sddec20-19

Automated Team Attendance Tool

Final Presentation Fall 2020

Angela Schauer, Brandon Johnson, Nathan Oran,
Lance Demers, Connor Sullivan

Advisor/Client: Mohamed Selim

Our Team

Connor Sullivan
Software Engineering

Angela Schauer
Software Engineering

Brandon Johnson
Electrical Engineering

Lance Demers
Computer Engineering

Nathan Oran
Computer Engineering

Team Mail List: sddec20-19@iastate.edu

Project Plan

A high-level overview

What is Team-Based Learning (TBL)?

Instead of the traditional lecture, students collaborate and learn from each other in addition to the instructor.

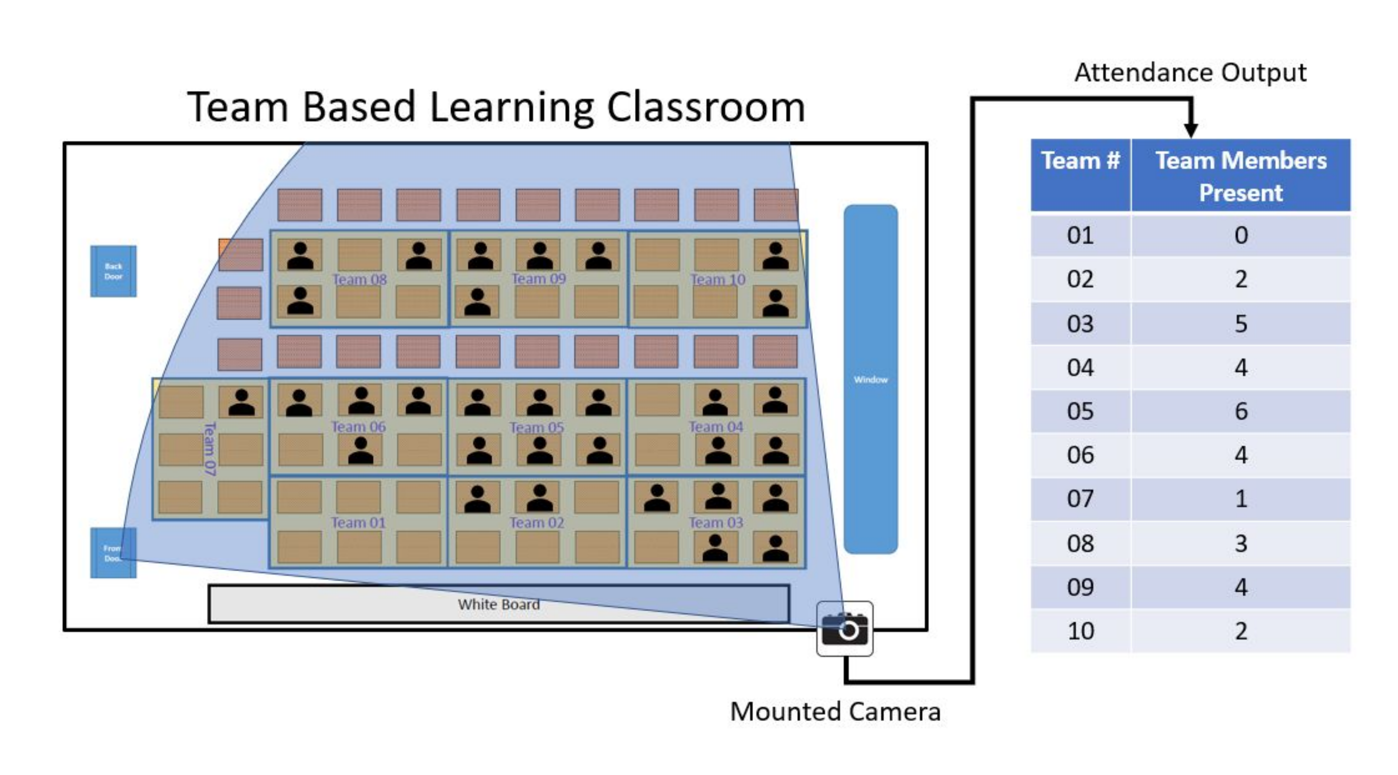
Our project is to automate the attendance procedure of Team-Based Learning classrooms to eliminate the time taking attendance during class.

Our solution uses a Raspberry Pi, a Raspberry Pi Camera, and an artificially intelligent computer vision library (Yolov3).

The Problem Statement

- For TBL classes to succeed it is invaluable that all team members show up.
- Taking attendance can take a good portion of class time
- Even with using tools to help speed it up it takes about 2-3 minutes
- Over a whole semester would add up to about one whole class period

A Conceptual Sketch



Team #	Team Members Present
01	0
02	2
03	5
04	4
05	6
06	4
07	1
08	3
09	4
10	2

Functional Requirements

- **Hardware Requirements:**
  - Run on Raspberry Pi
  - Use a camera to take clear pictures
  - Mounted such that the camera has full visibility of the classroom
- **Software Requirements:**
  - Control and receive pictures from the camera
  - Accurately detect the students within the taken picture (required 0% error)
  - Determine which teams are missing members
  - Consolidate the attendance and send the report to the professor
  - Allow the professor the create and adjust multiple seating charts for use in the system

Technical and Non-Technical Constraints

- **Technical Constraints**
  - We were limited by the density which the detection software could work, so we developed a method to break the image into smaller sections to be able to accurately detect the objects we desired.
  - With our design we are limited on the placement of the camera since it requires being plugged into an outlet for power.
- **Non-Technical Constraints**
  - COVID limited our ability to gather proper testing data, and properly develop and test our project.
- **Non-Functional Requirements**
  - The tool should be faster than previous attendance methods.
  - The tool should be just as accurate as previous attendance methods.

Potential Risks and Mitigation

Risk	Mitigation
Students do not want their picture to be archived or stored	Photos will be deleted from the server once they are no longer needed YOLO does not need to be trained to detect an individual’s face for the success of our project
An image that is taken does not meet the requirements of being a quality photo	Multiple pictures will be taken throughout the class to ensure accuracy

Resource and Cost Estimates

- **Necessary Components:**

Raspberry Pi Model 4	- $35
Pi Camera V2	- $30
MicroSD Card	- $5
Micro USB Charger	- $5
Mount	- $20

- Total Cost: $95

Project Milestones

Software

- Student Detection
  - Detect Using Yolov3
- Seating Chart Submission
  - Interface for Professors to enter team-based seating charts and schedule attendance capture time
- Attendance Mapping
  - Determining Attendances and Absences

Hardware

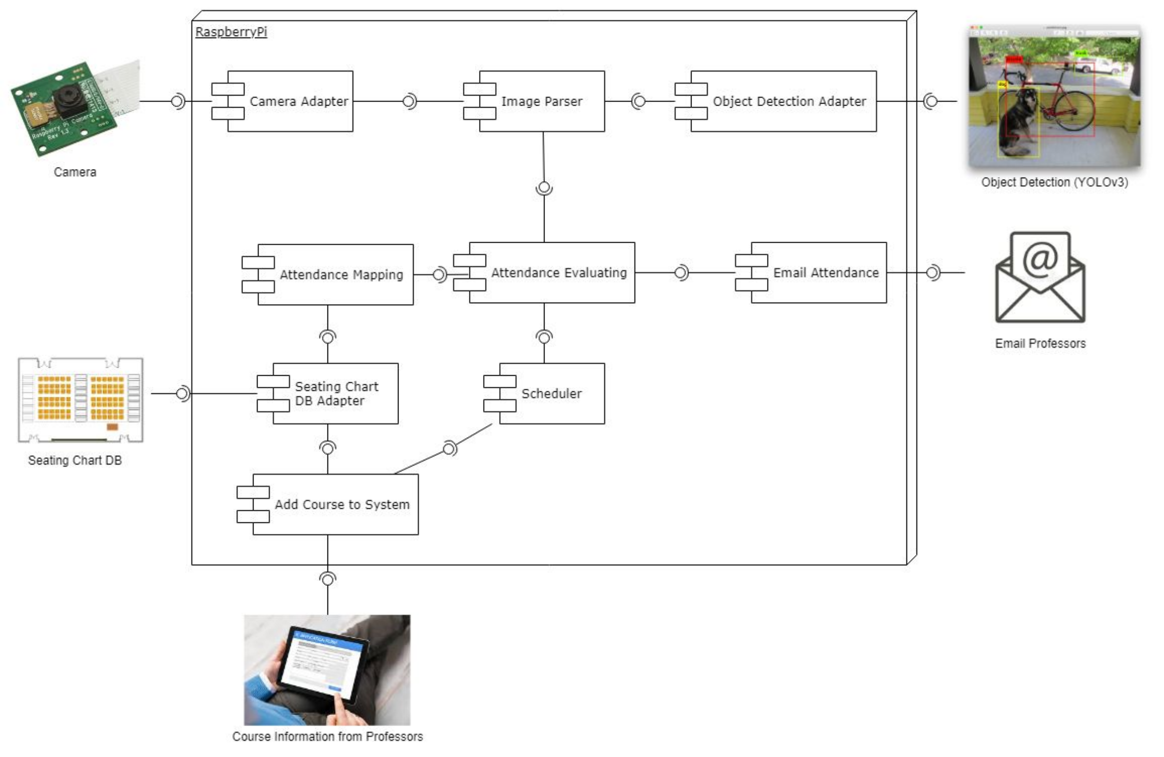
- Raspberry Pi Configuration
  - Network Setup and Dependencies
- Attendance Scheduler
  - Capture the image of the classroom based on the submitted capture time
- System Mount
  - How the system will sit in the classroom
- Email Server
  - Using mutt to send the report to professors

System Design

Functional Composition

- **Attendance Evaluator**
  - Responsible for processing images into attendance reports
- **Scheduler**
  - Responsible for calling the Attendance Evaluator to run at the proper class times, and sending the reports to the professor
- **YOLO v3**
  - Object Detection tool of choice
- **Camera**
  - Raspberry Pi Camera Module V2
- **Seating Chart DB**
  - Stores class seating chart information for each class

Component Diagram



Mounted System Technologies Used

Pi Environment

- Raspbian Buster
- Python
- OpenCV
- YOLOv3
- Linux CLI
- Bash Shell

Backend Technology Platforms

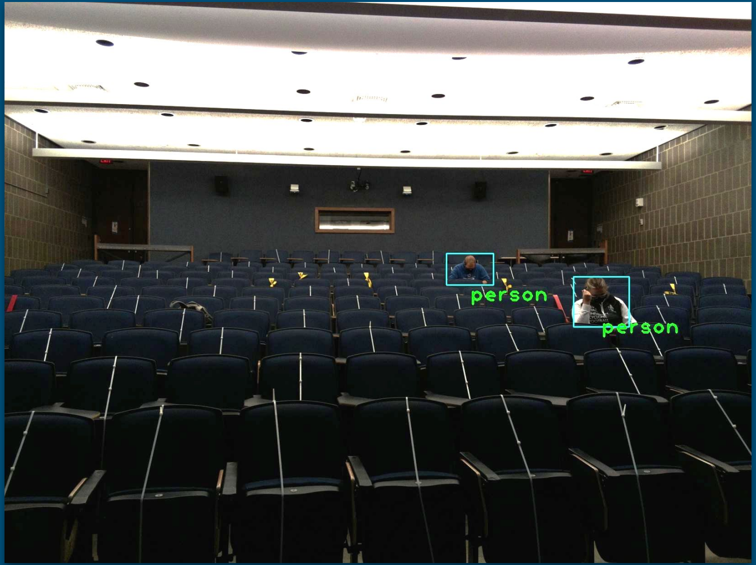
Backend

- MySQL Database
    - Table relationships
    - Look-up tables
    - Data storage tables
- ASP.NET Web Application and API
    - C#
    - Entity Framework
    - REST
    - Dependency Injection

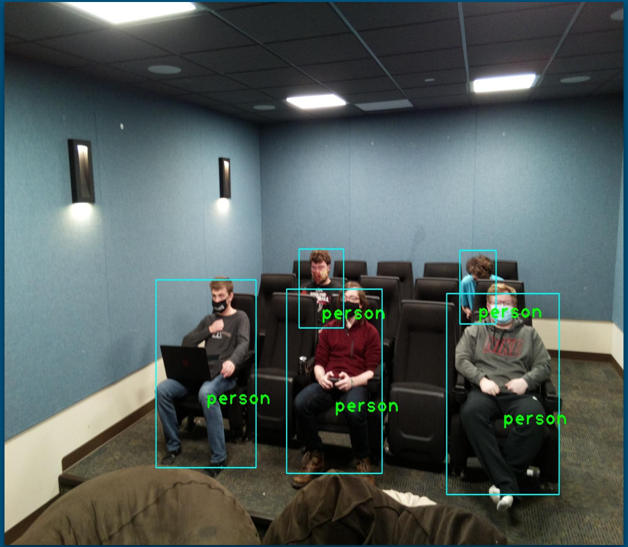
Test Plan

- System
  - Procure images of the actual classrooms the program would be used in with varying numbers of students in the seats
  - Focus on proof of concept with small groups then ramp up difficulty
  - Would report accuracy as 100% up to X amount of students/a room Y big
- Backend
  - A suite of http requests were created in Postman to test all of our API endpoints to confirm the relationship between the API and Database

Prototype Implementation - YOLOv3



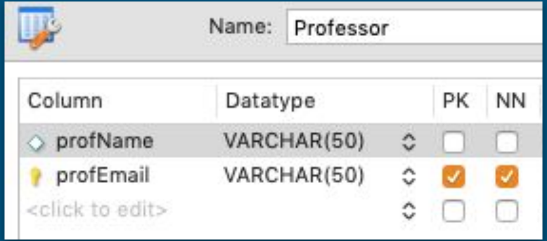
Carver 101 Sample Test Result



Off Campus Sample Test Result

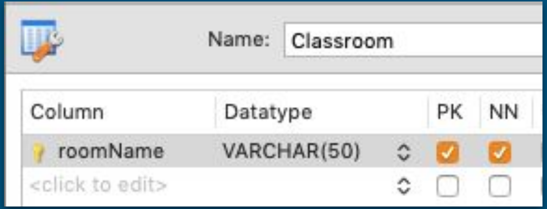
Prototype Implementation - Database

- Database Tables Breakdown
  - Professor
  - Classroom
  - Class
  - SeatingChart
  - Times



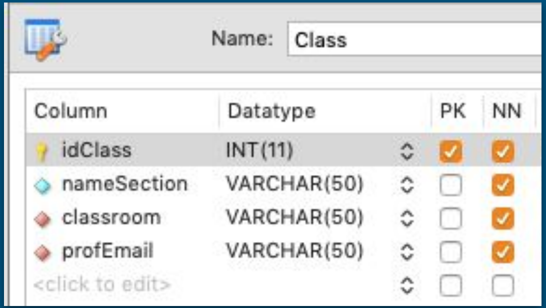
Name: Professor

Column	Datatype	PK	NN
profName	VARCHAR(50)	□	□
profEmail	VARCHAR(50)	☑	☑
<click to edit>		□	□



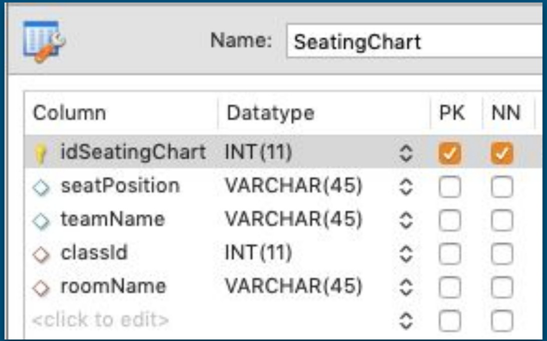
Name: Classroom

Column	Datatype	PK	NN
roomName	VARCHAR(50)	☑	☑
<click to edit>		□	□



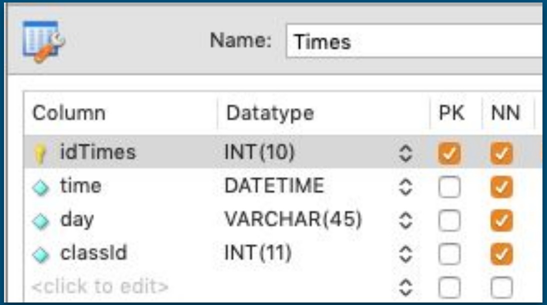
Name: Class

Column	Datatype	PK	NN
idClass	INT(11)	☑	☑
nameSection	VARCHAR(50)	□	☑
classroom	VARCHAR(50)	□	☑
profEmail	VARCHAR(50)	□	☑
<click to edit>		□	□



Name: SeatingChart

Column	Datatype	PK	NN
idSeatingChart	INT(11)	☑	☑
seatPosition	VARCHAR(45)	□	□
teamName	VARCHAR(45)	□	□
classId	INT(11)	□	□
roomName	VARCHAR(45)	□	□
<click to edit>		□	□

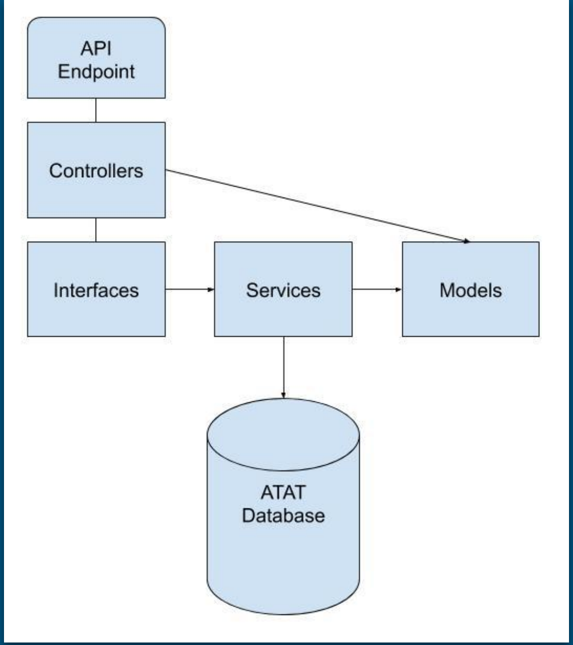


Name: Times

Column	Datatype	PK	NN
idTimes	INT(10)	☑	☑
time	DATETIME	□	☑
day	VARCHAR(45)	□	☑
classId	INT(11)	□	☑
<click to edit>		□	□

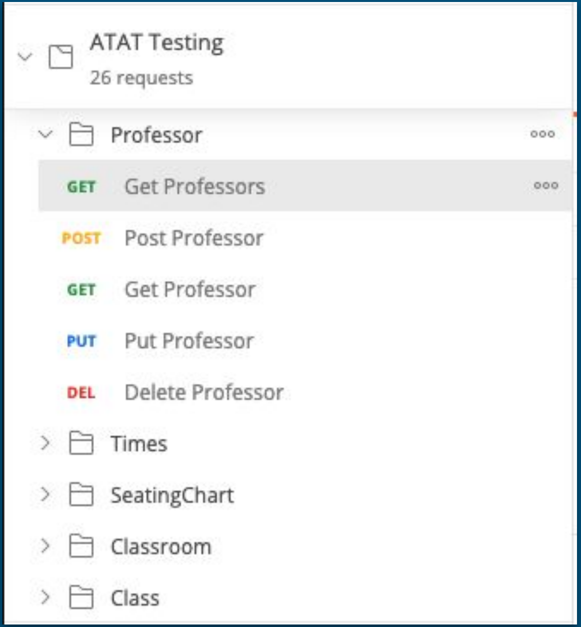
Prototype Implementation - API

- **API Implementation**
  - DBContext
  - Models
  - Controllers
  - Interfaces
  - Services
- **Postman Testing**



API Architecture Diagram

Postman http request set



ATAT Testing
26 requests

Professor
- GET Get Professors
- POST Post Professor
- GET Get Professor
- PUT Put Professor
- DEL Delete Professor

Times

SeatingChart

Classroom

Class

System Design

Nathan

Engineering Standards and Design Practices

- Engineering Standards
  - IEEE 802.11
- General industry standards for code
  - Comments
  - Consistent formatting
  - Git Version Control
    - Branching
    - Code Reviews

Lance

Conclusion

Contributions of each member

Connor: Raspberry Pi, YOLOv3 — 58 Hours

Brandon: Raspberry Pi, YOLOv3 — 51 Hours

Angela: Database, API — 67 Hours

Lance: Database, API — 56 Hours

Nathan: Database, API, Reports — 68.5 Hours

All: Initial research and design, testing, presentations

Future Prospect of the project

- Create the seating charts for each classroom with static plot points
- GUI to enter class information
- Algorithm to map data from object detection software to given seating chart
- Scheduler for taking pictures
- Along with TBL classroom attendance AT-AT also has potential application in contact tracing during the pandemic

Questions and Answers