

Team Bender

Project 3 Report

Facial Recognition System

Team Role	Name
Project Manager	Matthew Johnson
Systems Engineer	Dylan Gransee
Architect	Adam Campbell
Developer	Claude Cullen
Web Master	Wan Zulsarhan Wan Shaari

Table of Contents

Table of Contents

Problem Statement

Feasibility

Preliminary Screen Sketches

Prototype Screenshots

Requirements

Risks

GQM

Design

Design Implementations

Economics

Work Breakdown Schedule

Budget

Summary of MRs

V & V Plan

Retrospective Report

Planned future extension

Review Results

Results of reviews of artifacts, based on the V&V plan and WBS

Problem Statement

Many instructors at universities around the world all take attendance to ensure students are coming to class on a regular basis. As classes get larger, taking attendance becomes more tedious and painstaking; this ends up wasting precious lecture time where the students could be learning. In addition to being disruptive, existing methods of attendance taking (like passing around a sign-up sheet) are vulnerable to forgery.

Our development team will design and construct a facial recognition system to quickly and easily take attendance by automatically capturing images of the classroom, accurately recognizing students in the image by using a facial recognition server, and saving this information in an attendance database for instructor viewing and updating. By using our product, instructors will be able to quickly and easily keep track of attendance in their classes.

Our system will be designed to handle the classroom setting, and we mainly anticipate use by college professors for attendance purposes. We believe that our system, even though it is being designed for classroom, will be highly portable to other applications and situations where attendance records may be important.

Feasibility

The increase in computational power has made many difficult tasks possible. With modern technology, the ability to run facial recognition on a large number of faces in a short time is becoming more and more possible. Through our research and preliminary testing of the available facial recognition servers (as well as some general cameras), we determined that the ability to recognize a classroom of faces is feasible.

By restricting the project initially to a simple web app where the users upload photos taken elsewhere, we eliminate unnecessary early complexity and are able to get a working product up and running faster. Since the project is modularly designed, we can swap out 3rd party facial recognition services, or even use multiple ones to increase accuracy and recognition. Nevertheless, the service from our current provider, Betaface, is functional and easy to use, and provides accurate feedback regarding facial recognition in a timely manner. We've tested images containing from 1 to 20 faces in a variety of environments, with different facial expressions and clothing, and at various sizes in the images. Betaface is able to recognize the faces nearly every time, with few false positives. In terms of person recognition, accuracy improves with more training images, and is quite good starting at around 3 training images.

Preliminary Screen Sketches

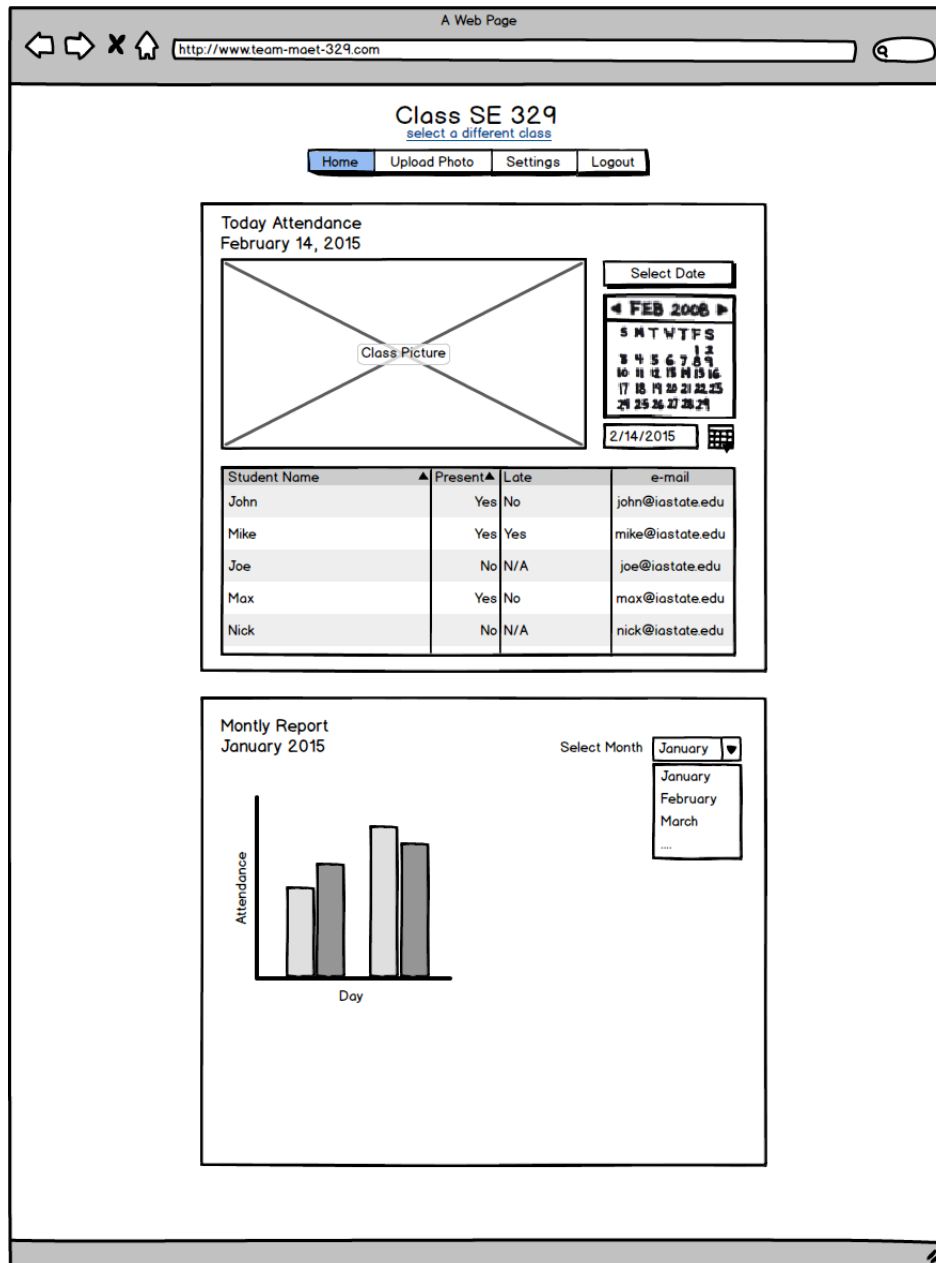


Figure 1: Landing page for teachers after login

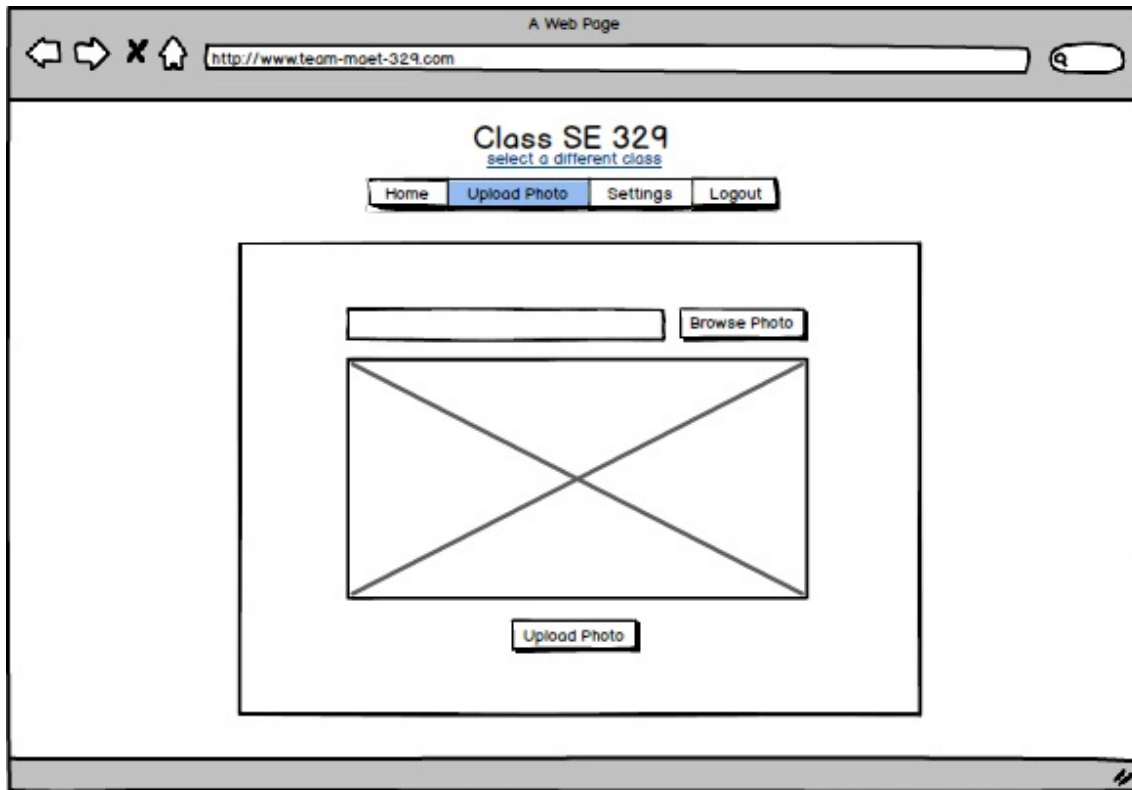


Figure 2: Upload photo page for adding class photo for a specific day.

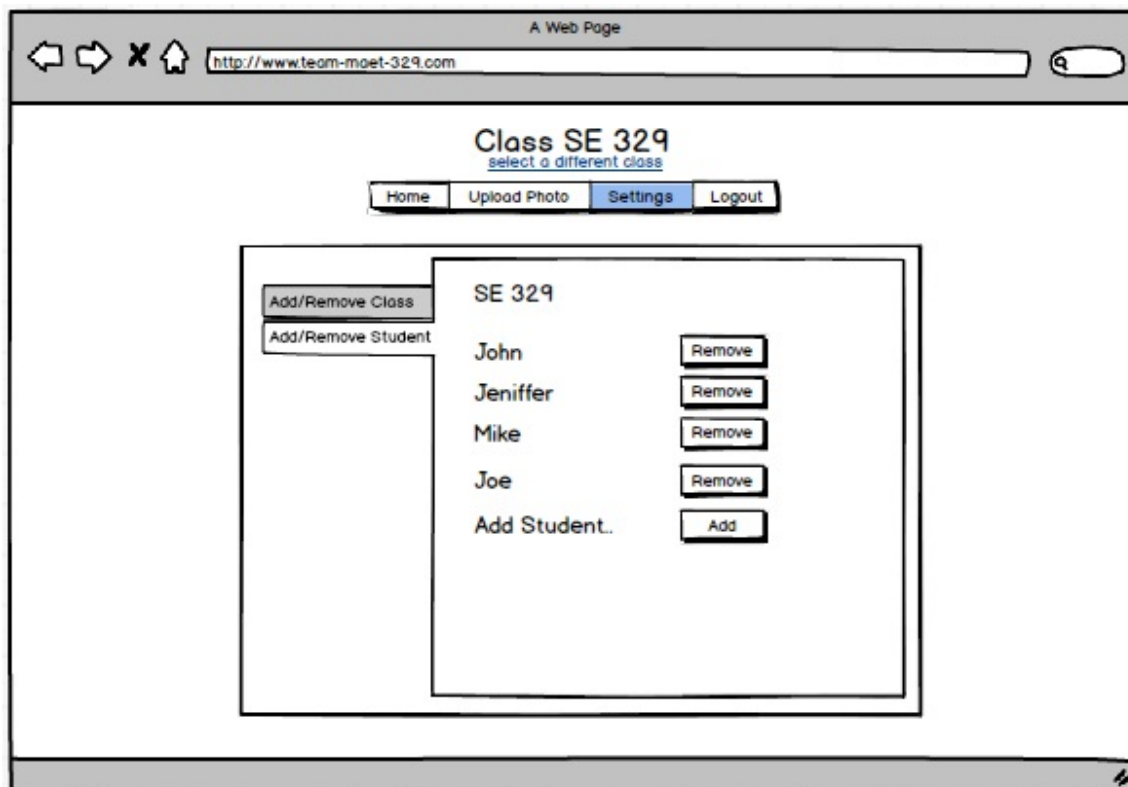


Figure 3: Settings page to modify students and classes

Prototype Screenshots

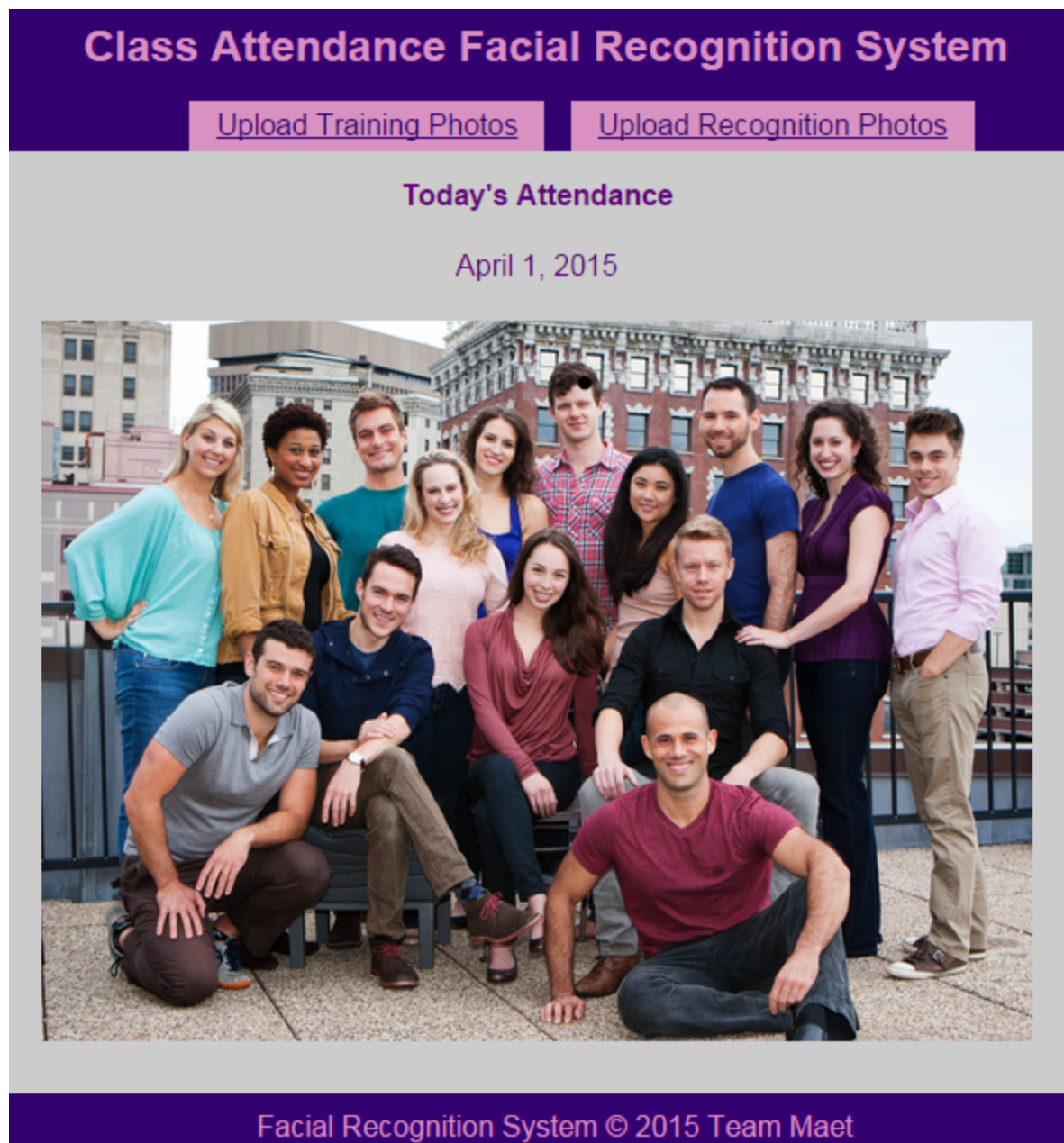
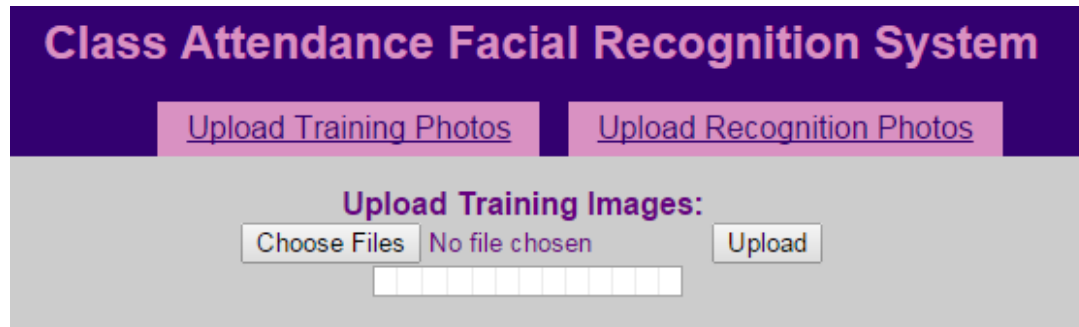


Figure 4: Prototype Homepage



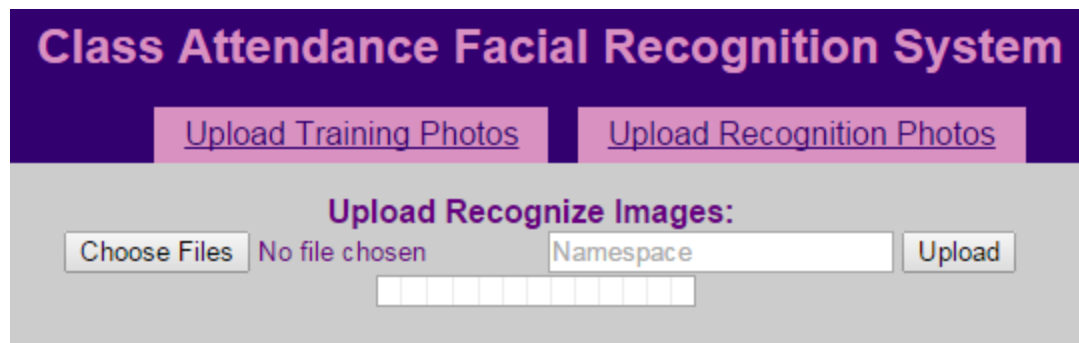
Class Attendance Facial Recognition System

[Upload Training Photos](#) [Upload Recognition Photos](#)

Upload Training Images:

Choose Files No file chosen Upload

Figure 5: Prototype Training Photo Upload Page



Class Attendance Facial Recognition System

[Upload Training Photos](#) [Upload Recognition Photos](#)

Upload Recognize Images:

Choose Files No file chosen Namespace Upload

Figure 6: Prototype Recognition Photo Upload Page

Requirements

- The server should be able to remain up for 15 hour periods, as to handle any request that may be generated between the hours of 6am to 9pm
- The server must be able to return the list of attending students to a teacher within 2 minutes of the photo being taken
- The server should be able to handle 40 classes taking attendance photos and submitting them to the server almost simultaneously
- To accurately ensure students are identified, at least 5 photos will need to be trained by a professor, but adding more photos (~10) would be best
- Attendance photos must be kept on the server for a month in case there are attendance errors that must be rectified
- A professor should be able to remove students from their classes trainers, when a student drops the course

Risks

Risks	Probability of Occurrence (0-1)	Criticality (0-100)	Risk Factor (O * C)	Mitigation Strategy
The number of simultaneous request for recognition is taxing on the server, and response times render the product unusable.	.8	80	64	We can attempt to route the request to multiple servers, lessening the number of recognitions each one has to do.
Classroom is too large to capture all the faces	.4	90	36	Place multiple cameras throughout the room (front, half way back, etc).
Students are not facing the camera	.6	45	27	System takes multiple photos throughout class.
A person who was attending a class, was not confidently recognized by the facial recognition server, and was not counted as attending.	.25	50	12.5	A service can be implemented for teachers where they can request the attendance photo for a certain day, and the student can identify themselves in the photo to prove they attended the class.
Students arriving late to class	.5	25	12.5	System takes multiple photos throughout class.
Poor network connectivity (cannot connect to servers)	.1	65	6.5	Pictures are saved locally until network connectivity is reestablished.
Teachers do not want to complete the trainer photos for their classes	.05	100	5	At the start of each semester students are able to get an updated picture taken.

				There could be a photo booth where students take images of themselves for the system.
Image quality too poor to make out faces	.05	80	4	Camera equipment will be researched and thoroughly tested to ensure it is sufficient for the size of the room.
Teachers do not want to remove students from the trainer photos for their classes	.20	10	2	An automated system could be in place, that checks the current enrollment for the class, and prunes dropped students.
Students have similar facial features (like siblings)	.01	75	.75	Require a larger confidence to recognize people with similar facial characteristics (e.g. people that get confused as each other very often).
Drastic appearance changes causes recognition to fail for that student	.01	60	.6	The system will allow for multiple training photos to be uploaded throughout the year, so any students with large changes will need to update their images.

GQM

Goals	Questions	Metrics
Reduce the time required to take attendance so that there is more class time.	How long does it take to take attendance with this system?	<p>Number of students in the class.</p> <p>Time required to take photos during class.</p> <p>Time required to check the photos with the facial recognition.</p>
	How long did it take to take attendance before?	<p>Number of students in class.</p> <p>Average time to check attendance per student.</p>
Detect multiple faces within a photo.	Can the server handle a multitude of faces within a photo?	The number of faces in a photo.
Accurately recognize faces for attendance.	Can our facial recognition server accurately match up the faces?	The number of trainer photos are available per person.
	Can the server handle an image of a classroom full of faces?	<p>Number of faces in the image.</p> <p>Size of the room (how close are the furthest students).</p> <p>Camera resolution.</p>
Ensure system is cost effective.	How much money does this system save automating attendance?	<p>Difference in time to take attendance with and without this system.</p> <p>Cost per minute of professor.</p>
	How much will it cost to implement the system?	<p>Cost to design and implement the system.</p> <p>Cost for hardware testing.</p>
	How much does it cost to maintain the system?	Cost to maintain the servers and code.

Design

The project is organized into several components. At the core of the application is the main server backend, which is responsible for communicating with the Betaface facial recognition component and storing the attendance records into the database. The web frontend interacts with the user and communicates with the backend by uploading pictures for processing and retrieving attendance results for display.

The front end has three primary actions: training, recognizing, and viewing attendance. In the training phase, the user uploads images of the class to the server backend. The backend uploads the picture to Betaface and receives a list of thumbnails of the recognized faces. The user then assigns a person ID to each face, associating the face and ID on Betaface's servers. In the recognizing phase, the user uploads an image of the class to the server, which then uses Betaface to determine which students are present and which are absent, saving the results in the database. The user can then view these records on the viewing screen.

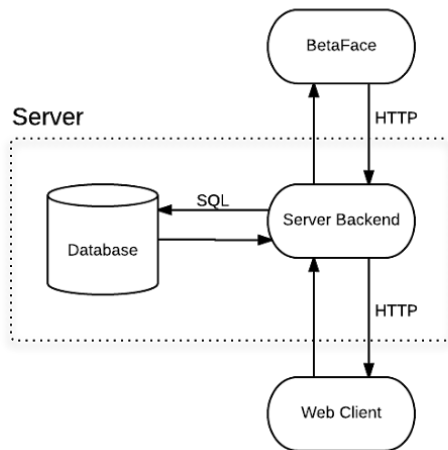


Figure 7: Modular Structure

Design Implementations

<https://github.com/awjc/Facial-Recognition-Project> (Repository)

Economics

Due to the lack of an efficient method to take attendance, our product will flourish because of its ease of use, fast response, and accuracy. With our product making the task of taking attendance easy, teachers will have more time to spend teaching students the important information they are in class to receive. Our product will also provide good documentation of the attendance for the class. If there was a dispute with a student on whether they had attended class or not on a specified day, our system would allow you to pull up images of the given day and if the student was there that day, they could point themselves out in the image. We expect teachers to want our

product and universities to notice that want and follow through with a purchase of our software. Selling for \$60 dollars per class, with an expected 40 average classes an hour, and 12 hours a day we expect to make \$9600 just from one Semester. So marketing to more than one college will yield large returns. After only two semesters we will have enough to cover our input costs, and maintenance costs for the systems and their servers.

Work Breakdown Schedule

<https://app.smartsheet.com/b/publish?EQBCT=e93478980f3642329fcc897dca631eb0>

Budget

Item		Quantity	Unit Cost	Units	Duration	Total Cost
API Keys		1	\$500	X	X	\$500
Testing Hardware					X	
	Cameras	30	\$30	per camera	X	\$900
	Camera Controllers	30	\$10	per controller	X	\$300
Servers		1	\$500	per month	3 months	\$1,500
Miscellaneous		X	X	X	X	\$1,000
Employee Pay						
	Project Manager	1	\$830	per month	3 months	\$2,490
	Systems Engineer	1	\$750	per month	3 months	\$2,250
	Architect	1	\$700	per month	3 months	\$2,100
	Tester/Integrator	1	\$650	per month	3 months	\$1,950
	Developer	1	\$650	per month	3 months	\$1,950
Total			\$4,980	per month	3 months	\$14,940

Budget takes into consideration the 3 month cost of running the project and the costs of producing 30 test systems to sell to customers.

Summary of MRs

Date	Modification Reports
2/5/15	Real time feedback for teacher usage within the class time.
2/5/15	Identifying if the image was taken during the class time.
2/10/15	Change back end server from planned Java to PHP
2/23/15	Change web layout on training page.
2/23/15	API for Application production.
3/4/15	Revision of problem statement
3/4/15	Revision of feasibility
3/4/15	Addition of unidentified risks
3/7/15	Updated work breakdown structure

3/10/15	Addition/revision of GQM items
3/10/15	Changed layout of budget to more formal table
3/24/15	Updated V&V plan
3/24/15	Added screenshots of current web application prototype
3/27/15	Added artifacts from V&V Testing
3/31/15	Updated retrospective report to reflect changes made

V & V Plan

Informal Reviews

Done by the Project Manager on a regular basis. The manager should take time everyday to try and look over the code in each module for errors. Any errors or problems that are found in the code will be brought to the attention of the team. The goal of these reviews is to increase the coding standards of the team as well as find errors in our application.

Operational Profile

Done by the Systems Engineer during the first day of his requirements updating during a sprint; takes effect after the first iteration. The operational profiling will consist of uploading multiple pictures of students in different classrooms, and making sure that their faces are recognized consistently. The application will have to work in a variety of different classrooms. This will ensure that our application will perform as expected when it is being used in real world situations.

Acceptance Testing

Done by the Systems Engineer during the first day of his requirements updating during a sprint; takes effect after the first iteration. The Systems Engineer will take the current product (or information about the product if it is not in a working state) to the professors and other potential stakeholders like College Board members and discuss with them what they like, what they don't like, and what they would like to be added. We will be utilizing Acceptance Testing to make sure that our software project meets the specifications defined by the customer, as well as ensuring that the customer is satisfied with the product we are creating.

Black and White box Testing

Created by the Tester after the Regression Tests have passed. Utilizing both should give us good code coverage as well as the best possible outcome for good testing. The tests should cover approximately 75% of the code at all times. As new code is added during each iteration, tests should be developed for that code. When previously existing code is altered, the tests that are testing that code should be verified.

Regression Testing

Ran Automatically, but checked over by the Tester for passing. These are run at the end of every iteration in order to ensure that the code has been implemented and integrated correctly. If errors are found in regression testing, the task of fixing them will be added to the next iteration.

Retrospective Report

What went well

- Had code reviews during all checkins to ensure code quality
- Documented project progress very thoroughly in both the code and the reports
- Started early working on the project and did not procrastinate. Allowed for easy balance of the work being spread out.
- Were able to perform user testing with sample images and classroom photos. This was a major milestone to have completed.

What went wrong

- Prior documentation was lacking which made it difficult to fully understand the project at first. The documentation was too general and not specific enough about key concepts.
- Model implementations took longer than expected. Initial time estimates were off and needed more time than we thought.
- Did not meet often enough as a group. We would have liked to have 1 or 2 meetings a week but we were unable to accomplish this.

Recommendations

- Start early and continuously work on the project. Gaps of no work completed makes people lose focus. Hard to start back up after lapses of non-productivity.
- Meet on a regular basis, one or more times a week.
- Keep a sprint backlog that is detailed. List milestones that have been completed and tasks that need to be completed in the future. Add bugs found as well to the backlog to monitor their status.
- Have an all team meeting very early to discuss the project. Ensure that everyone is on the same page and knows what needs to be completed and what are standards that need to be followed.

Planned future extension

Web Page

- Login and Sign-Up Page
- Setting Page

- Add/Remove Classes
- Add/Remove Students
- Data representation of the class attendance (Graph/Pie Chart)

Mobile Application

- User can upload the photo directly through the app

Database Integration

- Photos should be saved on to a database for future access.

Results of reviews of artifacts, based on the V&V plan and WBS



Figure 8: A good classroom image - students facing forward, no faces blocked



Figure 9: A bad classroom image - students facing away from camera, some faces blocked

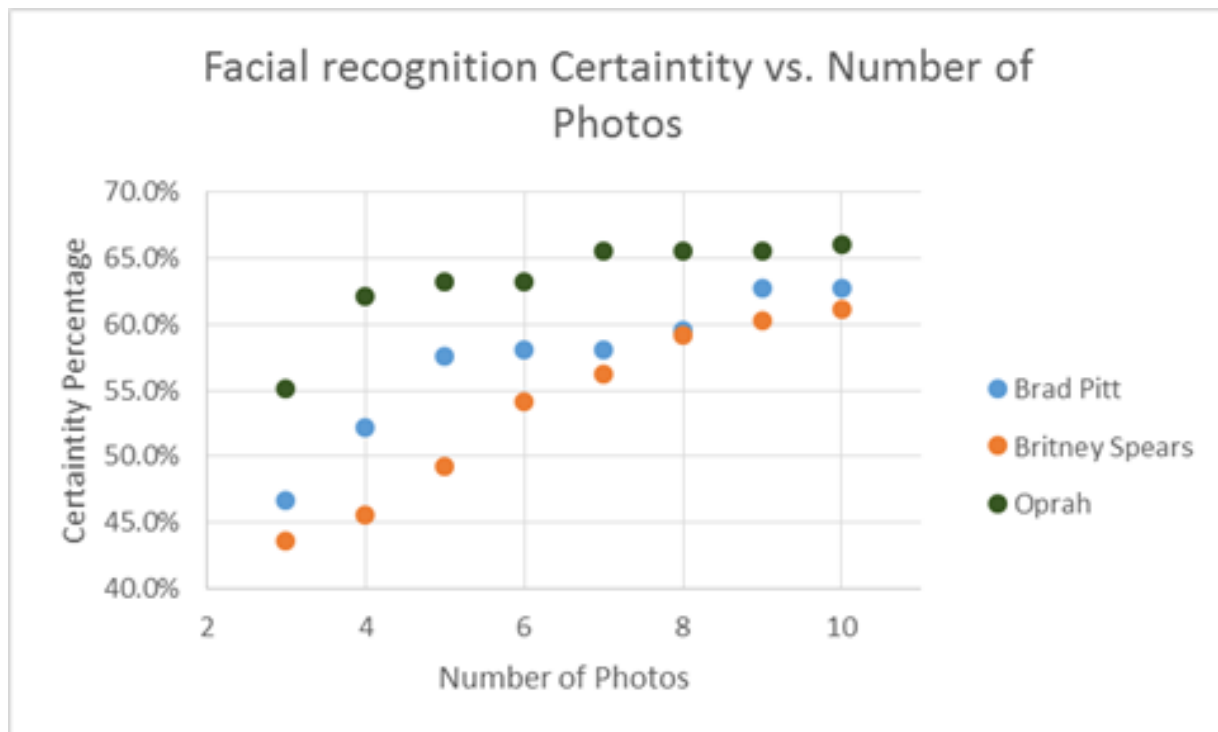


Figure 10: Graph of photo recognition certainty based on number of photos supplied

The recognition algorithm was implemented as follows: first, the image is sent to Betaface, which will respond with a list of 10 faces that have been previously tagged with a namespace. These images will also have an individual confidence for that particular instance of the face. The faces

with the top 3 confidences are averaged for each person to form a final confidence value, which can then be used to see which person the face matches.