - Define functions for extracting frames and applying filters.
    - Process the video file.
    - Predict heart rate using the ensemble model.
    - Perform heart rate estimation using signal processing on a synthetic signal.
    - Display plots and print estimated heart rate, breathing rate, and accuracy (based on the example ground truth).
17. **Run Gradio Interface (Optional):** Execute the cells under the "Gradio UI for Live Processing" section. This will:
    - Load the trained ensemble model.
    - Define a function `process_video` that takes a video file as input, processes it, predicts heart rate, and generates plots (again using a synthetic PPG for demonstration).
    - Launch a Gradio web interface. You will see a public URL that you can click to access the interface in your browser. You can then upload a video file through this interface to see the estimated heart rate and the generated PPG-like plots.

**Important Considerations:**

- **File Paths:** Double-check that the file paths for the video and ground truth files are correct in your Colab environment.
- **Library Versions:** The notebook specifies a particular version of `mediapipe`. Using a different version might lead to compatibility issues.
- **Training Time:** Training the neural network models can take a significant amount of time depending on the runtime resources allocated by Colab.
- **Synthetic PPG:** The "live" processing in this notebook uses a synthetic PPG signal for demonstration. Real-world contactless heart rate monitoring requires sophisticated techniques to extract a meaningful PPG signal from facial video.
- **Ground Truth Data:** Ensure your `ground_truth.txt` file contains the heart rate values corresponding to the frames in your video in the correct format (space-separated on a single line).

By following these steps in order, you should be able to run the project and observe the results of the heart rate estimation based on the provided video data and the trained neural network models. Remember that the accuracy of this implementation will depend on the quality of the data and the effectiveness of the models.