2019

School of Computer Science (SCOPE)
Vellore Institute of Technology

Handsfree Mouse Cursor Control with User Authentication

HUMAN COMPUTER INTERACTION PROJECT REPORT

TINA GUPTA (16BCE2098)

TANVI GUPTA (17BCE0909)

ANAGHA S P (17BCI0054)

GAGAN DEEP SINGH (17BCI0140)

SUBMITTED TO: PROFESSOR SWATHI J.N

Declaration by the students….

We **Tina Gupta, Tanvi Gupta, Anagha S P, Gagan Deep Singh** are the members of the group who made this project Titled **"Handsfree Mouse Cursor Control with User Authentication"** under the supervision of our course instructor **Professor SWATHI J.N – SCOPE department**.

Only we, the members of the group, are responsible for the content of this project report, and we will take full responsibility of any plagiarism issues in the report. We will also make sure the contents of the report will not be used by any other student in Vellore Institution of Technology, Vellore or outside.

Table of Contents

Abstract:

This HCI (Human-Computer Interaction) application in Python 3 will allow authorized users to control mouse cursor with facial movements, works by accessing the system's webcam when the application is run. Automatic facial expression recognition has become a progressive research area since it plays a major role in human-computer-interaction. The facial expression recognition finds major application in areas like social interaction and social intelligence. This project is deeply centered around predicting the facial landmarks of a given known face. By identifying the facial features, this tool also prints out all the possible people's names which are stored in the datasets. It implements the concept of supervised learning which is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. We can accomplish a lot of things using these landmarks, from detecting eye-blinks (squint) in a video to predicting emotions of the subject. The applications, outcomes, and possibilities of facial landmarks are immense and intriguing. In this project, these actions are programmed as triggers to control the mouse cursor.

Objectives of the Project:

1. To recognize and authenticate the person before starting the cursor control.

    1.1 Opening the webcam when the application is run.

    1.2 The system must extract the facial features from the current webcam frame.

    1.3 Access the database of photos for recognizing the face.

    1.4 Create an automated detection system capable of detecting faces based on the following metrics:

    - Eye colour
    - Nose configuration
    - Cheekbone's position
    - Face shape

    1.5 Displaying the name of the user thus authenticating the user with his name.

2. Processing the facial movement.

    2.1 After authenticating the user, cursor control is given to the user.

    2.2 The human facial movement is decomposed into rigid movements, e.g. rotation and translation, and non-rigid movement, such as the open/close of mouth, eyes, and facial expressions, etc.

3. After calibration, the retrieved head orientation and translation can be employed to navigate the mouse cursor, and the detection of mouth movement can be utilized to trigger mouse events.

    3.1 Using these predicted landmarks of the face, certain actions, like using the eye-aspect-ratio is calculated. This Eye Aspect Ratio is used to detect eye blinks, squint, etc. EAR value has to go up when the eyes open and vice-versa.

    3.2 Another action implemented in our project depends on Mouth Aspect Ratio to detect closed/open mouth. This value has to go up when the mouth opens and vice-versa.

4. This technique can be used as an alternative input device for people with hand and speech disability and for futuristic vision-based game and interface.

Literature review:

Author /Inventor (Owner) Details	Year of Publication	Title	Publisher's Details	Contributions of the Paper, Blog, Product.
Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic Comp. Dept., Imperial College London, U.K. School of Computer Science, University of Lincoln, U.K. EEMCS, University of Twente, The Netherlands	2013	300 Faces in-the-Wild Challenge: The first facial landmark localization Challenge Link	Published in: 2013 IEEE International Conference on Computer Vision Workshops	This paper, describes the First Automatic Facial Landmark Detection Challenge. The aim of this challenge is to provide a fair comparison between the different automatic facial landmark detection methods in a new in-the-wild dataset.
Tereza Soukupova and Jan Cech Center for Machine Perception, Department of Cybernetics Faculty of Electrical Engineering, Czech Technical University in Prague	2016	Real-Time Eye Blink Detection using Facial Landmarks Link	21st Computer Vision Winter Workshop Luka Cehovin, Rok Mandeljc, Vitomir Struc (eds.) Rimske Toplice, Slovenia, February 3–5, 2016	The contributions of the paper are: 1. Ability of two state-of-the-art landmark detectors to reliably distinguish between the open and closed eye states is quantitatively demonstrated on a challenging in-the wild dataset and for various face image resolutions. 2. A novel real-time eye blink detection algorithm which integrates a landmark detector and a classifier is proposed. The evaluation is done on two standard datasets achieving state-of-the-art results.
Vahid Kazemi Josephine Sullivan	2014	One millisecond face alignment with an ensemble of regression trees Link	Published in: 2014 IEEE Conference on Computer Vision and Pattern Recognition	This paper addresses the problem of Face Alignment for a single image, shown how using appropriate priors exploiting the structure of image data helps with efficient feature selection.
Adrian Rosebrock	2017	Eye blink detection with OpenCV, Python, and dlib Link Detect eyes, nose, lips, and jaw with dlib, OpenCV, and Python Link	Personal Blog	This blog post is part three current series on facial landmark detection and their applications to computer vision and image processing. How to build upon this knowledge and develop a computer vision application that is capable of detecting and counting blinks in video streams using facial landmarks and OpenCV.

Yi Sun, Ding Liang, Xiaogang Wang, Xiaoou Tang	2015	Face Recognition with Very Deep Neural Networks Link	Cornell University	This paper was the motivation to investigate effectiveness of face recognition. This paper proposes two very deep neural network architectures, referred to as DeepID3, for face recognition. These two architectures are rebuilt from stacked convolution and inception layers proposed in VGG net and Google Net to make them suitable to face recognition.
Omkar M. Parkhi Andrea Vedaldi, Andrew Zisserman	2015	Deep Face Recognition Link	Visual Geometry Group Department of Engineering Science University of Oxford	Recent progress in this area has been due to two factors: 1. End to end learning for the task using a convolutional neural network (CNN) 2. The availability of very large-scale training datasets. The two contributions of this paper : first, we show how a very large scale dataset (2.6M images, over 2.6K people) can be assembled by a combination of automation and human in the loop, and discuss the tradeoff between data purity and time; second, traverse through the complexities of deep network training and face recognition to present methods and procedures to achieve comparable state of the art results.

Technologies to be used:

Software:

- OpenCV -Python
- MATPLOTLIB
- SCIPY
- TENSORFLOW
- CV2
- dlib

Hardware:

- Webcam

Requirements:

The elicitation techniques used for collecting requirements are Indirect techniques which include:

- Logs and notes also referred as naturalistic observation since it is good for understanding context of user activity.

Requirements of **REGISTERED** users:

The registered user shall be able to:

- Authenticate himself/herself by face recognition
- Get access to control cursor without hands
- Activate/Deactivate click action without using hands
- Activate/Deactivate scrolling without using hands

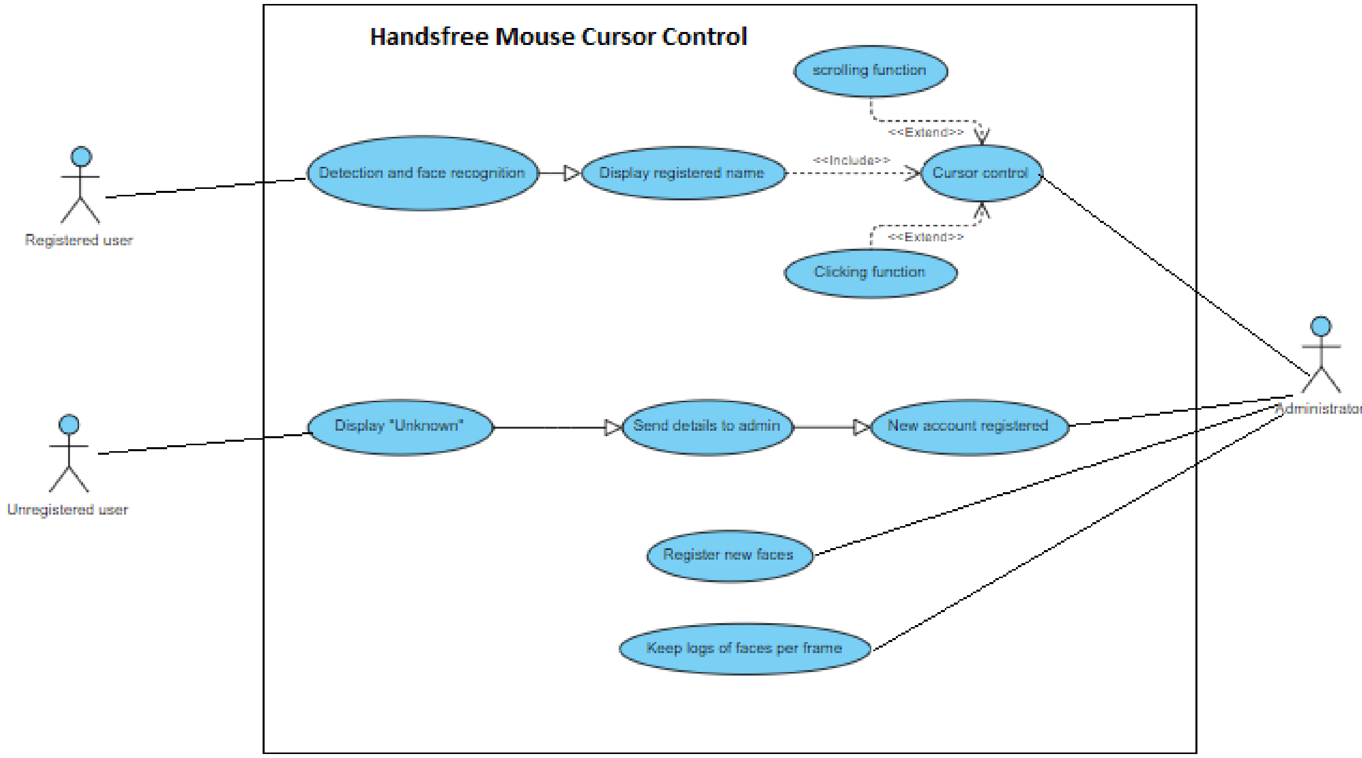
Requirements of **UNREGISTERED** users:

- The **UNREGISTERED** user shall be able to register himself/herself by sending his/her details to the administrator.

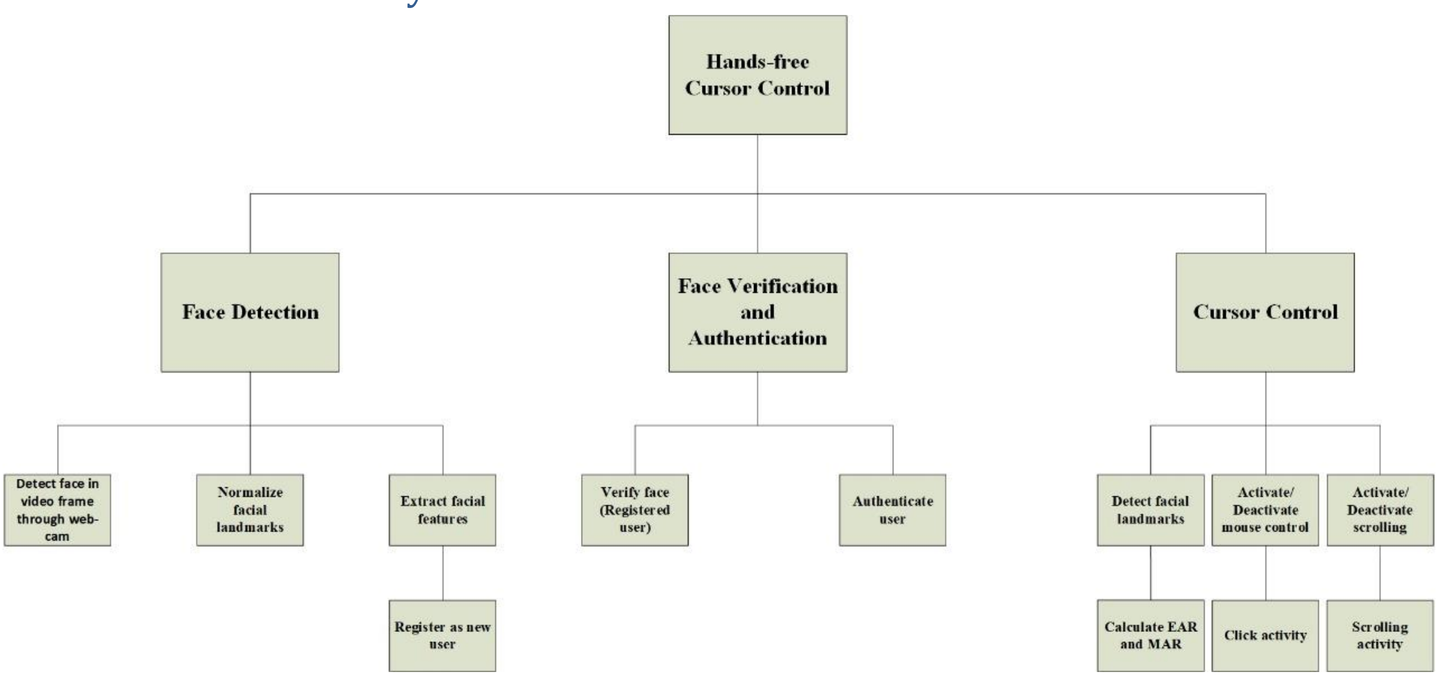
Requirements of **ADMINISTRATORS**:

- The administrator shall be able to:
- Authenticate himself/herself by face recognition
- Get access to control cursor without hands
- Activate/Deactivate click action without using hands
- Activate/Deactivate scrolling without using hands
- He / She shall be able to register a new user
- He / She shall be able to keep a log of the number of faces per frame

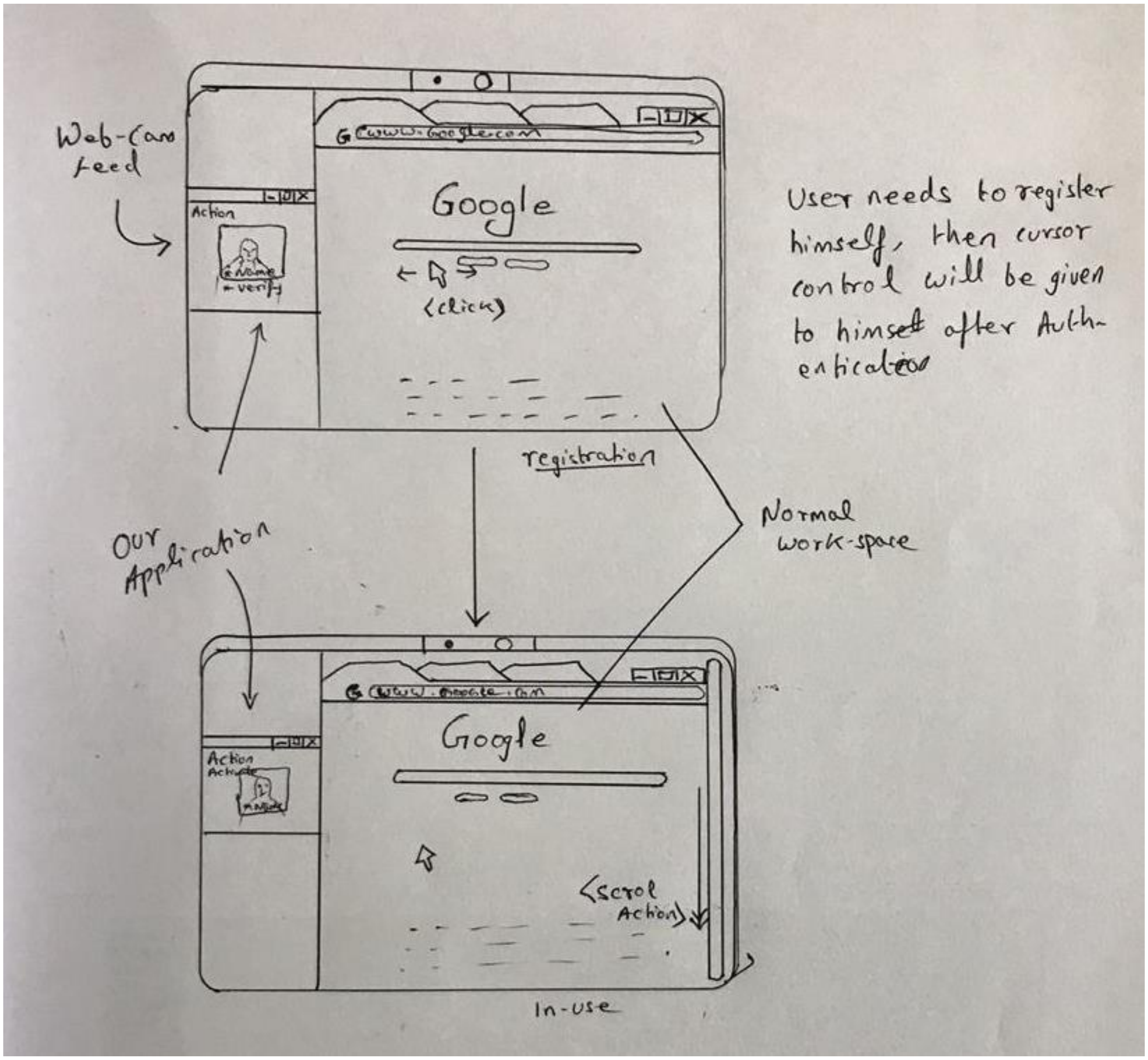
Use case diagram:



Hierarchical task analysis:



Story Boarding:



Primary Stakeholder Profiles:

USER

Cognitive ability:

- Educational level: It may be middle school/high school/undergraduate/graduate or post graduate.
- Computer literacy: For system,3-5 and for application 3-5.
- Typing skills: Unable to type.
- Domain knowledge: Any user from novice to expert is acceptable.
- Cognitive style: It may be visual, auditory or graphical.

Physical ability:

- Visual: Blind to Normal vision
- Colour vision: Colour blind to normal acceptable.
- Auditory: Deaf to Normal
- Haptic: Disabled to Fully functional.

Individual profile:

- Age: Preteen/Teen/ Young Adult/Adult/Middle Age/Senior
- Gender: Male/Female
- Occupation: Limited to general use only.
- Language: English

ADMINISTRATOR

Cognitive ability:

- Educational level: It may be graduate or post graduate level.
- Computer literacy: For system, 3-5. For application, 4-5.
- Typing skills: Intermediate to Expert typing skills are acceptable.
- Domain knowledge: Should be expert
- Cognitive style: Auditory or graphical

Physical ability:

- Visual: Should have clear vision.
- Colour vision: Normal
- Auditory: Normal
- Haptic: Fully functional

Individual profile:

- Age: Young Adult/Adult/Middle Age/Senior
- Gender: Male/Female

- Occupation: Computer engineer/Tech person
- Language: English

Categorization of tasks:

Name of the task	**Categorization of task**
Face detection	Tough
Face verification and authentication	Moderate
Cursor control	Moderate
Logs of faces per frame	Moderate

MHP Model:

MHP for Activating/Deactivating cursor control

Task involved (Using motor action)

Mental preparation (1.2 s) → mouth open (1 s + 70 ms) → system response (2 s) → Activation/Deactivation message (100 ms) → mouth relax (1 s + 70 ms)

Time = (1.2 + 0.07 + 1 + 0.1 + 2 + 0.07 + 1) s = 5.44 s

Task involved (Using voice command)

Mental preparation (1.2 s) → give command (1.1 s + 70 ms) → command detection by computer (3.5 s) → processing command (2.1 s) → Activate/Deactivate massage (100 ms)

Time = (1.2 + 0.07 + 1.1 + 3.5 + 2.1 + 0.1) s = 8.07 s

MHP for scrolling action (Scroll 10 lines on a webpage)

Task involved (Using motor action)

Squint your eyes (100 ms) → wait for activation, system response (2 s) → Turn your head up/down (100 ms) → system time to perform action (2 s) relax your head (100 ms)

Time = (0.1 + 2 + 0.1 + 2 + 0.1) s = 4.3 s

Task involved (Using voice command)

Mental preparation (1.2 s) → motor processing and action for voice (1.1 s) → command detection (3.5 s) → performing action, scrolling line by line (0.2 s X 10 lines) → done scrolling

Time = (1.2 + 1.1 + 0.2 X 10 + 3.5) s = 7.8 s

MHP Comparison table

Activating/Deactivating cursor control	Time
Using motor action	5.44s
Using voice command	8.07s

Scrolling action (Scroll 10 line on a webpage with standard line spacing)	Time
Using motor action	4.3s
Using voice command	7.8s

Inference: As the time to perform the task is less by performing that task using motor action than voice commands, this method has been chosen. Also, the success rate for the action to be completed would be better in the selected method as voice recognition and processing is prone to errors.

CMN-GOMS model:

GOMS Model for scrolling

GOAL: Scroll the currently active window

GOAL: Activate scrolling

Operator: Turn your head in the direction where the mouse pointer is desired.

Operator: Squint your eyes for a while.

Operator: Wait for scrolling to activate.

GOAL: Scroll the window

GOAL: Scroll the window upwards

Operator: Turn your head upwards.

Operator: Lower your head in relaxed position at desired point.

GOAL: Scroll the window downwards

Operator: Turn your head downwards.

Operator: Lower your head in relaxed position at desired point.

GOAL: Deactivate scrolling

Operator: Squint your eyes for a while.

Operator: Wait for scrolling to deactivate.

GOMS Model for Activating/Deactivating mouse cursor control

GOAL: Activate mouse cursor control

Operator: Wide open your mouth.

Operator: Wait for Activation, then relax.

GOAL: Deactivate mouse cursor control

Operator: Wide open your mouth.

Operator: Wait for Activation, then relax.

GOMS Model for click action

GOAL: Right click action

Operator: Activate mouse control.

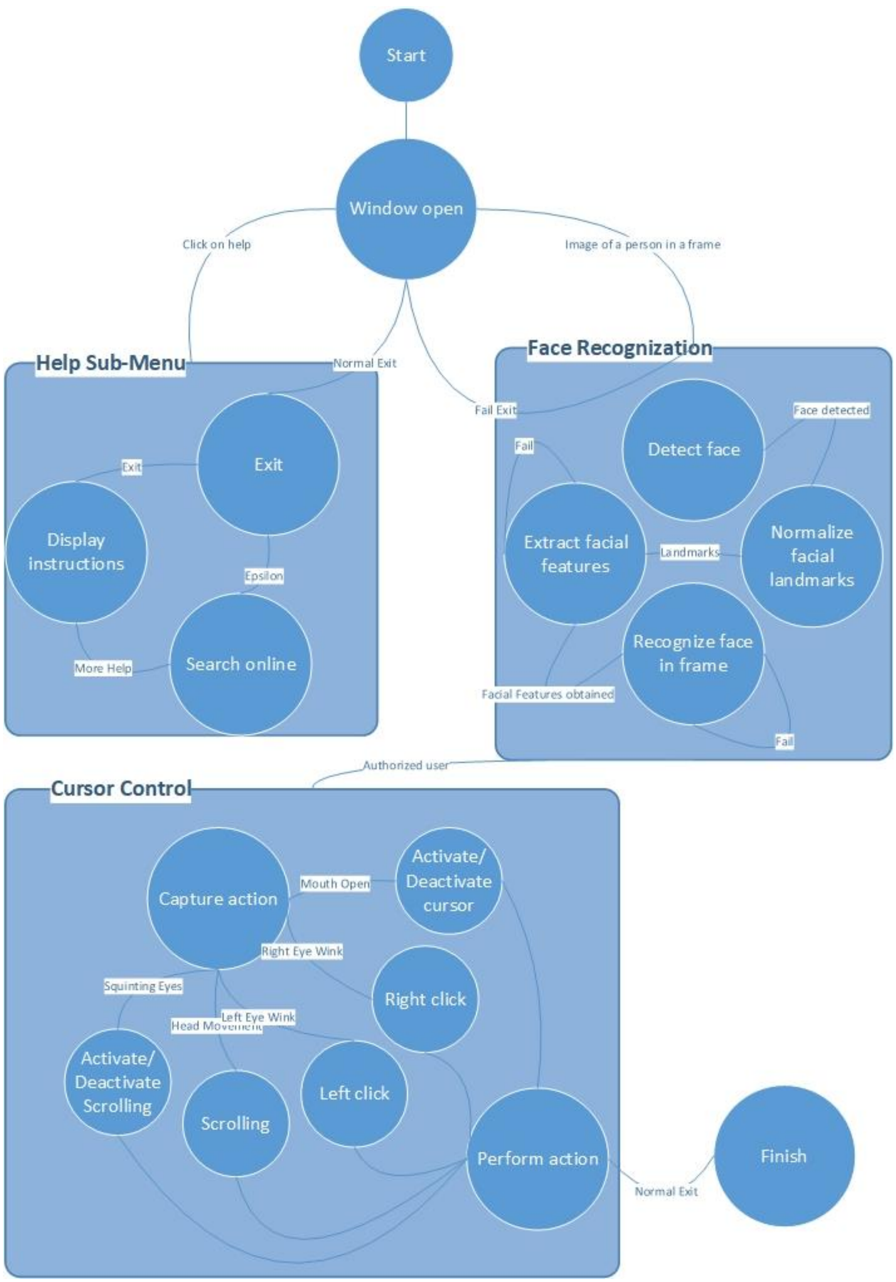
Operator: Wink right eye.

GOAL: Left click action

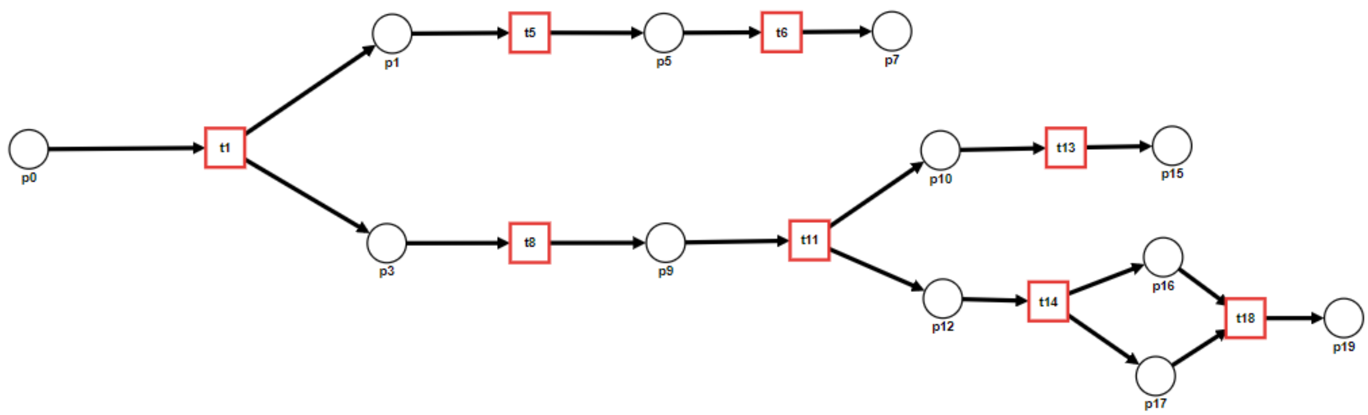
Operator: Activate mouse control.

Operator: Wink Left eye.

State Transition Network:



Petri-net:



Where,

P0-Windows open

T1- Detecting face

P1-Output "Unknown"

P2-Output Name

T5-Click "help"

T8- Authorization

P5- Make a new entry

T6-Store in database

P7-Finish, try again

P9- Capture Activated

T11-Record movements and activate cursor control

P12-Capture actions

P10-Output "try again"

T13-Deactivate cursor control

P15-Finish

T14-Perform actions

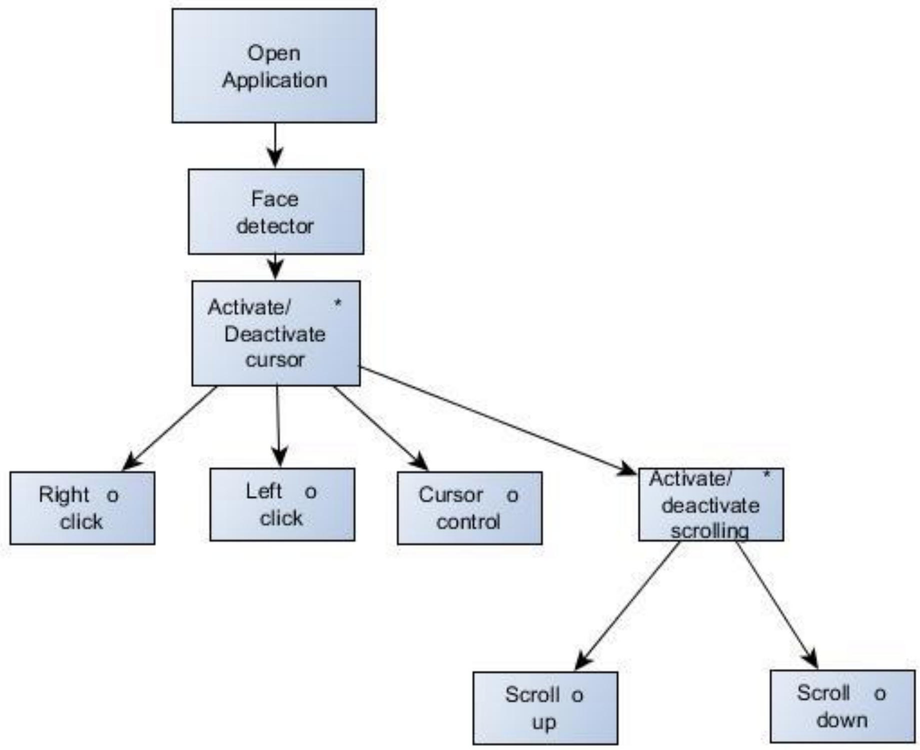
P16-Scrolling actions

P17-Clicking action

T18-Deactivate Cursor control

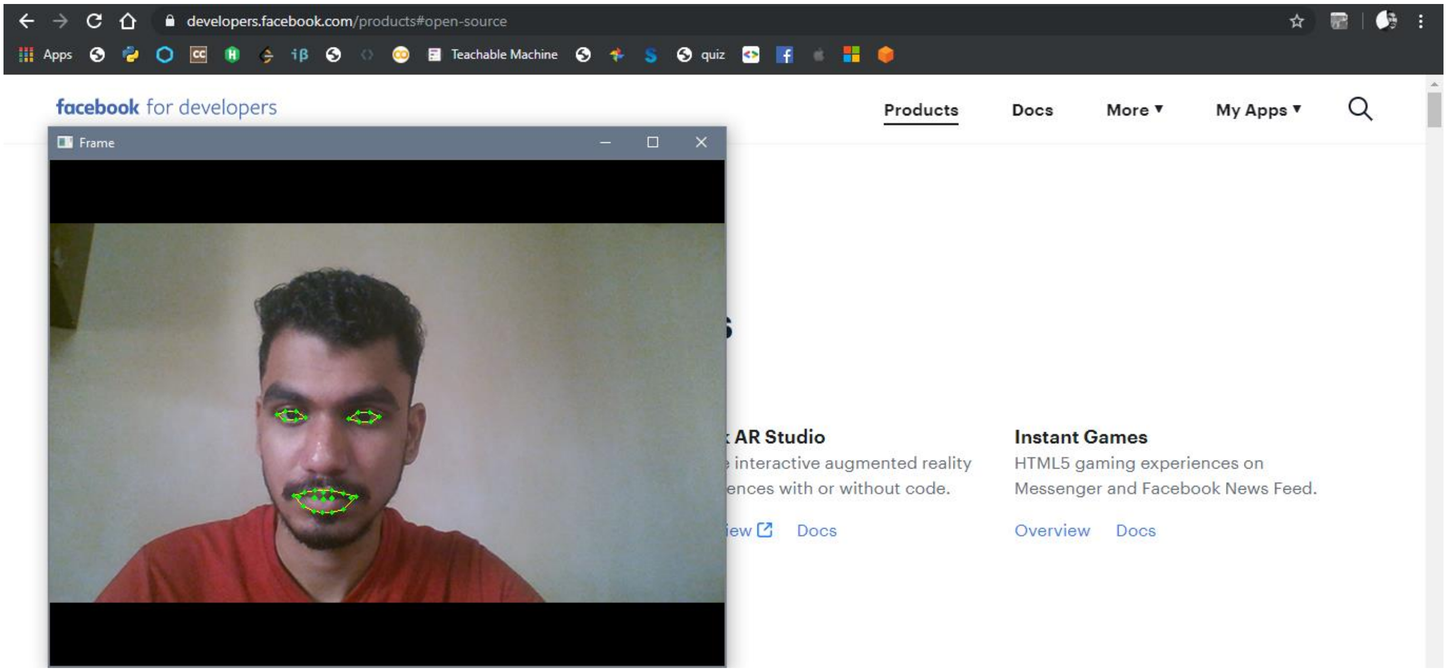
P19-Finish

Jackson Structure Diagram:

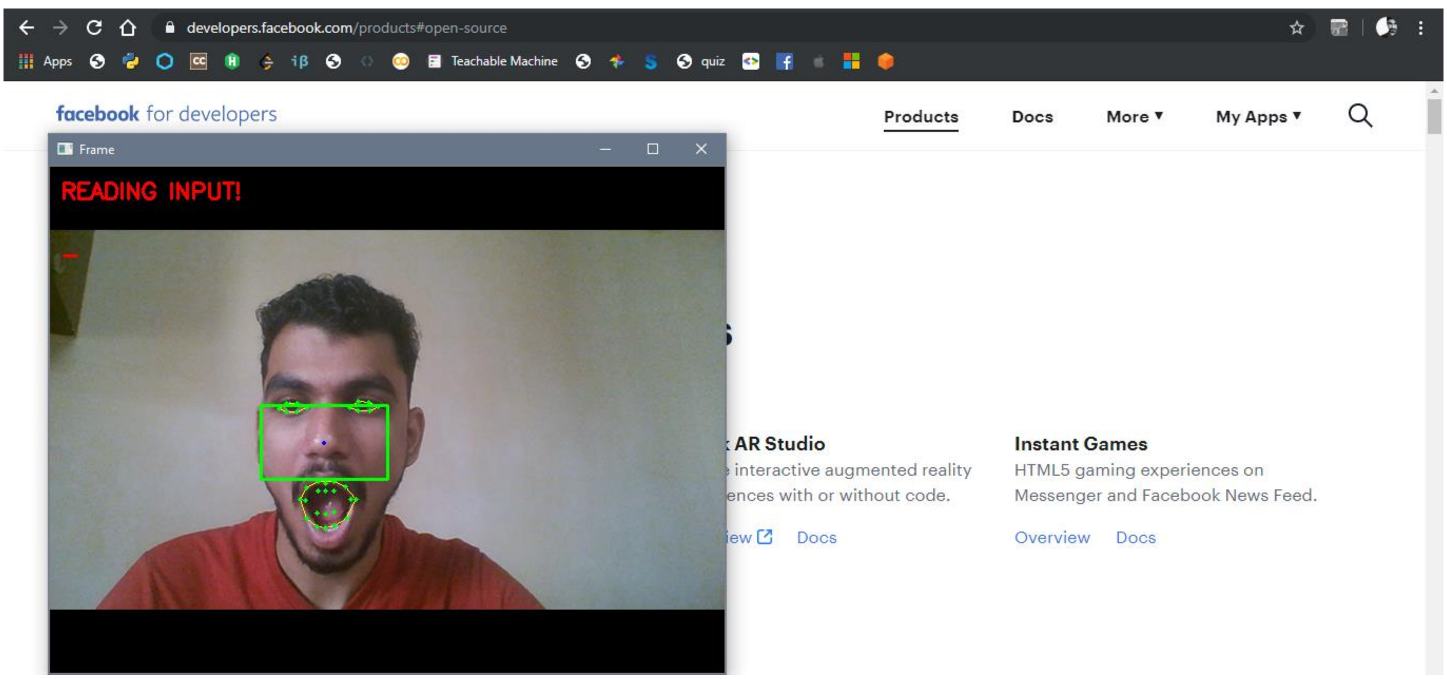


Screenshots:

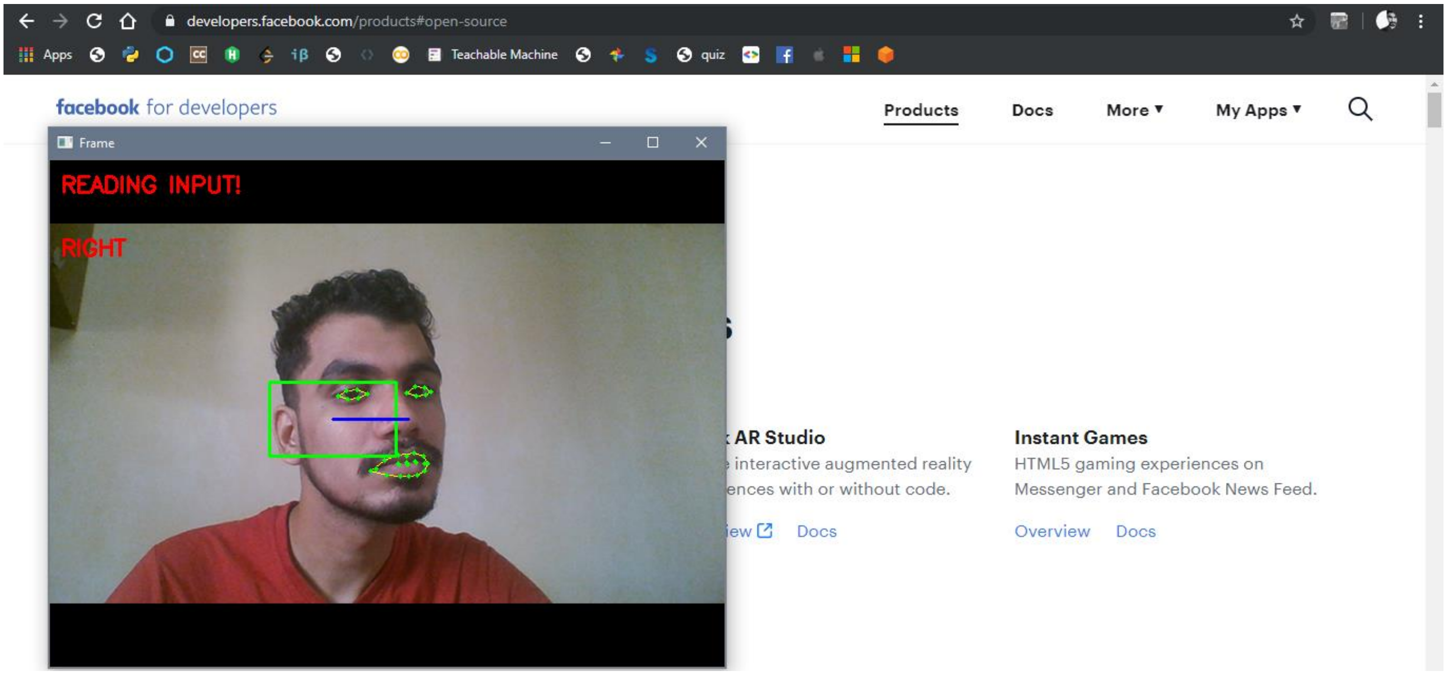
For the demonstration of the application we have opened a webpage in the background. However, the application also works when minimized.



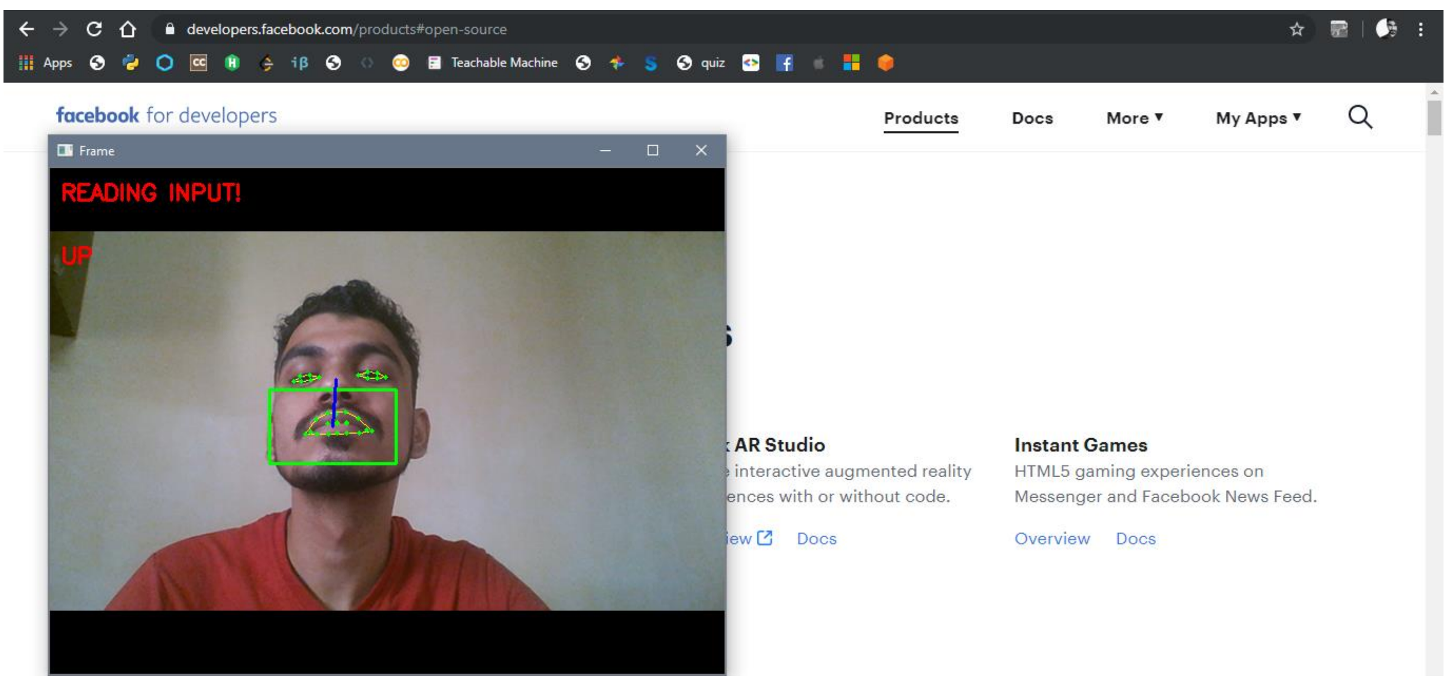
In order to activate cursor-control we have to open our mouth, when the application detected this action it displayed "READING INPUT!" on top left corner and cursor-control is activated.



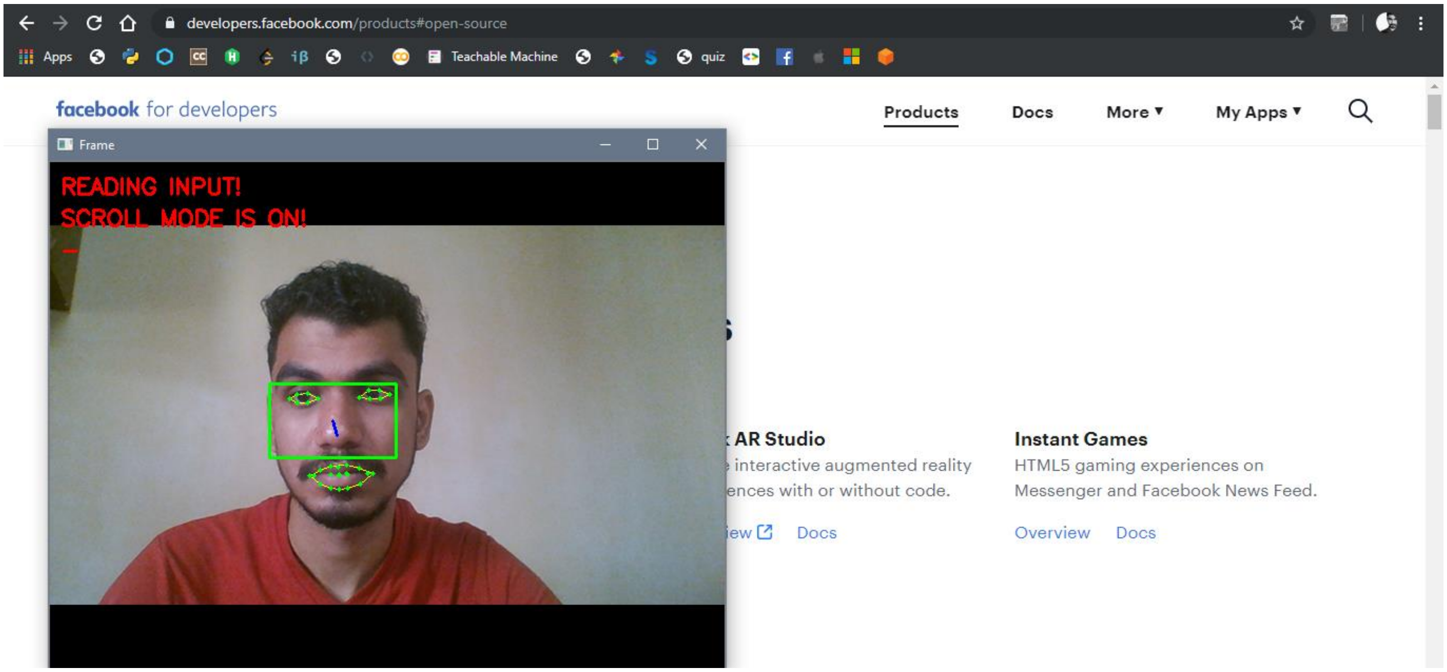
Now, if we turn our head towards right the application detects this action, and move the cursor right and indicate the same on top left corner.



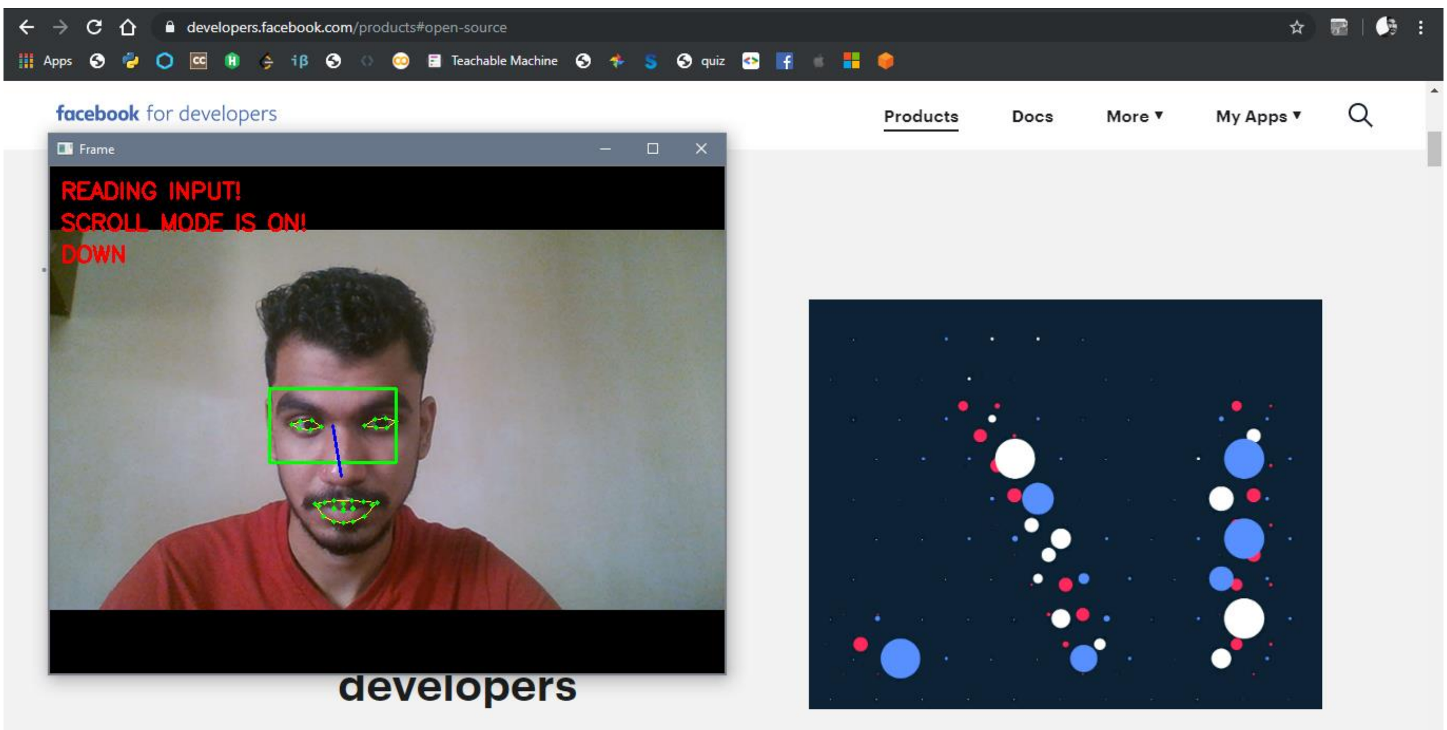
And if we turn our head up, the application detects this action, and move the cursor upwards and indicate the same on top left corner.



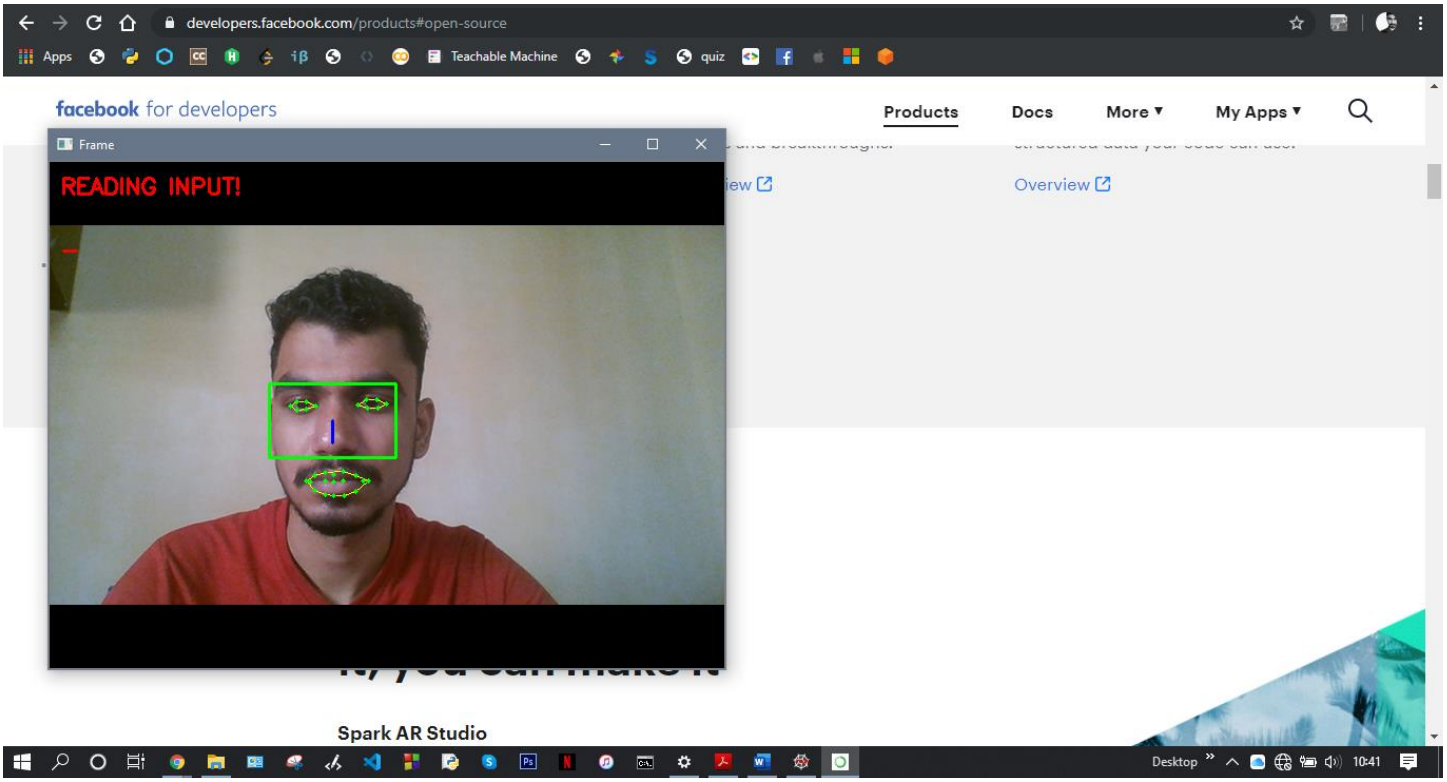
If we squint our eyes, the system detects this action and actives scrolling mode. The same is indicated in top left corner.



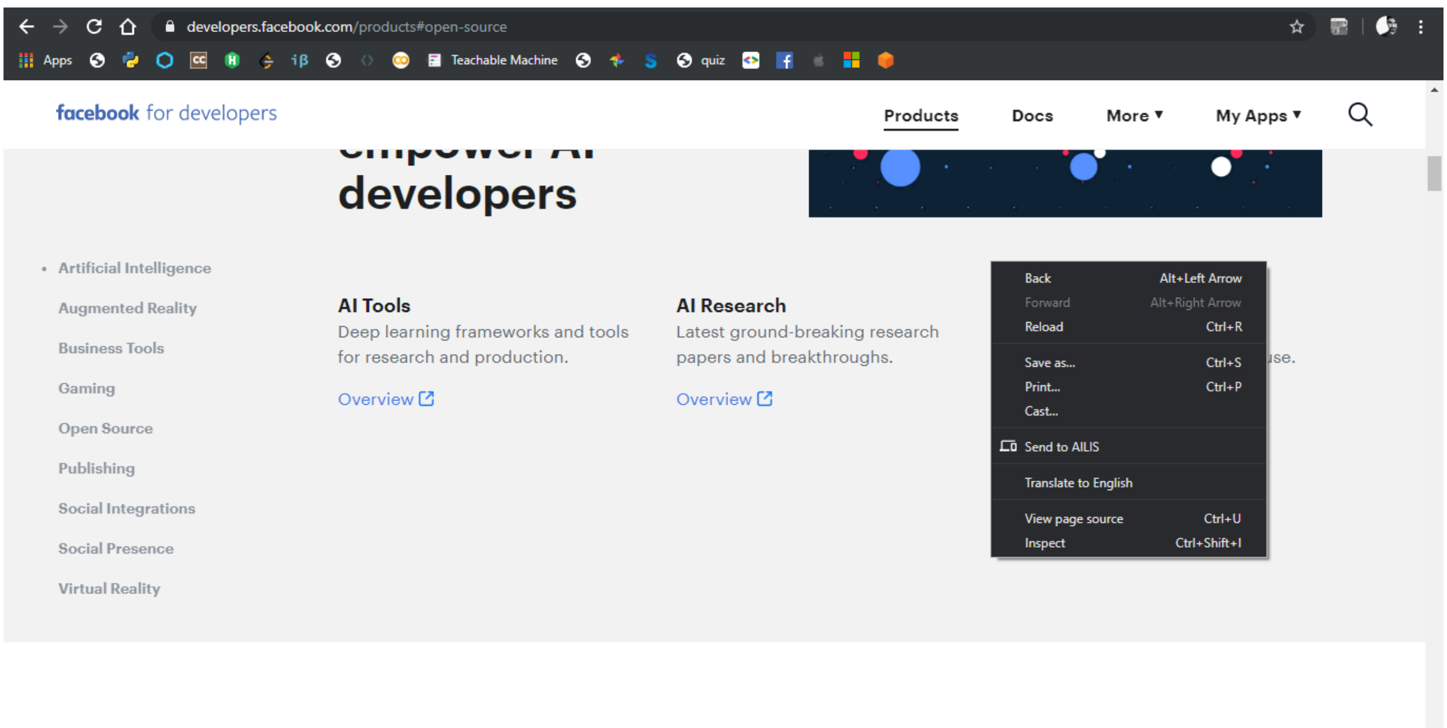
While the scroll mode is activated, if we turn our head down the window scrolls downwards and if we turn our head up then the window scrolls upwards.



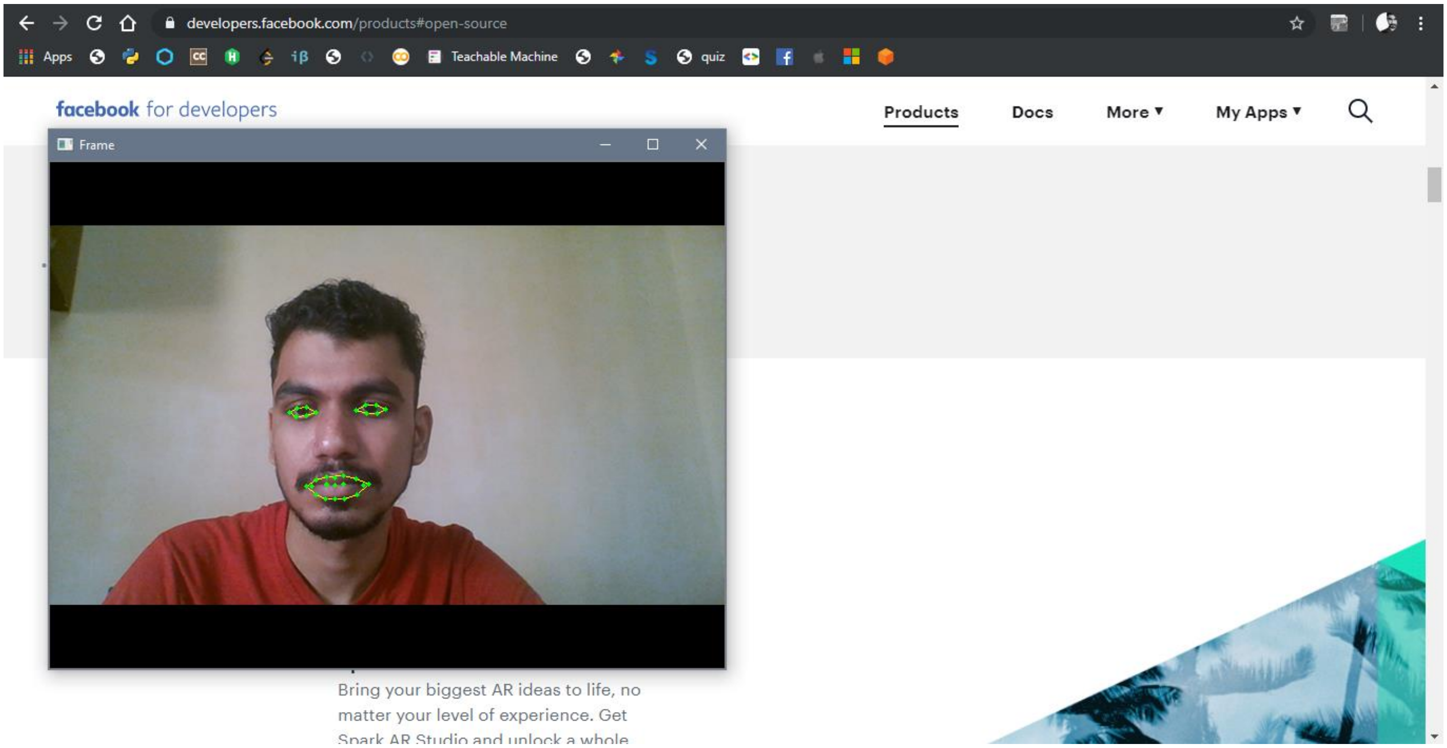
If we squint our eyes again, this action causes deactivation of scroll mode.



If we wink our right eye this will right click on page, and left eye wink for left click. (When I winked, the application detected this action and right clicked on the webpage, the browser becomes the currently active window and our application hides behind it, that's why not visible in the screenshot)



If we open our mouth again, this action deactivates the cursor-control.



Test Report:

Test Case ID	Test Objective	Test Data	Expected Results	Actual Results	Test Pass/Fail
001	To check whether the application open up with face detector or not.	Open or run the application. Either manually or through windows startup action.	Application should open with face detection through webcam.	Application opens with face detection through webcam.	Pass
002	To check whether application detect face or not.	Face input through camera.	If face is unknown then the same should be displayed on screen else the access should be given.	If face is unknown then the application displays "unknown" on screen else the access is given.	Pass
003	To check whether the cursor-control can be activated or not.	Face input through camera and mouth open as user action.	Cursor-control should activate and the same should be indicated in top left corner.	Cursor-control activated and the same is indicated in top left corner. However **sometimes** fails to detect this action.	Pass **(60 percent of time)**
004	To check whether the cursor can be controlled using head movements.	Face input through camera and head movement towards right, left, up and down.	When the head is turned towards right the cursor should move towards right, When the head is turned towards left the cursor should move towards left, up for upwards and down for downwards.	When the head is turned towards right the cursor moved towards right, When the head is turned towards left the cursor moved towards left, up for upwards and down for downwards.	Pass
005	To check whether the scrolling action can be activated or not.	Face input through camera and eyes squinting as user action.	Scrolling action should activate and the same should be indicated in top left corner.	Scrolling action activated and the same is indicated in top left corner.	Pass
006	To check whether the current window can be scrolled or not.	Face input through camera and within the scroll mode user must turn his/her head up or down.	Current window should scroll up if head is turned up and should scroll down if head is turned down.	Current window scrolls up if head is turned up and scrolls down if head is turned down.	Pass
007	To check whether the scrolling action can be deactivated or not.	Face input through camera and eyes squinting as user action.	Scrolling action should deactivate and the same should be indicated in top left corner.	Scrolling action deactivated and the same is indicated in top left corner.	Pass
008	To check whether the right click and the left click works or not.	Right eye wink for right click and left eye wink for left click.	Right click on the current window should be triggered when right eye is winked and left click should be triggered when left eye is winked.	Right click on the current window is triggered **sometimes** when right eye is winked and left click is triggered when left eye is winked. The model fails most of the time to	Fail

				detect this action.	
009	To check whether the cursor-control can be activated or not.	Face input through camera and mouth open as user action.	Cursor-control should activate and the same should be indicated in top left corner.	Cursor-control activated and the same is indicated in top left corner.	Pass

Final Outcome of the Project:

Conclusion

Final Outcome of the project is a product which can be used efficiently to control mouse cursor with facial movements. The actions done by the human maps to certain functions that the computer has to perform in an interactive environment. The software is simple to use and to understand for general public.

Limitations

As it is already stated that the software is simple to use by the peoples it is intended for, but in some cases, it is certainly not easy to work with. There are certain limitations of the models used like dlib's model to extract facial landmarks and face recognition model to detect face. We also found difficulties to perform various actions like mouse clicks and sometimes activation or deactivation of various functions. These limitations are noted and left for future development.

Future Scope

The future scope of the software only limited to the people for whom it is originally intended to. Further development can be done in improving the performance of various module. As there are other product and solutions with the people with similar stakeholder profile like voice command and eye-tracking software, this product can always prove to be a cheaper alternative to those. It does not demand any extra hardware and can easily be used with existing system.