CSCI 3308 Milestone 3

Features completed:

- Front end static page server created (nginx)
- Flask back-end server created (flask)
- Homepage and various other pages created
- Browser Camera Access  / Capture
- Database / Data Model Created

What worked:

- Image capture from website
- Docker containers spinning up
- Front end linked together

Issues:

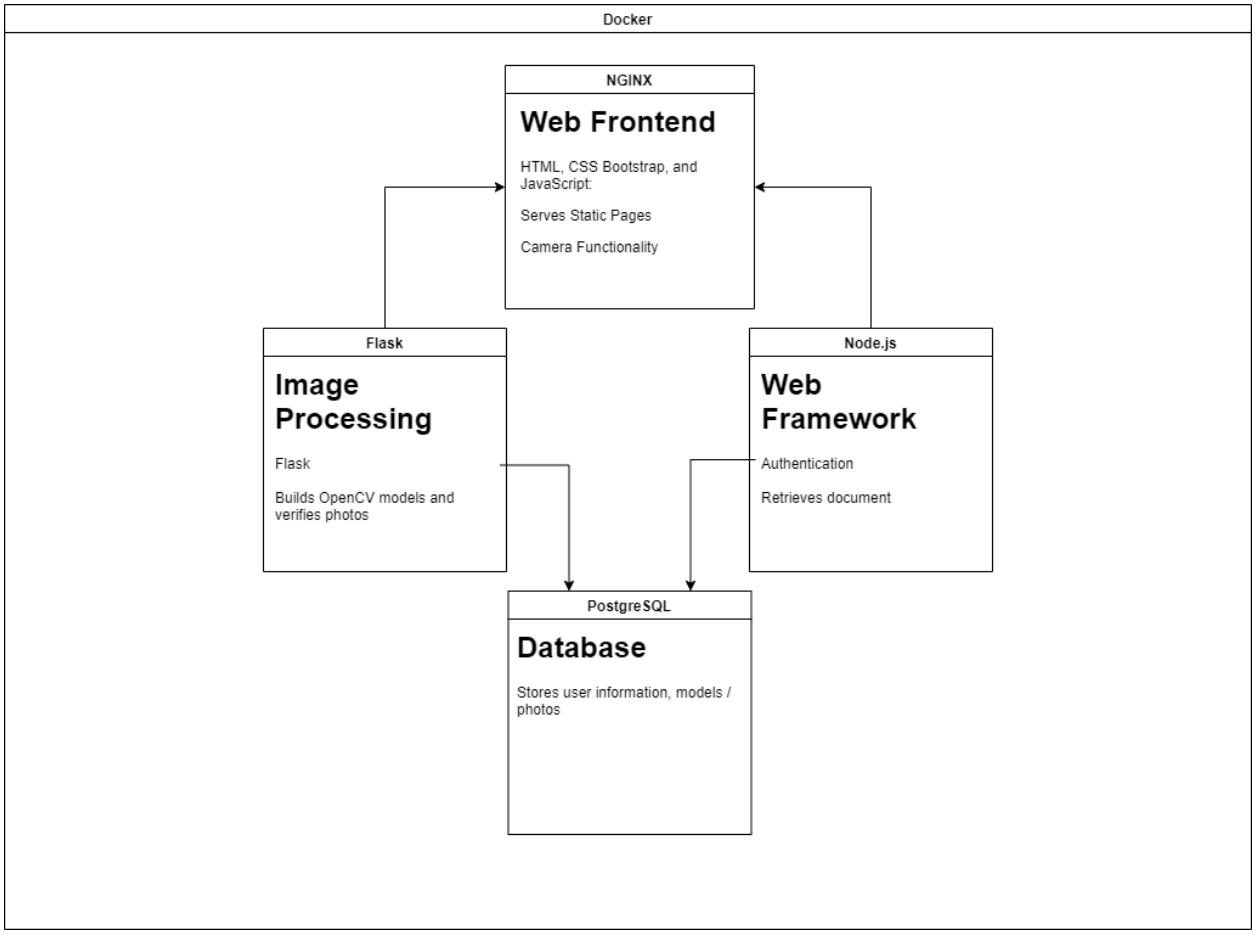
- Need to determine where/how we are storing models
- Facial recognition was struggling to distinguish faces
- Need to determine how to convert from Data URI to png and send to flask.

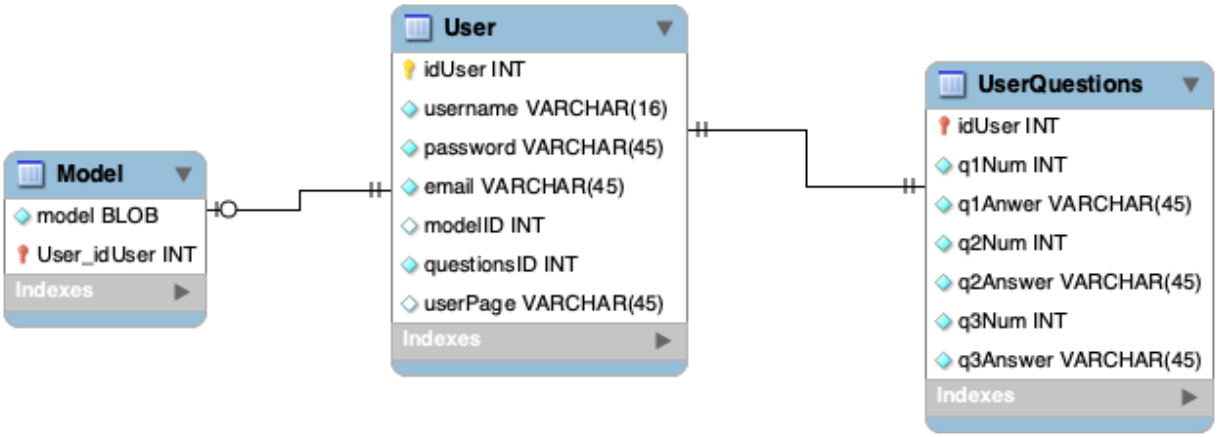
Suggestions:

- Upgrade python version
- Better understand OpenCV facial recognition
- REDIS inclusion

Individual Contributions:

- Greg: Front end coding and some database creation work (Created Data model and Create statements for Postgres)
- Michael: Created docker containers and rudimentary implementations for NGINX, Flask, PostgreSQL, and REDIS. Developed web page that can store images captured from camera as Data URI. Made Architecture Diagram
- Tyler: Front end website development: FAQ and About us
- Matt: Began development on python OpenCV facial recognition processing and training.


Docker
NGINX
Web Frontend
HTML, CSS Bootstrap, and JavaScript:
Serves Static Pages
Camera Functionality
Flask
Image Processing
Flask
Builds OpenCV models and verifies photos
Node.js
Web Framework
Authentication
Retrieves document
PostgreSQL
Database
Stores user information, models / photos


Model
model BLOB
User_idUser INT
Indexes
User
idUser INT
username VARCHAR(16)
password VARCHAR(45)
email VARCHAR(45)
modelID INT
questionsID INT
userPage VARCHAR(45)
Indexes
UserQuestions
idUser INT
q1Num INT
q1Anwer VARCHAR(45)
q2Num INT
q2Answer VARCHAR(45)
q3Num INT
q3Answer VARCHAR(45)
Indexes