Bucknell UNIVERSITY

Mobile App Development to Support Bell's Palsy Diagnosis

Nick Johnston

Advisors: Dr. Keith Buffinton[1], Dr. Joshua Stough[2], Dr. Arun Gadre[3]

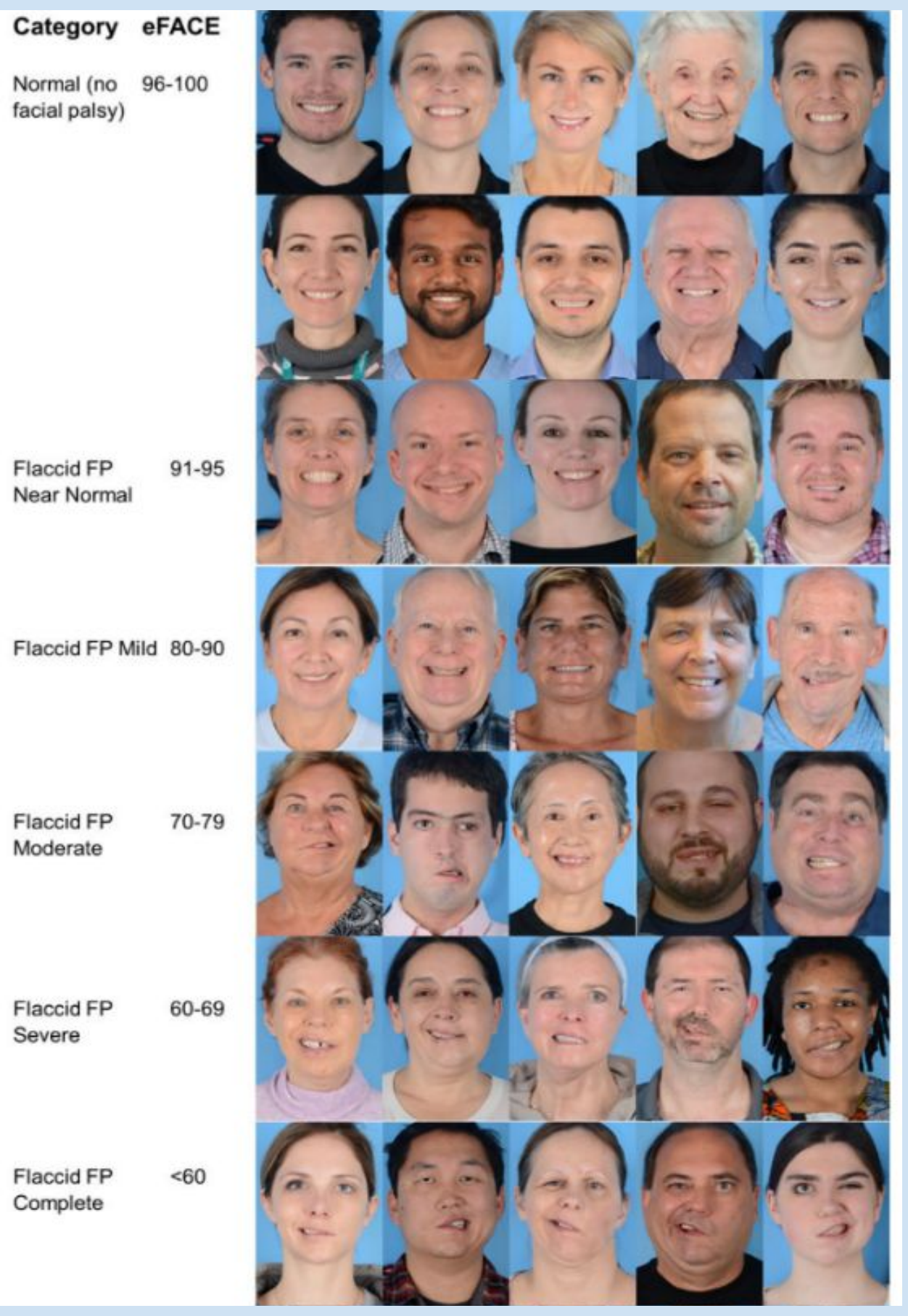
[1]Department of Mechanical Engineering / [2]Department Computer Science, Bucknell University, Lewisburg, PA

[3]Geisinger Commonwealth Medical School, Geisinger Medical Center

Geisinger

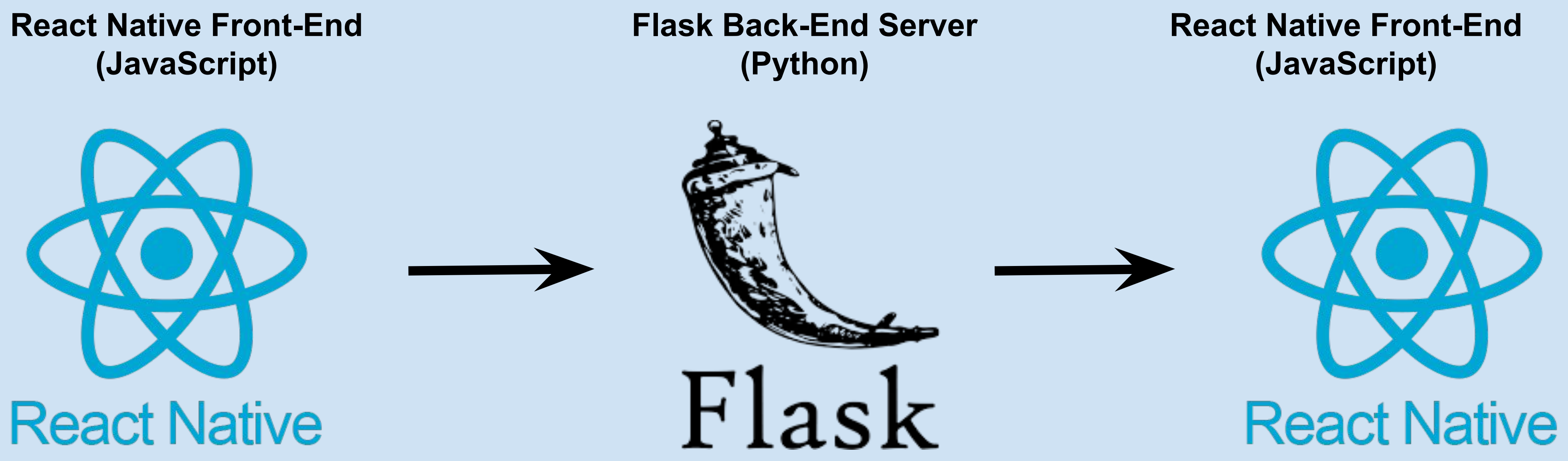
Background & Motivation

- **Goal**: Develop a mobile application capable of using computer vision and machine learning to assist clinicians in their diagnosis of patients suffering from facial nerve paralysis (FNP).
- **Observer bias** commonly arises when FNP patients are seen and diagnosed by clinicians[1], which showed that a machine learning (ML) based approach found less facial asymmetry in severe FNP patients and more asymmetry in healthy faces than clinicians. Meaning there is a tendency to over diagnose severe cases and under diagnose minor cases



Category	eFACE
Normal (no facial palsy)	96-100
Flaccid FP Near Normal	91-95
Flaccid FP Mild	80-90
Flaccid FP Moderate	70-79
Flaccid FP Severe	60-69
Flaccid FP Complete	<60

App Infrastructure

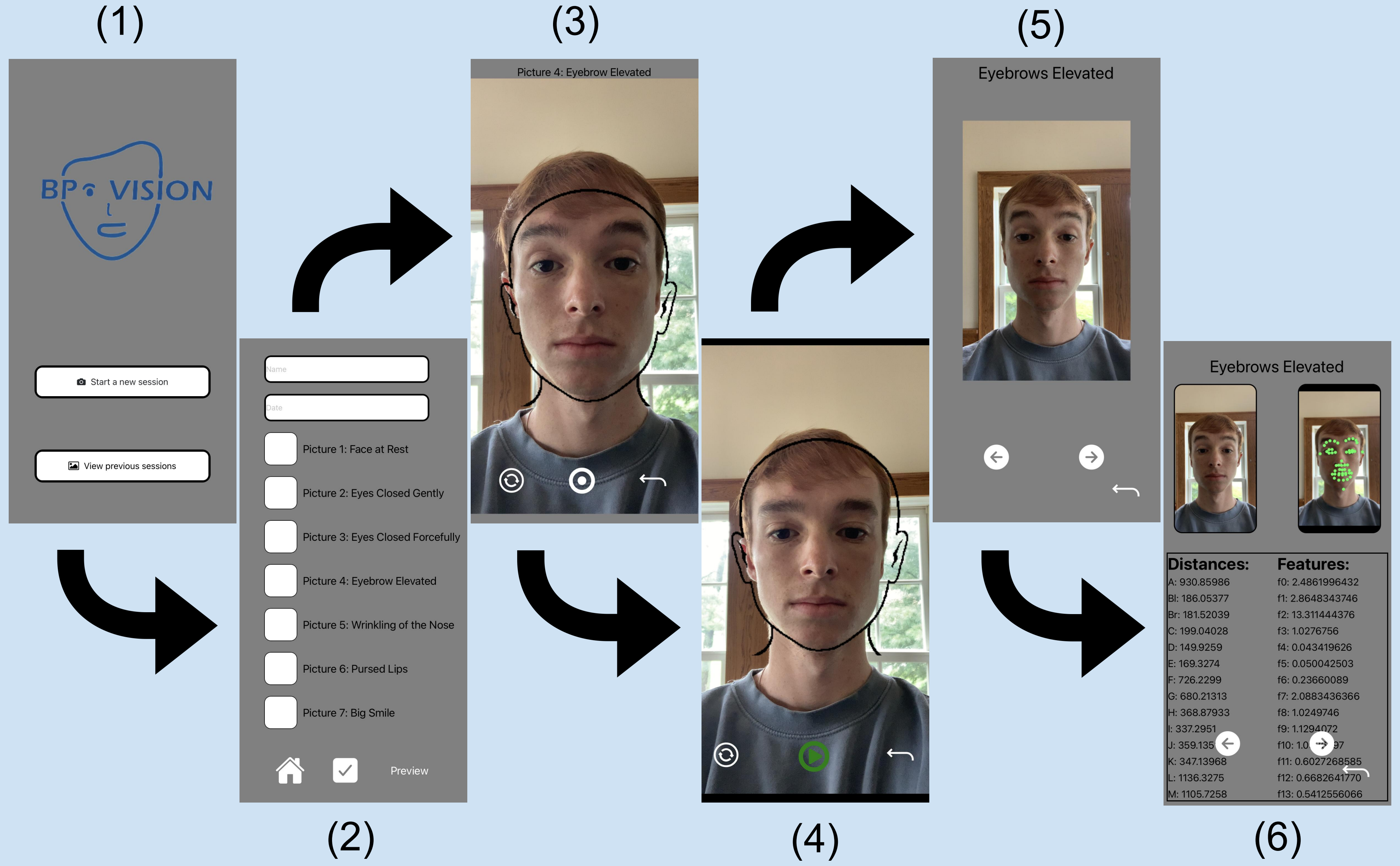


React Native Front-End (JavaScript)
React Native
Flask Back-End Server (Python)
Flask
React Native Front-End (JavaScript)
React Native

- The mobile application was built using the open source software framework, *React Native* [2], which allows developers to simultaneously develop applications for IOS and Android devices using a single code base written in JavaScript.
- The computer vision and machine learning algorithms [3] have been developed in Python which posed new challenges as JavaScript and Python are not able to communicate directly with each other. To solve this problem a back-end server was developed using a Python-based Flask server [4] to allow for real-time communication between the two languages.
- With this framework in place, users are able to send data created in React Native to the Flask server where appropriate computations are performed before sending the data back to React Native, where the results are displayed.

General App Flow

1) The home screen offers two options: "Start a new session" and "View previous session"
2) "Start a new session" leads to a page to input patient details and seven photos using different facial expressions.
3) To input photos, the user can either select from their camera roll or take a new picture using the provided camera.
4) After all the photos have been taken, the user is then prompted to record a video.
5) Once the video has been taken and confirmed, the user is brought to a final preview screen to confirm all seven pictures and the video have been correctly recorded.
6) The final screen displays the relevant calculations and plotted images.



Picture 1: Face at Rest
Picture 2: Eyes Closed Gently
Picture 3: Eyes Closed Forcefully
Picture 4: Eyebrow Elevated
Picture 5: Wrinkling of the Nose
Picture 6: Pursed Lips
Picture 7: Big Smile

Picture 4: Eyebrow Elevated

Distances:	Features:
A: 930.85986	f0: 2.4861996432
Bl: 186.05377	f1: 2.8648343746
Br: 181.52039	f2: 13.311444376
C: 199.04028	f3: 1.0276756
D: 149.9259	f4: 0.043419626
E: 169.3274	f5: 0.050042503
F: 726.2299	f6: 0.23660089
G: 680.21313	f7: 2.0883436366
H: 368.87933	f8: 1.0249746
I: 337.2951	f9: 1.1294072
J: 359.135	f10: 1.0[illegible]97
K: 347.13968	f11: 0.6027268585
L: 1136.3275	f12: 0.6682641770
M: 1105.7258	f13: 0.5412556066

User Interface and Design

The user interface has been designed to effectively present all the necessary information and interactive aspects, given the limited space available on a mobile device.

- Using icons for buttons instead of text saves space and gives the app a more polished look.
- Contrasting colors and functional text to speech and icon to speech, makes the app more accessible to those with disabilities

Privacy and Medical Data

This app works with medical information which is sensitive data, as such the following steps were taken to reduce the associated risks.

- The app is built so that information relating to a patient is never directly saved. All images are stored via links to the location on the doctors phone, which can be removed at the doctor's discretion, and videos are deleted shortly after analyzing them.
- Additionally, the back-end server does not save image or video data beyond the scope of the current process.
- Because personal, sensitive data is only ever saved locally on the doctor's device the app leverages the advanced security systems that companies like Apple and Google already have in place to protect the patients' information.

Future Work

Now that the fundamental functional goals for the app have been implemented, future research can focus on general improvements and refining the infrastructure, including:

- Converting the developmental back-end server to a full-time working server.
- Incorporating updates made to the machine learning algorithms to achieve more applicable results.
- Developing faster ways to process image and video data for increased efficiency.
- User testing to polish interfaces and make the app more user friendly.

References

- [1] Miller, Matthew Q., *et al.* "The Auto-eFACE: Machine Learning-Enhanced Program Yields Automated Facial Palsy Assessment Tool." *Plastic and reconstructive surgery* vol. 147, 2 (2021): 467-474. doi:10.1097/PRS.0000000000007572
- [2] Meta (2015). React Native (Version 0.71.8). https://reactnative.dev/
- [3] Huo, Bingnan. "Facial Nerve Paralysis Grading by Computer Vision and Machine Learning." Bucknell University, 2022. Git Repository: https://github.com/BingnanHuo/bp_research
- [4] Pallets Project (2010). Flask (Version 2.3.2) https://flask.palletsprojects.com/en/2.3.x/
- https://stackoverflow.com/