# Summary

This app and corresponding backend was developed for the CU Boulder Psychology department by Ann Marie Mahon, Alex Sandridge, Connor Dowd, Garrison Lee, Han Yin, Kakam Chen, and Keyu Wu as their senior capstone project. The project sponsor was Joshua Correll.

The app is a series of learning, training, and assessment tasks to train a user to reduce their “Cross-race Recognition Deficit” (CRD). User data and task scores are stored in the backend, and the latter are made available to the Psychology department through Google Sheets.

The project is maintained on Github at [https://github.com/asandridge/psych-capstone/](https://github.com/asandridge/psych-capstone/tree/tokens)

# How the app works

The user first registers using an email and password. They will then be taken to a “pre-assessment” to measure their CRD before having used the app. After they complete the pre-assessment they will be able to complete one level per day, up to level 8.

Each level is based around 8 faces. These faces and corresponding names will be taught to the user in a “learning task.” After the learning task the user will have 4 “training tasks” for the day, “name and face,” “who’s new,” “memory,” and “shuffle.” In name and face the user is given a name, and must select the corresponding face. In who’s new the user is presented 4 faces, and must select which face they did not see in that level’s training tasks. In memory the level’s 8 faces are displayed twice each on a 4x4 board, and after 10 seconds to memorize the position the user must then “flip over” the cards to match the duplicates. A correct guess gives the user 4 points, and an incorrect guess subtracts one point. In shuffle the user is shown a sequence of 4 faces, which after 10 seconds are shuffled. The user must then rearrange the faces into the original order. The user must score >75% in each training task to continue to the assessment tasks.

There are two assessment tasks for each level. The assessment tasks do not rely on the level’s 8 faces, instead using faces unfamiliar to the user to try to provide a more even measurement. The two assessment tasks are “same different” and “forced choice.” In same-different the user is shown a face and given a few seconds to memorize it. A “mask” pattern is then briefly shown before another face is displayed, and the user selects if the face is the same as the first shown or not. In forced choice a user is shown a face and then a mask as with same-different. The user is then shown 4 faces, and must select which face they were shown first. The assessment tasks do not have a minimum score, as they are instead intended to measure the user’s CRD over time.

After completing all 8 levels the user is then given a final “pos-tassessment” in the same format as the pre-assessment.

# Architecture

The front end was developed using Ionic and Angular. The backend was developed using Node JS with the express library, and is hosted on Heroku at crossfacerecognition.herokuapp.com. The database used is Postgres, and session tokens are implemented using JSON Web Tokens. The app is deployed on both the Google Play and Apple app stores.

# Potential Areas for Improvement

* Scoring/”balance” changes. Front end change.
* The app only supports one race (black) at the moment. Front and back end changes would be required to support more.
* Emails are used as logins, but are not verified or otherwise used in any way. Primarily a backend change.
* Database connections is a possible bottleneck. Changing to one connection instance sent as a parameter or upgrading the database plan may be needed. Back end change.

# Contact Information

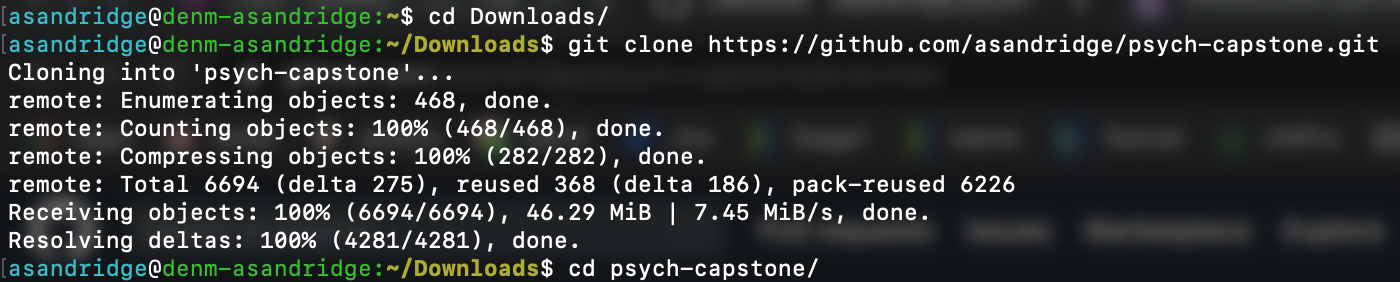
* Ann Marie Mahon (Project manager, back end): [mahon@colorado.edu](mailto:mahon@colorado.edu)
* Alex Sandridge (front end lead, back end): [alex.sandridge@gmail.com](mailto:alex.sandridge@gmail.com)
* Garrison Lee (front end, app store deployment): [gmlee970@gmail.com](mailto:gmlee970@gmail.com)

# Updating the Faces with Git

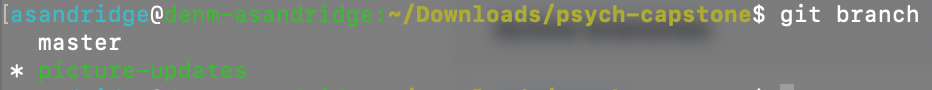
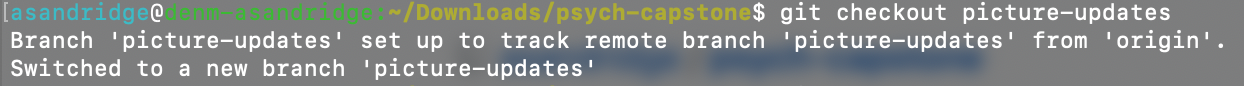
1. Make sure you are a collaborator on the repository. We will add you once you create your account.
2. Install git. <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
3. Open the terminal. Navigate to the folder you want to put the code (e.g. “cd Documents”)
4. Download the code (if first time):

“git clone <https://github.com/asandridge/psych-capstone.git>”

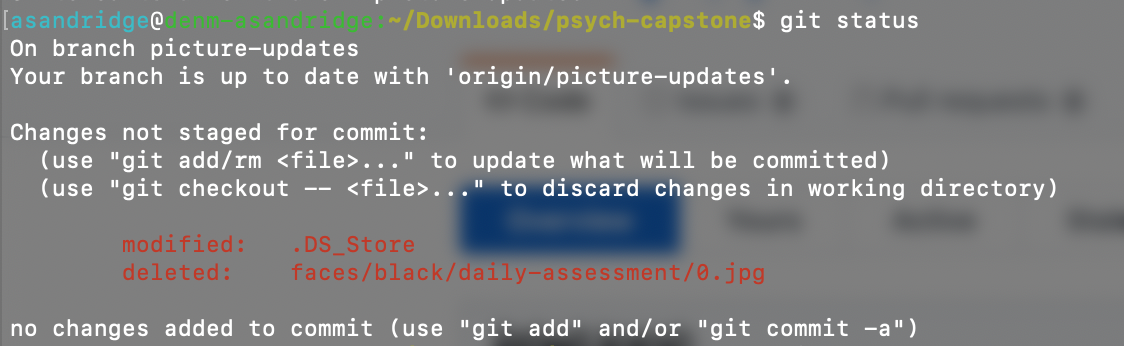
1. “cd psych-capstone”



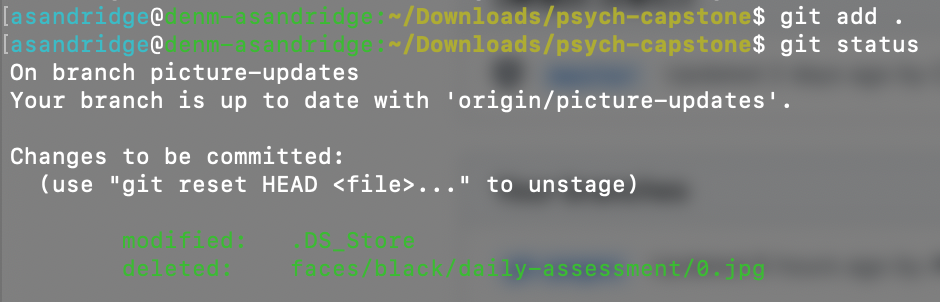
1. “git checkout picture-updates” will put you on the right branch. If it says it can’t find it, pull it down with “git pull origin picture-updates”. “git branch” will show all of the branches; make sure there is a star next to picture-updates.



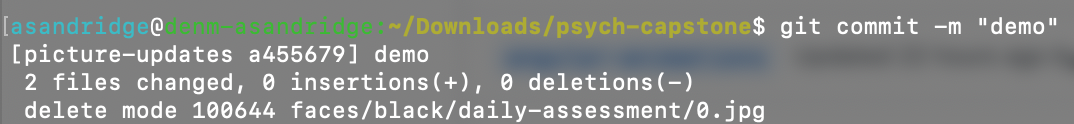
1. Change the pictures. This can be done in the regular user interface with copy and paste. Make sure the image names match ***exactly***. jpg for the assessments, png for the training.
2. “git status” - should show the updates in red



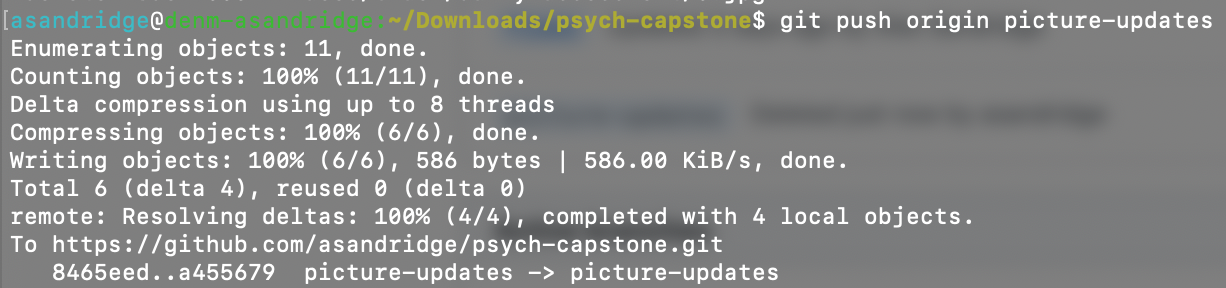
1. “git add .” - this will add all of the changes to the staging area. “git status” should show all of the image names to be green now



1. ‘git commit -m “your message”’ - this will save the changes to your local repository. Your message can be anything, just to let people know what you changed



1. “git push origin picture-updates” - this will upload the local picture-updates branch to the remote repository that we all share. Then you should be able to deploy.



1. <https://dashboard.heroku.com/apps/crossfacerecognition/deploy/github>. Scroll down to ‘Manual Deploy’. Select picture-updates from the dropdown and click ‘Deploy Branch’.
2. If something goes wrong in the deployment or git, just redeploy master and start over. You won’t be able to mess up any of the written code without an explicit approval from one of the team members.