Passenger and Driver Safety

## Created by:

## Miguel Millan, Justin Isrow, Colin Cambell, Fabian Lefevre

# Project vision

## Background

Passenger and driver safety continue to be a challenging problem in transit systems. With increased threat of terrorism, probability of such occurrences is rapidly growing. While there are many in bus camera systems available for monitoring the passenger activities in the bus, due to large cost of associated cellular charges, such solutions are cost prohibitive. As an alternate, bus systems may implement multi camera stream recording for post event evaluation. Using these camera streams to train a machine learning algorithm, it is possible to develop an intelligent event detection system capable of effectively detecting and warning about events in real time.

## Socio-economical Impact, Business Objectives, and Gap Analysis

## Security and Ethical concerns

## Glossary of Key Terms

# Project Execution and Planning

## Term Information

Our team will build an intelligent event detection system. This system will consist of cameras deployed in a bus which monitor passenger activity, and a web application which passes the camera feed to a machine learning algorithm to classify potentially hazardous events and notify monitoring personnel.

## Tools and Technology

## Project Plan

Last Updated: 1/13/19

Sprint 1: 1/7/2019 - 1/16/2019

* Project Plan
* Requirements
* Tool Chain Setup
* Project Scope Capture

Sprint 2: 1/16/2019 - 1/23/2019

* Basic Diagrams
* User Stories / Detailed Use Cases
* Cloud hosted server setup and configuration
* Camera system research / design

Sprint 3: 1/23/2019 - 2/11/2019

* Web UI skeleton and basic functionality
* Camera system deployment & initial data collection
* Device Registration implemented

Sprint 4: 2/11/2019- 2/25/2019

* Label / identify events in data for training
* Start manual training of the algorithm
* Add monitoring statistics to Web UI

Sprint 5: 2/25/2019 - 3/11/2019

* Continue collecting data and refining the algorithm
* Start testing the algorithm against a subset of labeled data
* Web UI Improvements

Sprint 6: 3/11/2019 - 3/25/2019

* Continue collecting data and refining the algorithm
* Web UI Improvements
* Start testing on unlabeled data and review post flagging

Sprint 7: 3/25/2019 - 4/9/2019

* Continued Testing and debugging
* Dry Run - full autonomous detection

Sprint 8: 4/9/2019 - 4/24/2019

* Continued Testing and debugging
* Final Competition

## Best Standards and Practices

To best manage this project, we decided to use the Scrum method and break up the work into two-week long Sprints so that it could be planned in more manageable fashion.

The team member roles and responsibilities are as follows:

* Miguel Millan – Project Lead / Documentation / Web Application Developer
* Justin Isrow – Data Collection / Algorithm Training
* Colin Campbell – Algorithm Training
* Fabian Lefevre – Data Collection / Documentation

We decided for the whole team to meet at least once a week on Friday to discuss progress for each team member and work through team goals. In addition, we planned on meeting immediately after end of sprint presentations to ensure each member knows what their individual responsibility is for the next sprint.

# System Requirement Analysis

## Functional Requirements

## Non-functional Requirements

## On-Screen Appearance of landing and other pages requirements

## Wireframe designs

# Functional Requirements Specification

## Stakeholders

## Actors and Goals

## User stories, scenarios and Use Cases

## System Sequence / Activity Diagrams

# User Interface Specifications

## Preliminary Design

## User Effort Estimation

# Static Design

## Class Model

## System Operation Contracts

## Mathematical Model

## Entity Relation

# Dynamic Design

## Sequence Diagrams

## Interface Specifications

## State Diagrams

# System Architecture and System Design

## Subsystems / Component / Design Pattern Identification

## Mapping Subsystems to Hardware (Deployment Diagram)

## Persistent Data Storage

## Network Protocol

## Global Control Flow

## Hardware Requirement

# Algorithms and Data Structures

## Algorithms

## Data Structures

# User Interface Design and Implementation

## User Interface Design

## User Interface Implementation

# Testing

## Unit Test Architecture and Strategy/Framework

## Unit Test definition, test data selection

## System Test Specification

## Test Reports per Sprint

# References