Дигитално процесирање на слика

Тема:

**Детекција на обележја на лицето**  
(Откривање на специфични карактеристики на лицето како што се очите,носот и устата во дадена слика или видео. Откривање на поспаност во видео.)

Изработиле: Стефан Колов 223226, Андреј Јакимовски 226086

**Содржина**

[1. Вовед 3](#_Toc169517525)

[2. Преглед на функционалности 3](#_Toc169517526)

[2.1 Детекција на лице во слика 3](#_Toc169517527)

[2.2 Детекција на лице во видео 3](#_Toc169517528)

[3. Инструкции на апликацијата 4](#_Toc169517529)

[3.1 Почеток на апликацијата 4](#_Toc169517530)

[3.2 Обработка на слика 4](#_Toc169517531)

[3.3 Обработка на видео 5](#_Toc169517532)

[4. Детали за имплементацијата 6](#_Toc169517533)

[4.1 Вклучени библиотеки 6](#_Toc169517534)

[4.2 Функција за детекција на обележја на лице во слика (detect\_face\_image) 6](#_Toc169517535)

[4.3 Функција за детекција на обележја на лице во видео (detect\_face\_video) 8](#_Toc169517536)

[4.4 Пресметка на EAR (Eye Aspect Ratio) 10](#_Toc169517537)

[5. Тестирање на функционалности 11](#_Toc169517538)

[5.1 Тестирање со слика 11](#_Toc169517539)

[5.2 Тестирање со видео 12](#_Toc169517540)

[5.3 Тестирање со невалиден влез 14](#_Toc169517541)

[5.4 Тестирање со невалидна патека 14](#_Toc169517542)

[6. Потенцијални напредни функционалности 15](#_Toc169517543)

[7. Заклучок 16](#_Toc169517544)

## 1. Вовед

Детекцијата на лица е значајна област во компјутерската визија и има широк спектар на апликации, вклучувајќи безбедност, биометриски системи, и човек-компјутер интеракции. Нашата апликација за детекција на обележја на лица е дизајнирана да овозможи корисниците лесно да детектираат специфични делови на лицето, како што се очи, нос и уста, во дадена слика или видео. Дополнително, апликацијата може да обработи видео и да открие дали личноста спие, користејќи напредни техники за обработка на слики и машинско учење. Оваа апликација користи неколку популарни библиотеки како OpenCV и dlib за обработка и анализа на слики и видео. Со нивна помош, можеме прецизно да детектираме лица и нивните обележја, како и да анализираме движења на лицето. На пример, преку детекција на затворени очи во продолжителен временски период, можеме да откриеме дали личноста спие. Во оваа документација ќе ги разгледаме деталите за функционирањето на апликацијата, инструкциите за користење, и имплементацијата на главните компоненти. Целта е да се обезбеди целосен преглед на апликацијата и да се помогне на корисникот да ја разбере и ефективно да ја користи.

## 2. Преглед на функционалности

Во овој дел ќе ги разгледаме главните функционалности на апликацијата за детекција на лице и обележја и откривање на поспаност. Апликацијата е дизајнирана да работи со слики и видеа, обезбедувајќи прецизна и ефикасна детекција на обележја на лицето и анализа на состојбата на очите (поспаност).

## 2.1 Детекција на лице во слика

* Вчитување на сликата и претварање во greyscale.
* Детектирање лице, нос, очи и уста во сликата.
* Обележување на точките на детектираните елементи.

## 2.2 Детекција на лице во видео

* Вчитување видео и претварање во greyscale.
* Детектирање лице во секој кадар (frame) од видеото.
* Обележување на точките на детектираните елементи од лицето во секој кадар.
* Проверување дали лицето спие користејќи го EAR алгоритмот.

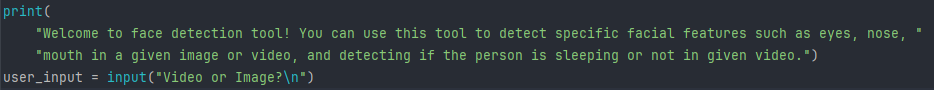
Овие функционалности овозможуваат прецизна и ефикасна детекција на обележја на лицето и анализа на состојбата на очите во реално време, што ги прави овие алатки идеални за апликации како мониторинг на возачи, безбедносни системи и други биометриски апликации.

## 3. Инструкции на апликацијата

Самата апликација е изградена во Python користејќи неколку библиотеки. Апликацијата се состои од 3 Python фајла кој се именувани како “main.py, image.py, video.py”. Во фајловите за Image и Video се наоѓаат соодветните имплементирани алгоритми за детекција на обележја на лицето. Во директориумот Items се наоѓаат потребните фајлови за работа на апликацијата како што се default слики и видеа и предикторот за обележја. Програмата се стартува и започнува од main.py со кој што следат дел од инструкциите за успешно користење на апликацијата.

## 3.1 Почеток на апликацијата

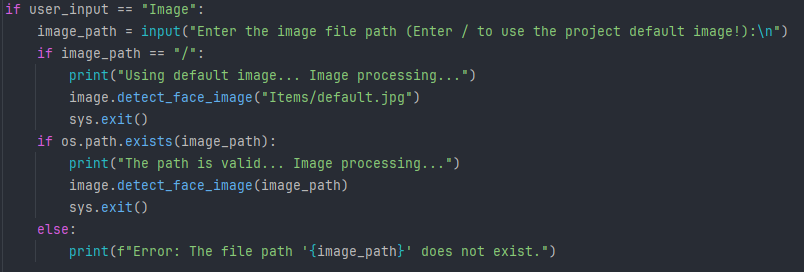
При стартување на апликацијата од main.py, на корисникот ќе му биде испечатена порака со краток опис за апликацијата и ќе му биде побарано да избере дали сака да обработи слика или видео. Изборот се внесува од стандарден влез (тастатура). Доколку корисникот внесе збор кој што не е Video или Image, сето тоа ќе биде проследено со соодветна грешка која што ќе биде испечатена и програмата нема да продолжи со работа. Тоа се е постигнато во овој дел од кодот:



Слика 1 – Дел од кодот од main.py

## 3.2 Обработка на слика

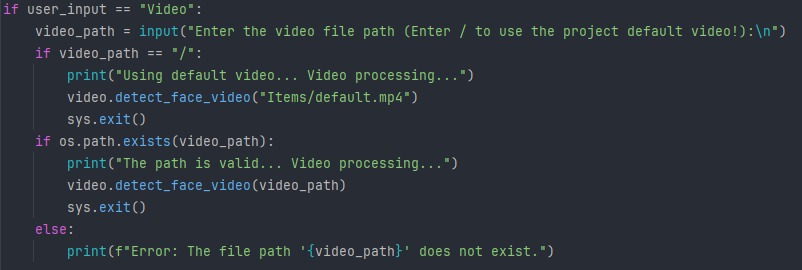
Ако корисникот избере "Image", ќе му биде побарано да внесе патека до сликата. Ако корисникот сака да ја искористи вклучената слика во проектот (default) потребно е да внесе “/”. Ако корисникот внесе патека која што не е валидна и програмата не може да ја најде или пристапи, ќе прикаже соодвнетна порака дека патеката не постои. Доколку корисникот внесе валидна патека, ќе се повика функцијата detect\_face\_image(image\_path) од image.py која како влезен аргумент ја прима патеката на сликата. Потоа сликата ќе се испроцесира во таа функција и функцијата ќе проследи некаква обработена слика како резултат.

****

Слика 2 – Дел од кодот од main.py

## 3.3 Обработка на видео

Ако корисникот избере "Video", ќе му биде побарано да внесе патека до видеото. Ако корисникот сака да го искористи вклученото видео во проектот (default) потребно е да внесе “/”. Ако корисникот внесе патека која што не е валидна и програмата не може да ја најде или пристапи, ќе прикаже соодвнетна порака дека патеката не постои. Доколку корисникот внесе валидна патека, ќе се повика функцијата detect\_face\_video(video\_path) од video.py која како влезен аргумент ја прима патеката на видеото. Потоа видеото ќе се испроцесира во таа функција и функцијата ќе проследи некакво обработено видео како резултат.



Слика 3 – Дел од кодот од main.py

## 4. Детали за имплементацијата

Во оваа секција ќе ги разгледаме деталите за имплементацијата на апликацијата за детекција на лице и негови обележја како и препознавање на поспаност во дадено видео. Апликацијата користи неколку библиотеки за обработка на слики и видеа, како и за детекција и манипулација на обележја на лиацето. Овие библиотеки овозможуваат ефикасна и прецизна анализа на визуелните податоци, што ја прави апликацијата корисна за различни примени, вклучувајќи безбедност и мониторинг.

## 4.1 Вклучени библиотеки

Апликацијата користи неколку Python библиотеки за обработка на слики и видеа, како и за детекција на обележја на лицето, а тие се:

* **OpenCV** - Библиотека која обезбедува алатки за работа со слики и видеа, вклучувајќи функции за вчитување, прикажување и манипулација на визуелните податоци.
* **dlib** - Библиотека за детекција на лице и обележја на лицето, која користи напредни алгоритми за препознавање на структури на лицето.
* **face\_utils од imutils** - Модул кој овозможува полесна манипулација со обележјата на лицето, овозможувајќи конверзија и визуeлизација на точките на лицето.
* **dist од scipy.spatial** - Модул за пресметка на растојанија помеѓу точки, кој се користи за откривање на поспаност кај личноста преку пресметка на EAR (Eye Aspect Ratio).

## 4.2 Функција за детекција на обележја на лице во слика (detect\_face\_image)

Функцијата detect\_face\_image е дизајнирана за детекција на обележја на лицето во статични слики. Таа ја прима патеката на сликата како влезен аргумент и потоа извршува неколку чекори за обработка и детекција:

* **Иницијализира детектор за лице и предиктор за обележја** - Се иницијализираат HOG-базираниот детектор за лице и предикторот за обележја од dlib библиотеката. Овие алатки овозможуваат прецизно препознавање на лицето и неговите клучни точки.
* **Вчитува слика и ја претвора во grayscale** - Сликата се вчитува и се претвора во grayscale формат, што го олеснува процесот на детекција бидејќи елиминира непотребни информации за бојата и се фокусира на структурите на лицето.
* **Детектира лице во сликата** - Се користи детекторот за лице за да се идентификуваат лицата во сликата. Овој чекор вклучува анализа на сликата и детекција на области каде што се наоѓаат лицата.
* **Ги изминува сите обележја како што се очите,носот, устата** - Откако лицата се детектирани, се користи предикторот за обележја за да се идентификуваат точките на лицето како очите, носот и устата. Овие точки се обележуваат на сликата со кругови.
* **Ја прикажува обработената слика** - Обработената слика со обележани лицеви точки се прикажува на екран за визуализација на резултатите.

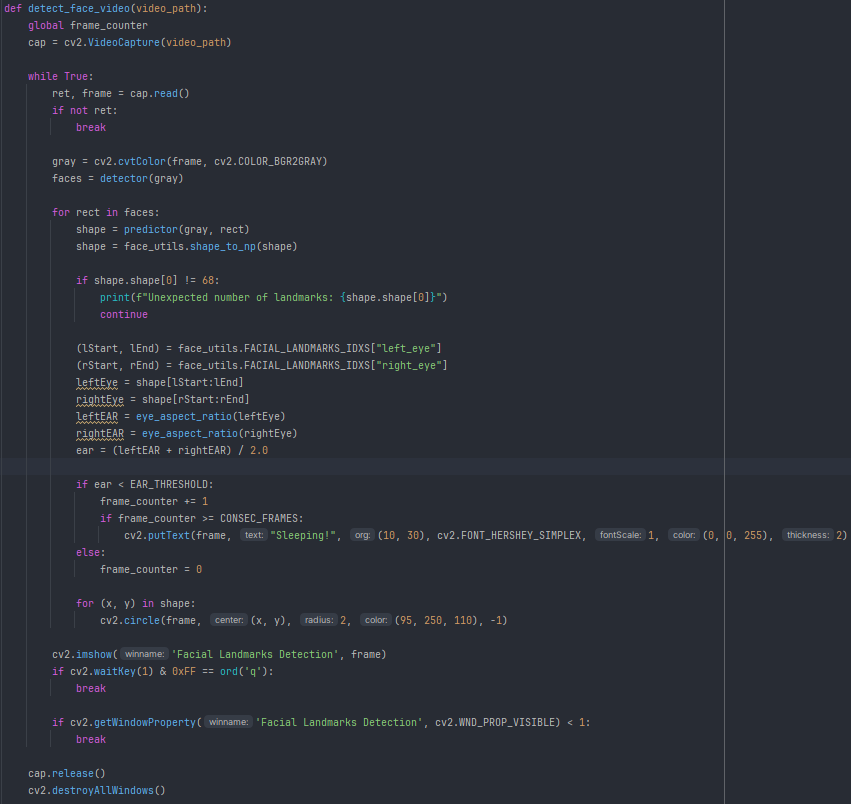
  
Слика 4.1 – Функција за детекција на обележја на лицето во слика

Функцијата започнува со иницијализација на **detector** и **predictor** каде што детекторот е HOG базиран детектор од библиотеката dlib кој се користи за да ги препознае лицата во дадена слика. Предикторот е исто така од библиотеката dlib кој ја користи датотеката “shape\_predictor\_68\_face\_landmarks.dat”, која содржи модел за препознавање на обележја на лицето. Со image ја вчитуваме сликата користејќи ја функцијата imread() од OpenCV библиотеката, и потоа ја претвараме во grayscale со функцијата cvtColor(). Променливата faces служи за детектирање на лицата во grayscale сликата користејќи го детекторот на лица. Резултатот е листа на правоаголници кои ги опкружуваат лицата. Циклусот започнува и ги поминува сите детектирани лица. За секое детектирано лице, предикторот за обележја ги идентификува обележјата и ги враќа како објект, потоа сите тие се ставаат во низа од објекти што овозможува полесна манипулација на податоците. Со последниот циклус, се изминуваат сите обележја, и за секое обележје со функцијата circle() се црта круг со соодветен радиус, боја и координати. На крај се прикажува обработената слика во посебен прозорец со наслов “Output”, и останува прикажана се додека корисникот не притисне копче за затварање на истата.

## 4.3 Функција за детекција на обележја на лице во видео (detect\_face\_video)

Функцијата detect\_face\_video е основна компонента за анализа на видеа, со цел детекција на лице и негови обележја во секој кадар од видеото. Оваа функција не само што детектира лица, туку и го користи EAR (Eye Aspect Ratio) алгоритмот за да открие дали лицето во видеото спие. Ова се чекорите и деталите за тоа како функционира оваа функција:

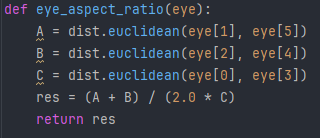
* **Иницијализира детектор за лице и предиктор за обележја** - Се иницијализираат HOG-базираниот детектор за лице и предикторот за обележја од dlib библиотеката. Овие алатки овозможуваат прецизно препознавање на лицето и неговите клучни точки.
* **Вчитува видео и го претвора секој кадар во grayscale** - Видеото се вчитува и се претвора во grayscale формат, што го олеснува процесот на детекција бидејќи елиминира непотребни информации за бојата и се фокусира на структурите на лицето.
* **Го обработува секој кадар** - Со бесконечна јамка се обработува секој frame.
* **Детектира лице и обележја од секој кадар -** Се користи детекторот за лице за да се идентификуваат лицата во видеото. Овој чекор вклучува анализа на секој кадар од видеото и детекција на области каде што се наоѓаат лицата.
* **Пресметува EAR за детекција на спиење на лицето** – Се користи за да се открие поспаност со помош на Euclidian Distance.
* **Го прикажува обработеното видео** - Се прикажува во посебен прозорец со исцртани точки околу детектираните обележја во секој кадар.

Слика 4 – Функција за детекција на обележја на лицето во видео

Функцијата detect\_face\_video започнува со иницијализација на **detector** и **predictor**, каде што детекторот е HOG базиран детектор од библиотеката dlib кој се користи за да ги препознае лицата во даден кадар од видеото. Предикторот е исто така од библиотеката dlib кој ја користи датотеката „shape\_predictor\_68\_face\_landmarks.dat“, која содржи модел за препознавање на обележја на лицето. Потоа, видеото се вчитува користејќи ја функцијата VideoCapture() од OpenCV библиотеката, овозможувајќи читање на секој кадар од видеото еден по еден. Во секој чекор од бесконечната јамка while True, се чита следниот кадар од видеото. Ако не може да се прочита кадарот, што значи дека видеото е завршено, јамката се прекинува. Секој кадар се претвора во grayscale со функцијата cvtColor() од OpenCV за полесна обработка. Променливата faces служи за детектирање на лицата во grayscale кадарот користејќи го детекторот на лица, и резултатот е листа на правоаголници кои ги опкружуваат лицата. За секое детектирано лице, предикторот ги идентификува обележјата и ги враќа како објект, а потоа тие објекти се конвертираат во низа со shape\_to\_np() за полесна манипулација. Потоа, координатите на очите се извлекуваат и се пресметува EAR (Eye Aspect Ratio) за левото и десното око. Ако EAR е под прагот за доволен број на последователни кадри, се прикажува порака „Sleeping!“ на екранот. Потоа, бројачот се ресетира. За секое обележје, со функцијата circle() од OpenCV се црта круг со соодветен радиус, боја и координати. На крај, обработениот кадар се прикажува во посебен прозорец со наслов „Facial Landmarks Detection“, и останува прикажан се додека корисникот не притисне копче за затворање. По завршувањето на видеото или при прекин од страна на корисникот, ресурсите се ослободуваат со функциите release() и destroyAllWindows() од OpenCV.

## 4.4 Пресметка на EAR (Eye Aspect Ratio)

EAR (Eye Aspect Ratio) е метрика која се користи за детекција на затворени очи. Пресметката на EAR се базира на геометрискиот однос помеѓу ширината и висината на окото. Оваа метрика е корисна за детекција на затворени очи во реално време и е особено ефикасна за апликации како што се мониторинг на возачи, детекција на замор и спиење, и други биометриски апликации. EAR се пресметува како однос помеѓу растојанијата меѓу хоризонталните и вертикалните точки на окото.

****

Слика 5 – Функција за пресметување на EAR (Eye Aspect Ratio)

Во функцијата, eye претставува низа од координати на различни точки на окото, претпоставено 6 точки кои се дефинирани како eye[0] до eye[5]. Во рамките на EAR пресметката:

* A и B се две страни на триаголникот што ги формираат внатрешните дијагонали на окото. A е растојанието помеѓу точките eye[1] и eye[5], додека B е растојанието помеѓу точките eye[2] и eye[4].
* C е дијаметарот на кругот дефиниран од точките eye[0] и eye[3], што претставува најдолгата дијагонала на окото.
* EAR се пресметува како просек на A и B, поделено со C, што дава нормализирана вредност која може да се користи за споредување на состојбата на очите меѓу различни кадри од видеото.
* Оваа метрика е ефикасна бидејќи работи во реално време и може да се користи за надгледување на возачи, детекција на замор и спиење, како и за други биометриски апликации каде што состојбата на очите е критична.

## 5. Тестирање на функционалности

Тестирањето вклучува повеќе аспекти за да се обезбеди дека апликацијата одговара на очекувањата и да се осигура дека постигнува одлични перформанси и точност во различни сценарија. Тестирањата вклучуваат различни тест сценарија како што се слики, видеа, невалидни влезови и невалидни патеки. Со овие аспекти на тестирање, се осигуруваме дека апликацијата не само што ги задоволува основните барања туку и дека ги надминува очекувањата во однос на точност, стабилност и корисничко искуство.

## 5.1 Тестирање со слика

Тестирањето се одвива така што се проверува дали апликацијата успешно детектира обележја како што се очите носот и устата на детектираните лица. Ова вклучува и тестирање на прецизноста на секоја од функциите. Како пример ќе ја користиме оваа слика:

****

Слика 6 – Не обработена слика

По стартувањето на апликацијата и внесување на горенаведната слика, апликацијата успешно ги пронајде обележјата како што се устата, носот, очите и го исцрта целото лице. Ги обележа со исцртување на зелени кругови со радиус од 3 пиксели. Резултатот е прикажан подолу на сликата:



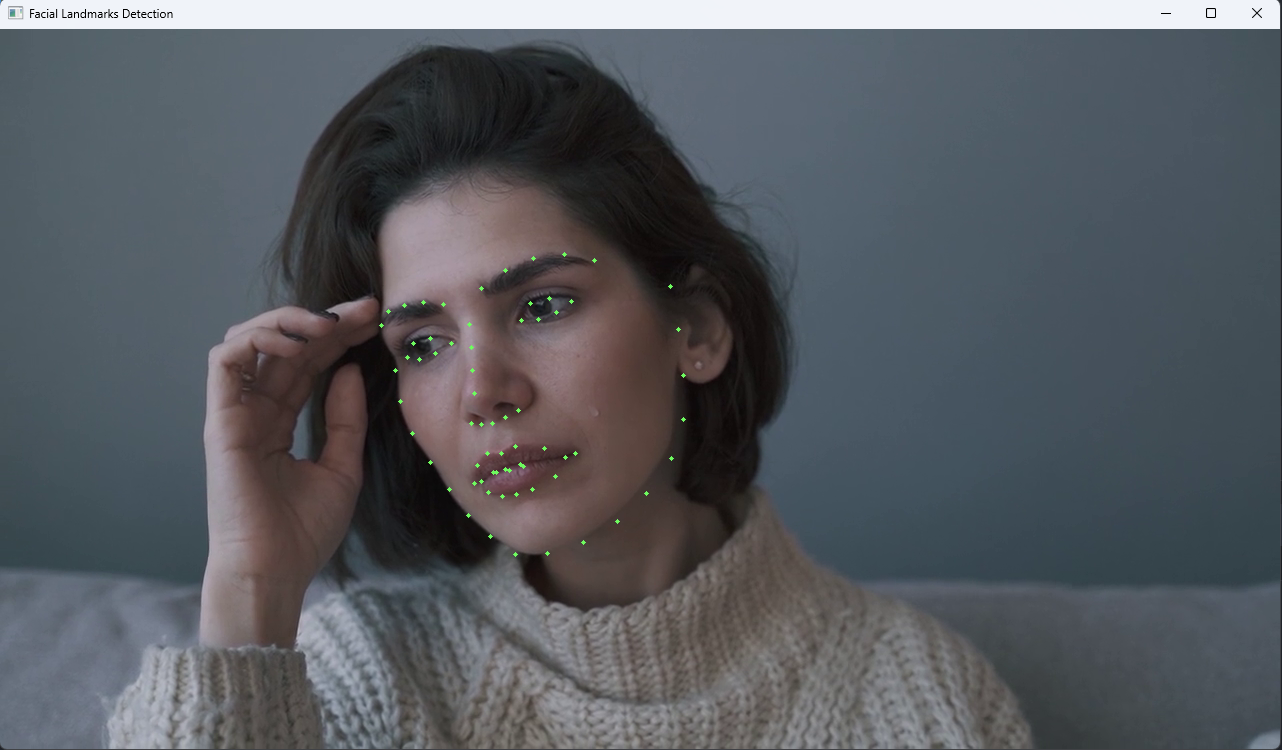
Слика 7 – Резултантна слика со детектирани обележја на лицето

## 5.2 Тестирање со видео

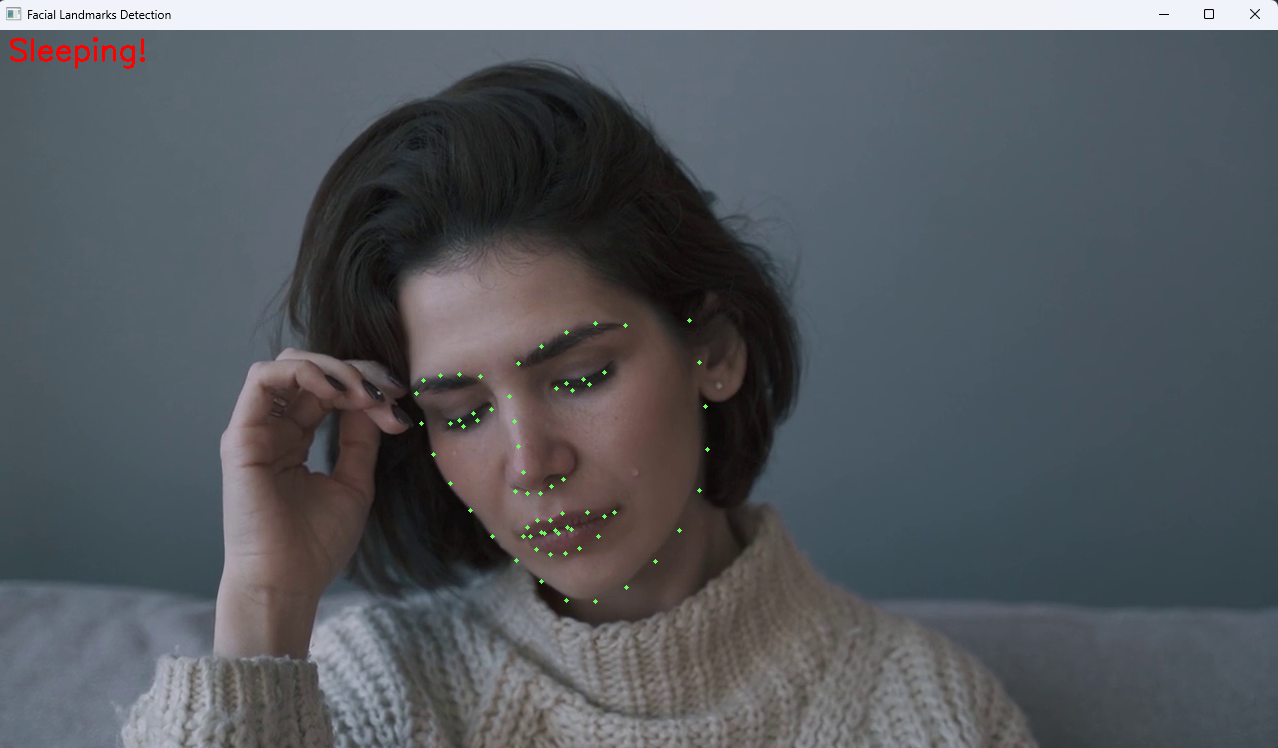
Тестирањето се одвива така што се проверува дали апликацијата успешно детектира обележја како што се очите, носот и устата на детектираните лица во видеото. Ова вклучува и тестирање на прецизноста на секоја од функциите. Ние како решение за обележја на лицето имплементиравме алгоритам којшто го обележува лицето додека трае видеото, а кога лицето на видеото ќе ги затвори очите во горниот лев агол се појавува порака "Sleeping!"



Слика 8 – Видео без детектирани обележја на лицето



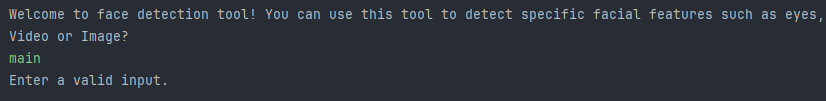
Слика 9 – Резултантно видео со детектирани обележја на лицето во секој кадар

****

Слика 10 – Резултантно видео со детектирани обележја на лицето и порака “Sleeping!” при детекција на поспаност во даден кадар

## 5.3 Тестирање со невалиден влез

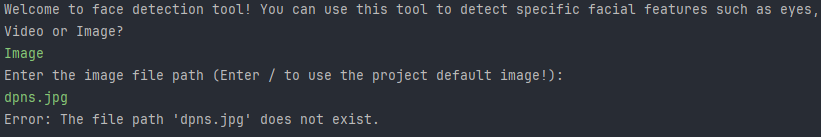
Тестирањето се одвива така што се проверува дали апликацијата успешно детектира невалидни влезови, како што се зборови што не одговараат на Image или Video.

****

Слика 11 – Резултат при внесување на невалиден влез

## 5.4 Тестирање со невалидна патека

Тестирањето се одвива така што се проверува дали апликацијата успешно детектира невалидни патеки, кои што не постојат во еден систем.



Слика 12 – Резултат при внесување на невалидна патека на датотека

## 6. Потенцијални напредни функционалности

Во рамките на овој проект, идентификувани се неколку потенцијални напредни функционалности кои би можеле да го зголемат корисничкото искуство и примената на апликацијата во различни контексти, а тие би биле:

#### **6.1 Препознавање на емоции**

Додавање на функционалност за автоматско препознавање на емоции би овозможило апликацијата да идентификува различни емоционални состојби кај лицата, како радост, тага, гнев, радост и слично. Ова може да биде особено корисно во апликации за пазарно истражување, психолошки студии итн. На пример, во маркетинг кампањи, препознавањето на емоциите може да помогне во разбирањето на реакциите на потрошувачите на различни производи или услуги, овозможувајќи компаниите да ги прилагодат своите стратегии според емоционалниот одговор на клиентите.

#### **6.2 Детекција на умор и стрес**

Имплементација на функции за детекција на умор и стрес преку анализа на обележјата на лицето би можело да биде корисно во работни средини каде што е важно да се следи состојбата на вработените. Ова може да помогне во откривање на ефикасните работни часови на вработените во некоја компанија. На пример, во транспортната индустрија, системите за мониторинг на умор кај возачите би можеле да спречат несреќи, додека во корпоративните канцеларии, ваквите системи би можеле да помогнат во одржувањето на здравјето и благосостојбата на вработените.

## 7. Заклучок

Во рамките на оваа документација, е обработен проект за детекција на обележја на лицето и детекција на поспност на лицето преку анализа на EAR (Eye Aspect Ratio). Проектот вклучува различни компоненти како што се обработка на слики и видеа, користејќи библиотеки како OpenCV, face\_utils, dlib и scipy. EAR, како геометриска метрика, се покажа како ефикасна метода за детекција на затворени очи, што е особено корисно за апликации како мониторинг на возачи или биометриски системи.

Преку пресметка на односот помеѓу растојанијата на точките на окото, може да се детектира кога лицето спие, што го прави идеален техника за подобрување на безбедноста и перформансите во различни контексти. Оваа техника не само што обезбедува практичност во реални ситуации, туку и потврдува потенцијалот и значењето на EAR во областа на компјутерска визија и биометрија.

Во иднина, може да се развива и адаптира за уште повеќе напредни и надградбени апликации. Примери за такви апликации вклучуваат напредна анализа на физичките и емоционалните состојби на лицата, или автоматизирано мониторирање на здравјето на корисниците во реално време.

Со непрекинат развој и истражување, EAR може да стане важен дел од следната генерација на технологии што ги подобруваат секојдневните животи и безбедност на луѓето.