
Software Design Documentation

VehID

Version 1.0

Remington Greko, Spencer Hirsch, Thomas Johnson,
Alexis Nagle

Project Advisor: Dr. Marius Silaghi

Project Client: Clayton Levins, Executive Director of
Smart North Florida

October 2, 2023

Contents

- 1 Introduction** **3**
 - 1.1 Purpose 3
 - 1.2 Scope 3
 - 1.3 Overview 3
 - 1.4 Definitions and Acronyms 3

- 2 System Overview** **4**

- 3 System Architecture** **5**
 - 3.1 Architectural Design 5
 - 3.2 Decomposition Description 6
 - 3.3 Design Rationale 6

- 4 Data Design** **7**
 - 4.1 Data Description 7
 - 4.2 Data Dictionary 7

- 5 Component Design** **8**

- 6 Human Interface Design** **10**
 - 6.1 Overview of User Interface 10
 - 6.2 Screen Images 10
 - 6.3 Screen Objects and Actions 11

1 Introduction

1.1 Purpose

This software design document describes the architecture and system design of VehID, a vehicle identification machine learning model. This system is intended to be used for law enforcement purposes, such as, AMBER Alerts and other vehicle related crimes. This software will be useful for both local and federal governments as it will allow them to receive real-time information about new leads in crimes.

1.2 Scope

Our software, VehID, is a machine learning model that will utilize three separate Convolutional Neural Networks (CNNs) (1.3.1) to accurately identify vehicles based on set criteria. Each CNNs (1.3.1) is designated to identify a specific criteria, the first being color of vehicle, the second being the make and model of the vehicle, and third being, the the license plate of the vehicle. In the event that a match is found location of the the vehicle is reported to the proper authorities. This system would automate the entire process of locating vehicles found to be involved in crimes.

1.3 Overview

This document is intended to layout our project and our intended execution of the construction of our software. The following sections will allow a reader to better understand the way in which our software will be laid out and process by which we constructed the software. It is important for a reader to understand the ways in which the various parts of our product work with one another in the event a portion is changed, the primary functionality needs to be maintained. This document will give detailed overview of our product from a more technical software development perspective.

1.4 Definitions and Acronyms

Reference No.	Abbreviation	Definition	Source Reference
1.3.1	CNN	Convolution Neural Network: A type of feed-forward neural network commonly used for image recognition and processing.	Wikipedia
1.3.2	AMBER Alert	America's Missing, Broadcast Emergency Response; Public system for distribution of information about missing/kidnapped children	Wikipedia
1.3.3	DOT	United States Department of Transportation	

2 System Overview

As previously stated, our product will utilize three CNNs (1.3.1) all working together in order to identify a vehicle in real-time based on a network of traffic cameras, the goal would be to integrate it with the Department of Transportation (DOT) (1.3.3) network. Our first CNN (1.3.1), vehicle color recognition, will search all cars in a frame to see if the color is present in any of the vehicles. If the color is present, then we will use our second CNN (1.3.1) to identify if the vehicle fits the make and model of a target vehicle. If both of these criteria are met we will then use our third CNN (1.3.1) to identify if there is a license plate in frame. If a license plate is available and our database contains either a full or partial match, the system will check to determine whether or not the vehicle is a match. If the license plate is not a match, our system will still flag the vehicle, as the operator of the vehicle could have changed the license plate. However, our system will flag this match different compared to a full or partial license plate match. This alert will then be returned to our user through our user interface.

The machine learning portion is the main portion of our product, however, we do have supporting components. One of our supporting components is the database that we will be using to allow for our model to constantly surveil. There are two ways that our database is updated, one being automatic, through web scraping, and the second being through user input. There are online databases that publish the information of vehicles involved in crimes. For example, the AMBER Alert page shows ongoing AMBER Alerts, we can pull this data and add it to our database for our model to now search for vehicles matching the criteria. The same can be done for stolen vehicles, there is an online database containing all of the information about stolen vehicles, we can add this data to our database for our model to now surveil for vehicles matching these descriptions.

All information can be viewed through our User Interface, which will allow for users to view our database and view recent updates. Updates will vary based on priority and be dispatched to the necessary local agencies to aid in the continuation of the investigation.

3 System Architecture

3.1 Architectural Design

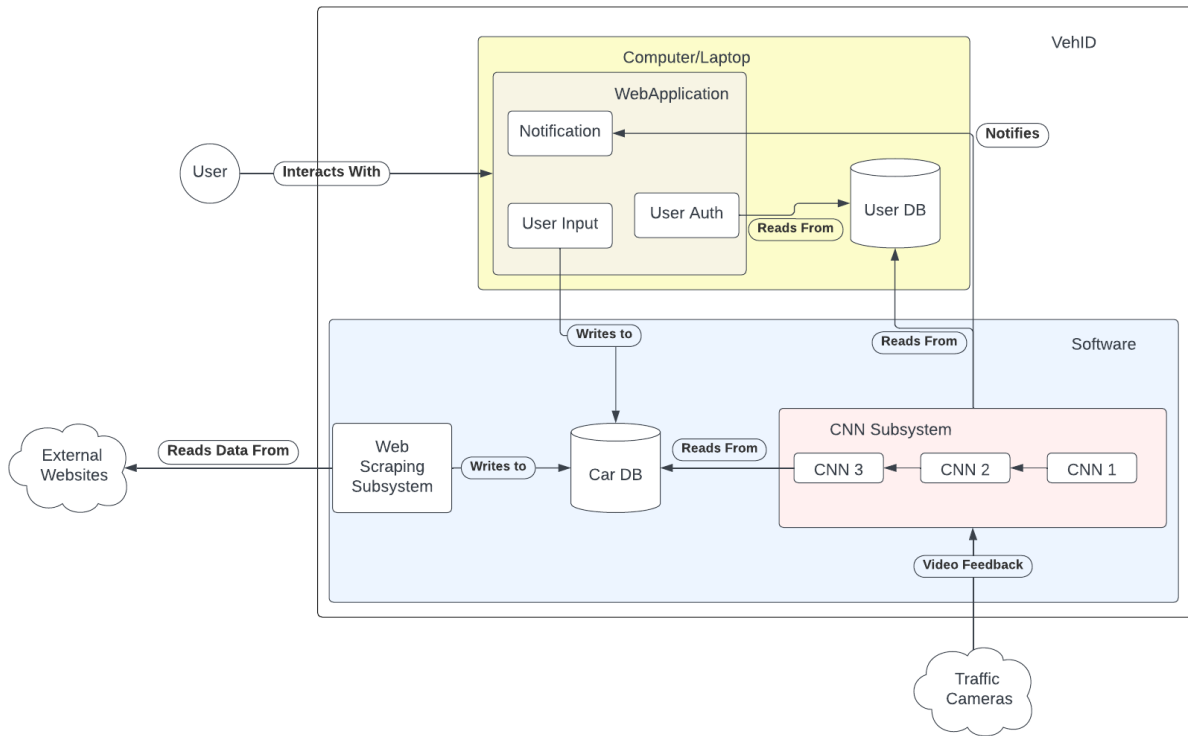


Figure 1: System Architecture

3.2 Decomposition Description

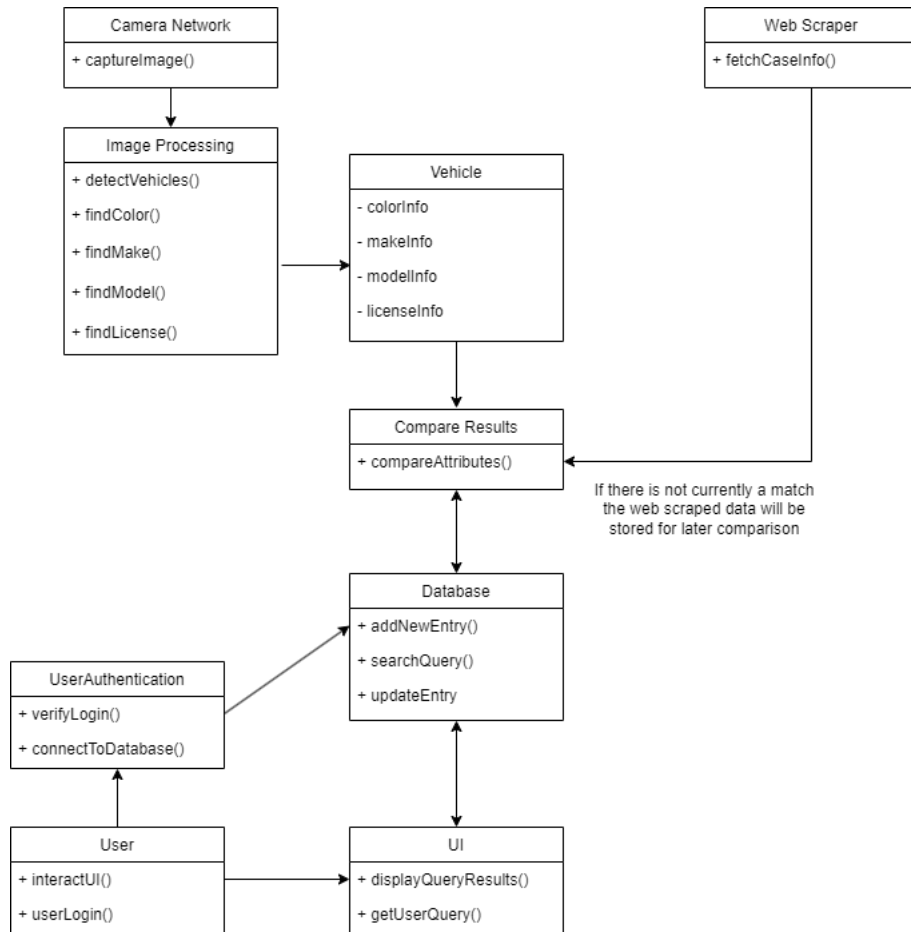


Figure 2: Data Flow Diagram

3.3 Design Rationale

Given the architecture in figure 1, the main point of consideration was the vulnerability of the DB having two points of concurrent input. Considering some input will come from users, sanitation of data will be very important. Ultimately we didn't see a way to avoid user's inputting data as from a functionality standpoint as its important users can contribute to any lost vehicles.

Handling data from external resources is also important, which is why we have a dedicated subsystem for web scraping information, and verifying the data. The CNN (1.3.1) subsystem will also verify data before beginning the process of identifying vehicle attributes.

Seeing as how co-dependent the subsystems are on other subsystems we wanted to provide liberty to our subsystems by having them be a part of the same larger system. This should make the transportation of data and cohesive.

4 Data Design

4.1 Data Description

Our project is going to utilize CNNs (1.3.1) as the primary data structures within the system. However the use of the CNNs (1.3.1) is purely for processing and will not store any data other than the parameters and attributes of the model once it is trained. To aid the CNNs (1.3.1), our system will have a database that stores the search query along with any information that the system retrieves about each search query. This will include descriptive attributes of the specified vehicle that will aid in the search such as color, make, model, license plate, and year. The database will then also store all information for possible matches to the search query such as location, direction, and additional information. Lastly, the database will hold information corresponding to the search query such as an associated name, address, or social security number in order to properly report findings of each case.

4.2 Data Dictionary

- Camera Network
 - Type: Hardware
 - Description: Captures images for the system to process
 - Functions: `capture_image()`
- Compare Results
 - Type: Class
 - Description: Compares captured image data to web scraped data
 - Functions: `compare_attributes()`
- Database
 - Type: Class
 - Description: Stores and handles retrieval/modification of vehicle data
 - Functions: `add_new_entry()`, `search_query()`, `update_entry()`
- ImageProcessing
 - Type: Class
 - Description: Processes images using OpenCV and CNNs (1.3.1) to extract vehicle information
 - Functions: `detect_vehicles()`, `find_color()`, `find_make()`, `find_model()`, `find_license()`
- UI
 - Type: Class

- Description: Provides a user interface for the user to interact with the system
- Functions: `display_query_results()`, `get_user_query()`
- User
 - Type: Class
 - Description: Represents the user
 - Functions: `user_login()`, `interact_UI()`
- User Authentication
 - Type: Class
 - Description: Manages user login and authentication
 - Functions: `verify_login()`, `connect_to_database()`
- Vehicle
 - Type: Class
 - Description: Represents a vehicle and its attributes
 - Functions: `get_color()`, `get_make()`, `get_model()`, `get_license()`
- Web Scrape
 - Type: Class
 - Description: Gathers information to search for target vehicles
 - Functions: `fetch_case_info()`

5 Component Design

- Camera Network
 - `capture_image()`: This function will take video footage and extract individual frames.
- Compare Results
 - `compare_attributes()`: This class will take the attribute that become attached to each vehicle image and compare it to the attributes of vehicles within the database to determine if there are any matches.
- Database
 - `add_new_entry()`: This function will allow users to add new entries to the database manually.

- search_query(): This function will allow users to search the database for a specific query entry.
- update_entry(): This function will allow users to manually update fields for any specific entry.
- ImageProcessing
 - detect_vehicles(): This function will take the individual frames extracted from the captureImage() function and identify each vehicle in the image with a bounding box.
 - find_color(): This function will utilize an already trained CNN (1.3.1) to predict the color of a vehicle.
 - find_make(): This function will utilize an already trained CNN (1.3.1) to predict the make of a vehicle.
 - find_model(): This function will utilize an already trained CNN (1.3.1) to predict the model of a vehicle.
 - find_license(): This function will utilize an already trained CNN (1.3.1) to predict the license plate of a vehicle.
- UI
 - display_query_results(): This function will display the query results of a specified search.
 - get_user_query(): This function will allow the user to specify a certain search.
- User
 - user_login(): This function will be a simple user login taking in a username and password.
 - interact_UI(): This function will handle the user interactions within the interface.
- User Authentication
 - verify_login(): This function will be used to authenticate users to give access to different features of the interface as they apply to the users account type.
 - connect_to_database(): This function will connect the user to the database and give access to edit the database if specified within the user's account type.
- Vehicle
 - get_color(): This function will return the color attribute of the given vehicle.
 - get_make(): This function will return the make attribute of the given vehicle.
 - get_model(): This function will return the model attribute of the given vehicle.
 - get_license(): This function will return the license attribute of the given vehicle.

6 Human Interface Design

6.1 Overview of User Interface

Our user interface will be used in order to support our database, a user does not need to access the machine learning model but may need to access the attributes of a database. Due to this, our system will have two different user functionalities, a database administrator and a regular user. Some of the permissions will be shared across both user types. The shared functionalities will allow for a user to view recent alerts, and the information associated with the database, such as locations, vehicle information, associated people, police reports, and other related attributes. Not only do we want a user to be able to view database contents, we also want our users to be able to add to the database. This is the functionality that primarily defines what an administrator is and what a regular user is. A regular user will be able to propose changes to the database, while an administrator will be able to accept proposed changes or add new information to the database.

6.2 Screen Images

These are the two views of the main interfaces the users will be interacting with.

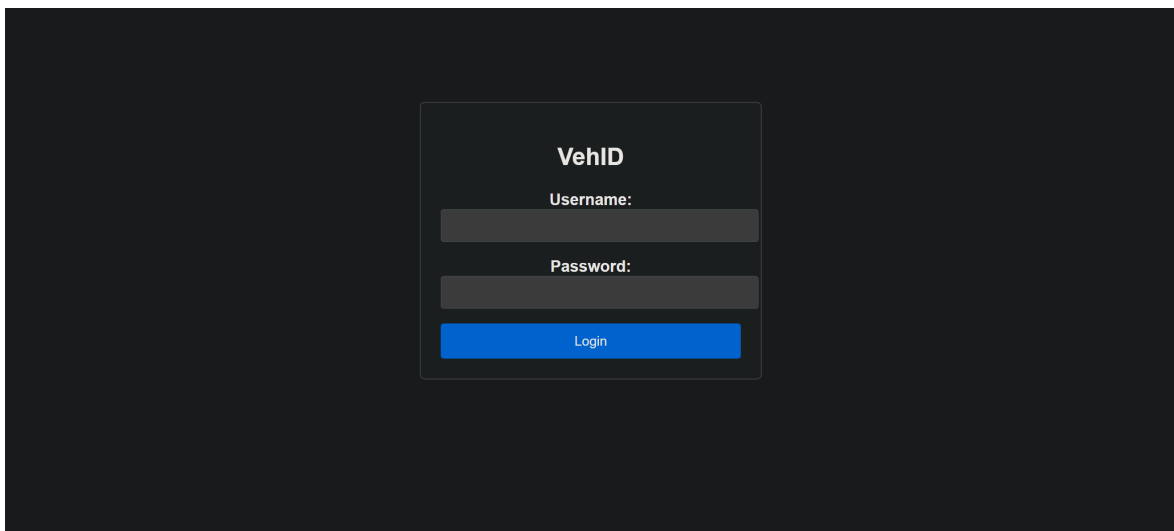


Figure 3: Login Screen

VehID Web App

Search for data...




Color	Year	Make	Model	License Plate	Case Info	Description	Location	Image
Red	2020	Toyota	Camry	ABC-123	Case 12345	Red Toyota Camry parked illegally.	123 Main St, City	
Blue	2019	Honda	Civic	XYZ-789	Case 67890	Blue Honda Civic blocking driveway.	456 Elm St, Town	
Silver	2022	Ford	Escape	DEF-456	Case 54321	Silver Ford Escape parked in no-parking zone.	789 Oak St, Village	

Figure 4: Database Web View

6.3 Screen Objects and Actions

The user interface will have two main components: the login and the database. The login will include a simple username and password entry that will be authenticated to give the user the access level specified within the account. The database will look different for different user types. Traditional users will simply be able to view the database in a table format and search for a specific case based on any criteria they wish. The administrator users will have the same access as the tradition users, but in addition they will have two extra selections: add entry and edit entry. These extra selections will allow the user to manually edit the database. The add entry will allow the user to manually enter an entire query into the database. The edit entry will allow the user to search for then modify an existing entry.

Works Cited

“Convolutional Neural Network.” Wikipedia, Wikimedia Foundation, 3 Sept. 2023, en.wikipedia.org/wiki/Convolutional_neural_network.

“Amber Alert.” Wikipedia, Wikimedia Foundation, 12 Sept. 2023, en.wikipedia.org/wiki/Amber_alert.