Samaritan Security

SDMAY20-45
Team Members: Kate Brune, Ryan Goluch, Ann Gould, TJ Paris, Saba Shaarbaf-Toosi
Faculty Advisor: Dr. Goce Trajcevski

Introduction/Motivation

Main Aspects of Security

- Access Control
  - Physical Components
    - Key Cards
    - Locks
- Surveillance
  - Intrusion detection sensors
  - Security Cameras

Problem:

Market research shows that surveillance systems typically do not consist of more than passively streaming camera feeds. The logical response to utilize full camera functionality is to automate the analysis of video feeds.

Solution:

Samaritan overlays a pre existing surveillance system:

- Automates process of monitoring
- Uses facial recognition to alert the company, in real time, when anyone other than a registered employee is on the premises.

Goal: To make the daunting task of securing your company easier.

Design Requirements

Functional Requirements

- The user shall be able to view live camera feeds
- The system can detect if a person is unauthorized
- The user is alerted when an unauthorized person is on the premises
- Each organization's data is isolated from other organizations

Nonfunctional Requirements

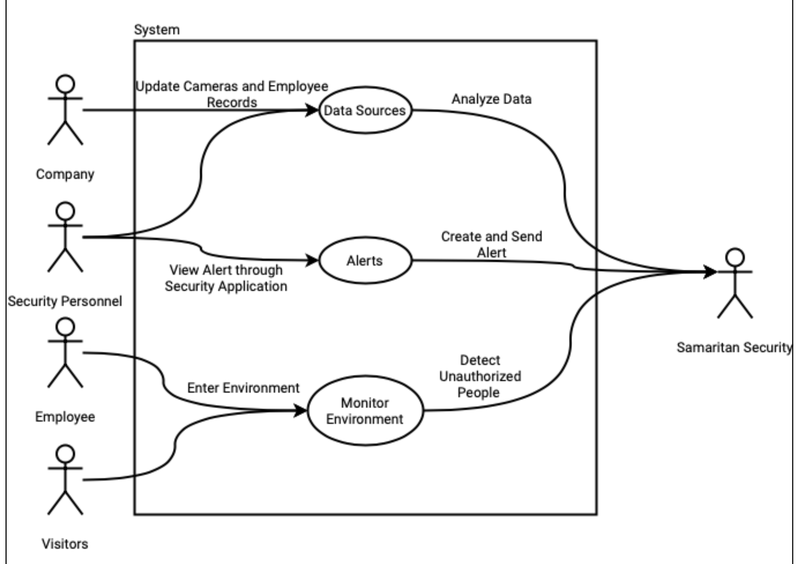
- Users are alerted of unauthorized access in real time
- The system has a constant up time

Operating Environment

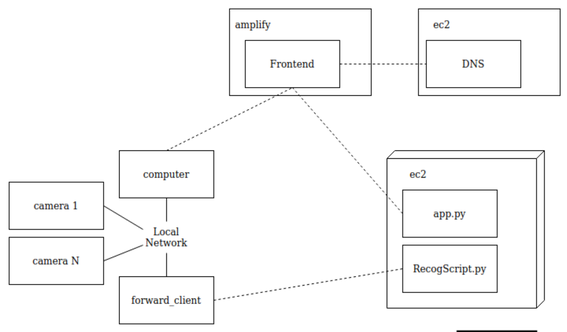
- Online web application
- Hosted on AWS EC2
- Utilizes IP connected cameras

Design Approach

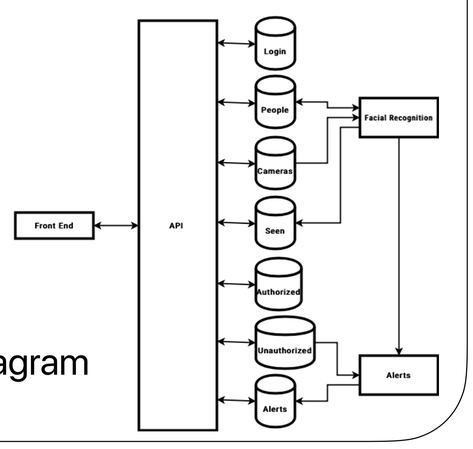
Conceptual Sketch



Block Diagrams



System Block Diagram



Backend Block Diagram

Functional Modules

Facial Recognition Module: Gets all of the cameras/information about people that are connected to our system. Locates and identifies users in feeds.

API: Bridges backend functionality to the front end.

Alerts: Notifies a user when an unauthorized person has been detected.

DNS: "Ties" an org's name to the org's IP

Forward Client: Forwards local camera feeds to Samaritan's server

Frontend Client: Creates initial system set up, sets up data source configurations, monitors the system, allows viewing threats. Connects with API.

Technical Details

Technologies Used

Platforms Used

Engineering Standards and Design Practices

P14764 - Software Engineering - Software Life Cycle Processes - Maintenance

- Focuses on an iterative style of development
- Built with maintainability through actions like GitHub Workflows

15289-2015 - ISO/IEC/IEEE International Standard Systems and software engineering

- Iteratively created and defined over lifecycle
- Uses defined specifications to find what needs to be fixed in code

Testing

Testing Environment

- Python Unittest
- React Testing Library + Jest
- Github Workflows

Testing Strategy

- User/Acceptance Tests
- Tests written to insure individual functionality
- Workflows used to test dependencies and validate syntax of committed code