Part 1:

gst-launch-1.0 filesrc location="/Users/zaynsmacantosh/Desktop/gstream/Crowd of People Walking in London 4K UHD Stock Video Footage.mp4" ! decodebin ! videoconvert ! videoscale ! video/x-raw,width=640,height=640 ! jpegenc ! multifilesink location=output_%05d.jpg

Output :

Setting pipeline to PAUSED ...

Pipeline is PREROLLING ...

Redistribute latency...

Redistribute latency...

Redistribute latency...

Pipeline is PREROLLED ...

Setting pipeline to PLAYING ...

Redistribute latency...

New clock: GstSystemClock

Got EOS from element "pipeline0".

Execution ended after 0:00:02.854130875

Setting pipeline to NULL ...

Freeing pipeline ...

Part 2 :

1) **Created a gstream.cpp file using vscode**

```
  #include <opencv2/opencv.hpp>
#include <filesystem>
#include <iostream>
#include <sstream>


namespace fs = std::filesystem;


int main() {
// Path to the directory containing the JPEG frames
std::string frameDir = "/Users/zaynsmacantosh/Desktop/gstream";
```

```
// Create a directory to store cropped faces
fs::create_directory("cropped_faces");

// Iterate over each JPEG frame
for (const auto& entry : fs::directory_iterator(frameDir)) {
std::string framePath = entry.path().string();

// Read the frame
cv::Mat frame = cv::imread(framePath);
if (frame.empty()) {
std::cerr << "Failed to read frame: " << framePath << std::endl;
continue;
}

// Convert the frame to grayscale for face detection
cv::Mat grayFrame;
cv::cvtColor(frame, grayFrame, cv::COLOR_BGR2GRAY);

// Load the pre-trained face detection model
cv::CascadeClassifier faceCascade;
if (!faceCascade.load("haarcascade_frontalface_default.xml")) {
std::cerr << "Failed to load face detection model!" << std::endl;
return 1;
}

// Detect faces in the frame
std::vector<cv::Rect> faces;
faceCascade.detectMultiScale(grayFrame, faces);

// Create a folder for this frame's faces
std::stringstream folderName;
folderName << "cropped_faces/frame_" << fs::path(framePath).stem();
fs::create_directory(folderName.str());

// Crop and save each detected face
for (size_t i = 0; i < faces.size(); ++i) {
cv::Rect faceRect = faces[i];
cv::Mat croppedFace = frame(faceRect);

std::stringstream faceName;
faceName << folderName.str() << "/face_" << i << ".jpg";
cv::imwrite(faceName.str(), croppedFace);
}
}

std::cout << "Face cropping completed successfully!" << std::endl;
return 0;
}
```

2) compiled it using

g++ -std=c++17 -I/opt/homebrew/opt/opencv/include/opencv4 gstream.cpp -o gstream `pkg-config --libs opencv4`

3) got the executable file and ran it using

./gstream

Output :

Face cropping completed successfully!
(saved in the file cropped_faces

4) got a new folder which saves all the cropped faces.