System Design Document

Traffic Citation and Reporting System

Prepared by Lefan Wang, Alexzander Mazzuca, Peter Kastias, Aquan Morgan

COSC/ITEC 3506N Software Engineering

Algoma University

March 16, 2022

Purpose

The purpose of the SDD is to document and track necessary information that describes the system at the architecture level, including subsystems and their services, hardware mapping, data management, access control, global software control structure, and boundary conditions. The SDD serves as the documentation that contains all architectural and design details about the system.

Audience

The Intended audience for this document include:

1. TCRS Project Team
2. Traffic System Engineers
3. Application Development Team
4. Researchers
5. Data Scientist

As mentioned in the system requirement document the application will be developed using java therefore readers would find it beneficial to have knowledge of the java language.

1.0 Introduction

1.1 Purpose of the System

TCRS is a new application designed to help keep communities safe by using an automated system which detects cars traveling above the speed limit. It is designed to work in tandem with the TCRS application where users can easily pay tickets, officers can easily issue tickets and administrators can make adjustments to the system.

1.2 Design Goals

The goal of this project is to design, build and test an efficient high-performance traffic citation system that is easy to use, flexible, reliable, and easy to learn while also providing a solution to the current problem surrounding proper documentation of citations and infractions.

Design goals of the system are to:

- Increase accessibility
- Support timely processing of tickets
- Flexibility
- Reusability
- Security

- Ability to create user profiles
- Ability to pay for a ticket
- Ability to generate invoices
- Ability to issue tickets

1.3 Definitions, Acronyms, and Abbreviations

SDD - System Design Document
TCRS - Traffic Citation Reporting System
SQL - standard query language

1.4 References

Technische Universitaet Muenchen. (2006). *Ontologies in the software engineering process*. https://ase.in.tum.de/. Retrieved March 16, 2022, from https://ase.in.tum.de/lehrstuhl_1/files/teaching/ws0607/Wissensbasiertes%20SE/OntologiesInSE.pdf

NASA. (2016). *System design document (SDD) - NASA*. nasa.gov. Retrieved March 16, 2022, from https://ntrs.nasa.gov/api/citations/20160011412/downloads/20160011412.pdf

Schalk, G., Hinterberger, T., McFarland, D. J., & Mellinger, J. (2004, July). *Software design document - washington university in St. Louis*. Retrieved March 16, 2022, from https://classes.engineering.wustl.edu/ese497/images/9/96/2004Schalk_BCI2000Implementation.pdf

Atlanta Regional Commission. (2017, August 30). *Www.its.dot.gov*. Research Archives. Retrieved March 16, 2022, from https://www.its.dot.gov/research_archives/msaa/pdf/MSAA_SystemDesignFINAL.pdf

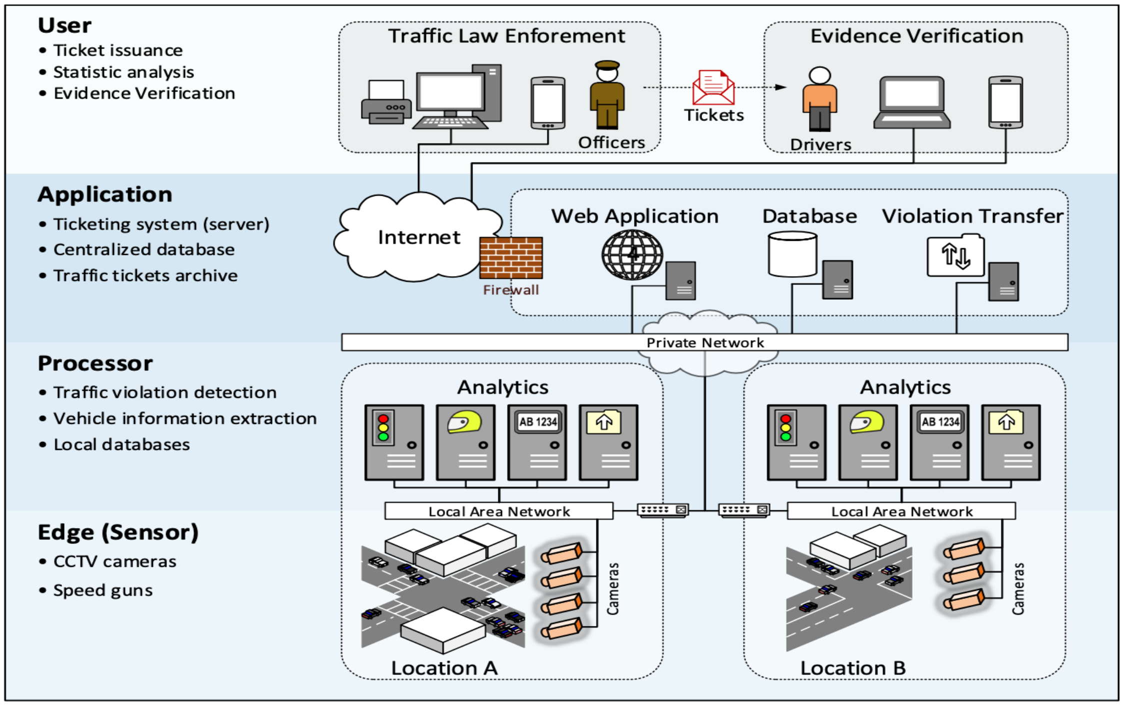
Limsoonthrakul, S., Dailey, M. N., Marikhu, R., Timtong, V., Chairat, A., Suphavilai, A., Seetamanotch, W., & Ekpanyapong, M. (2021, January 24). *Design and implementation of a highly scalable, low-cost distributed traffic violation enforcement system in Phuket, Thailand*. MDPI. Retrieved March 16, 2022, from https://www.mdpi.com/2071-1050/13/3/1210

1.5 Overview

TCRS is a web based application that will simplify the management of giving citations to traffic offenders. The traffic citation system allows officers to issue fines and their amount, provide the driver with law that was broken, location and time the infringement occurred while also providing the driver the most accurate and reliable information where they can quickly address their fines and amounts. The concept of a Traffic citation is not new however with our application we are able to assist the police and the people with regard to improving the settlement of traffic tickets. The system is aimed to reduce violations, increase safety, and improve driver habits.

2. Current Software Architecture

Currently traffic Citation systems exist where the system provides information concerning the amount of fines, articles violated, time and place of the trial however there are concerns surrounding accuracy and the amount of time it takes to deal with tickets

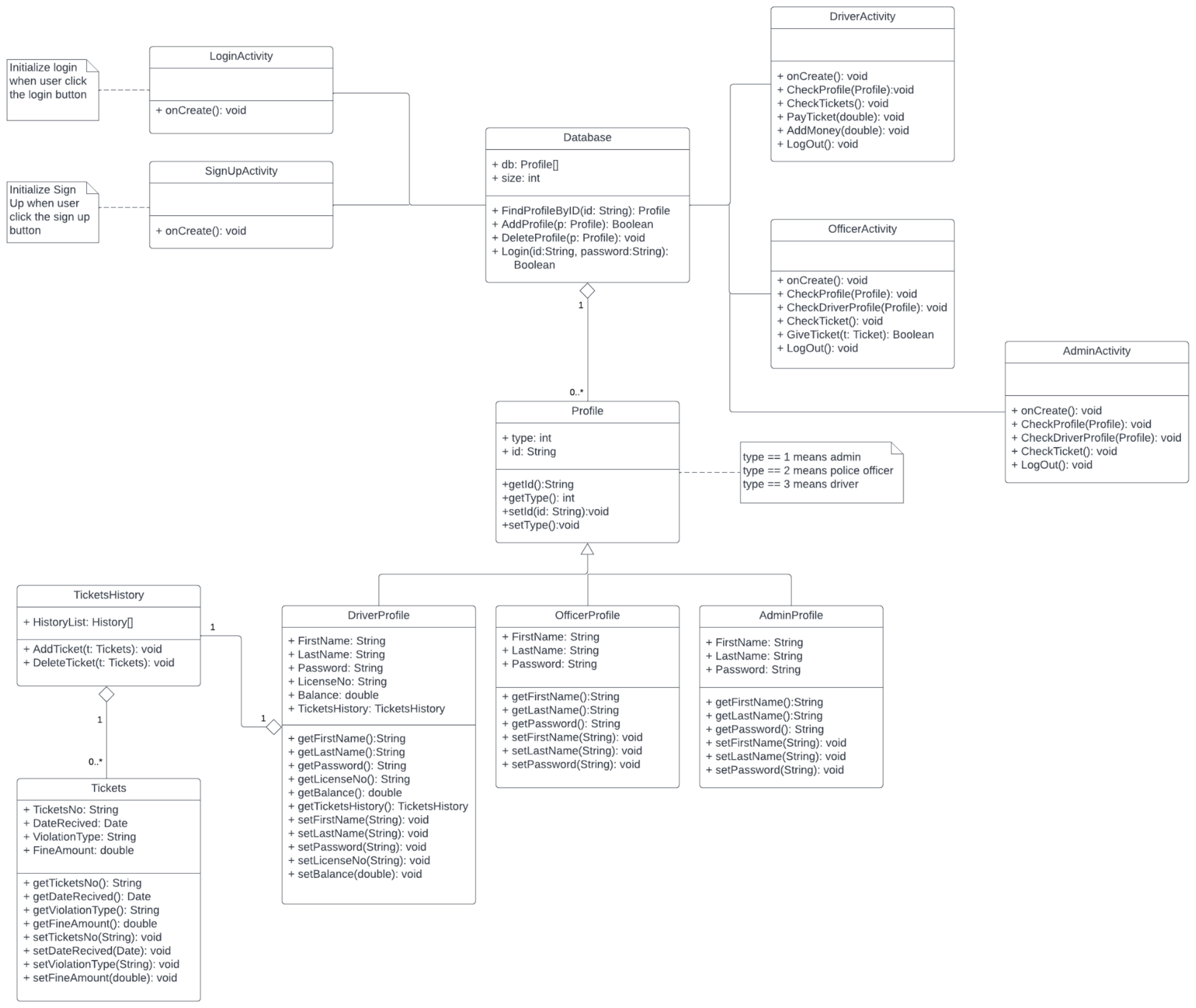


.

(Limsoonthrakul et al., 2021)

3. Proposed software architecture

3.1 Overview



UML Diagram

TCRS uses the broker pattern, our program defines the ‘Database’ as the broker. After the user logs in, different types of user will have a corresponding ‘type ID’(E.g. Officers will get type 2, drivers will get type 3), and gain unique access according to type ID to read, write or modify the profile. The database is not only the place to hold all the data, but also provides services like finding a particular profile by ID number, modifying profiles and checking if user ID matches the password at login stage.

3.2 Subsystem decomposition

Database subsystem
The Database subsystem is designed to hold and modify user profile data and tickets history data. Depending on the type of the user, it will give different authorities to allow users to add, modify and delete profile data or tickets data. Database subsystem is the key component of TCRS program and every action from the user end will go through it.

DriverActivity subsystem
The AriverActivity subsystem allows the user of driver type to check and pay tickets, view their profile information and add money to their account. Along with the Database subsystem, it forms the driver end of this program.

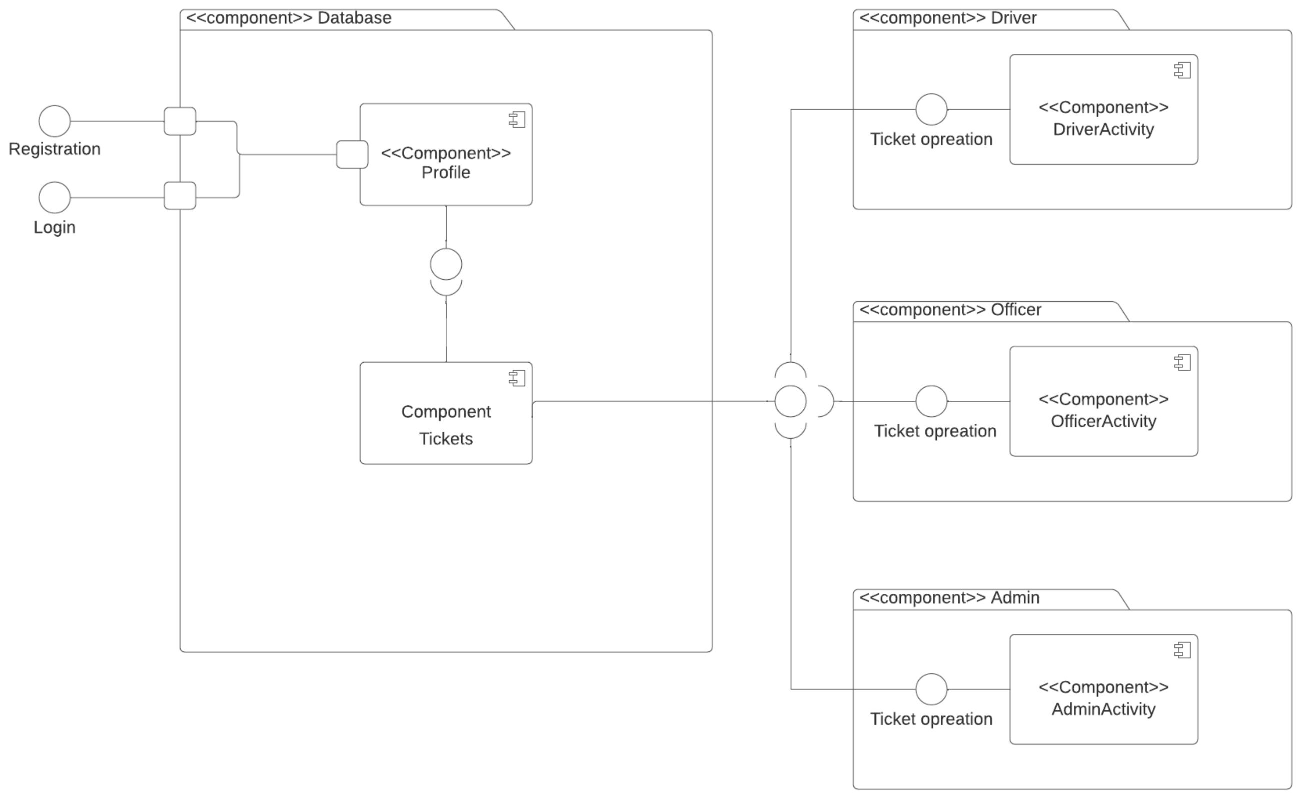
OfficerActivity subsystem
The OfficerActivity subsystem allows the user of officer type to check the drivers' profile, view their own profile information, check and give tickets. Along with the Database subsystem, it forms the officer end of this program.

AdminActivity subsystem
The AdminActivity subsystem allows the user of admin type to check the drivers' profile, view their own profile and check the drivers' tickets history. Along with the Database subsystem, it forms the admin end of this program.

After comparing the password and user ID, if it matches, the user will get a user type number which indicates the type of current user. According to that type number, the user will go to the corresponding subsystem.

3.3 Software/hardware mapping

TCRS is an Android application implemented in Android Studio using Java language. All the data will be stored locally.
Our program will have 4 components: Database, Driver, Officer and Admin. The component diagram will be shown below.

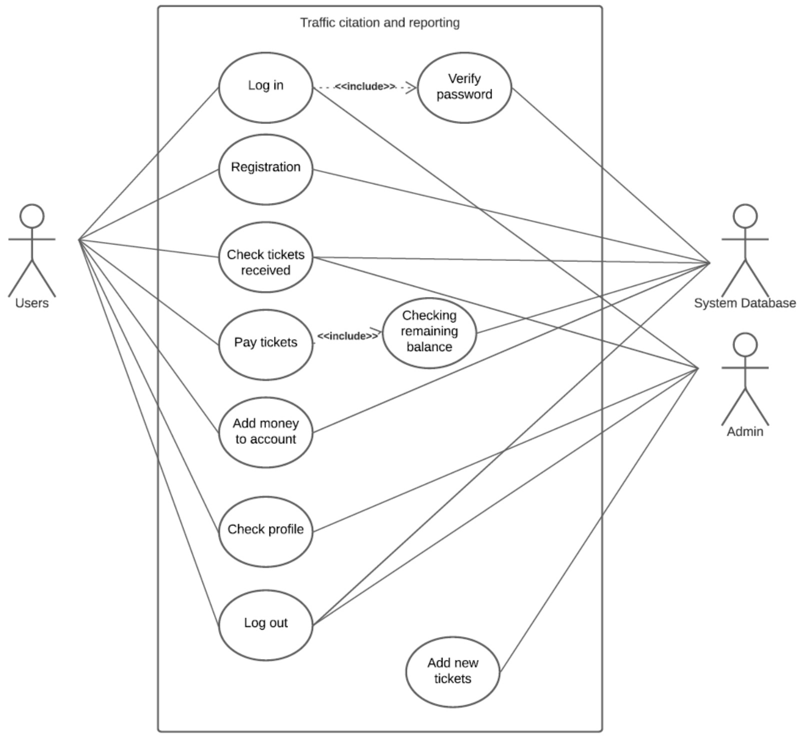


Component Diagram

3.4 Persistent Data Management

The databases used by the TCRS will be SQL this will allow data to be manipulated using the relational schema. SQL is a standard query language. TCRS will benefit from using SQL because we will be able to create tables of user profiles, add tickets and their amounts and many other functions that will be able to get from using SQL.

3.5 Access Control & Security



System Administration has unlimited access to any data
Users (Officers) only have access access to profile information read/write ticket information
Users (driver) will have access to edit profile information read ticket information and make payments

Access Matrix

Actors/Objects	Profile information	Ticket	Balance

Administrator	read()
	create()
	delete()
	submit()
		read()
		create()
		delete()
		submit()
		adjust()
			read()
			create()
			submit()
			adjust()

Officer	read()
	create()
		read()
		create()
		submit()
		delete()	

Driver	read()
	create()
		read()	read()
			submit()

Authentication
The authentication system used to verify the association between the user and the database will be user identification & password.

3.6 Global software control
The software of the server should be in an event-driven manner so that it does not crash or overflow in the context of multiple users attempting to use the system. The control will be maintained for both the admin, driver, and officer servers.

3.7 Boundary conditions
Some of the boundary conditions will be to overload a system with N amount of tickets. Utilizing this will see the max limit of tickets that can be issued and dealt with theoretically. Additionally, having multiple people use the system at once can be another aspect of boundary conditions that can be placed. Additionally startup and shutdown testing can be tested by inputting incorrect information and observing what happens. For instance if a user claims to be an officer but is just a driver, then they cannot use the same features as an officer. We will have to make sure wrong inputs and selections will be properly sorted.

4. Subsystem architectures

Database subsystem
The database subsystem is designed using the client-server architecture pattern. The Database component is the server providing services like adding, deleting, modifying and checking to profile client and ticket client.
TCRS will implement the database as a two-dimensional array. The DriverProfile, OfficerProfile and AdminProfile will extend the Profile interface. And the Profile object will be stored as elements in the array.

DriverActivity subsystem
The driver activity subsystem is designed to be implementing the user interface experience for the driver. The driver should be able to upload their payments, and gain more information about their tickets and driving information as a user.
To upload their payment, it will be stored in a double, and the drivers infractions profile will be displayed as a string.

OfficerActivity subsystem
The officer subsystem architecture should have the structure for modifying and issuing tickets as well as validating whether the ticket has been cleared or not, and with this special access to issuing tickets as well.
This can all be done with string functions to process tickets, update balance owed with double variables, and confirming whether a ticket has been paid with boolean elements.

AdminActivity subsystem
With the admin activity subsystem system security architecture should be in mind, as well as the fundamentals to maintain the integrity of the website, and no false modifications.
The admin activity holds the security because it also upholds critical information for the website infrastructure such as passwords, id's, licenses, and account numbers. The admin is able to do all this through the admin interface, which checks the onCreate methods from other users.

5. Glossary

DriverProfile
UserFirstName: It is a string of the users first name
UserLastName: It is a string of the users last name
UserPassword: A unique password created for the account

Database
Profiles are held in an array that points to the profiles
The size of it is kept track in an integer for memory reasons.

LoginActivity:
OnCreate = Initializes the login process when the user logs in. a void process.

SignUpActivity: onCreate a void process.

TicketHistory: The history of the tickets is also held in an object array.
AddTickets: Adds to the array, this will be done with a void method.
DeleteTickets: Deletes a ticket from an array with a void method.

Tickets
TicketsNo: String which stores the ticket numbers.
DateRecieved: Holds the date when the ticket was received
ViolationType: String of what the violation was.

FineAmount: Held in a double value and gets the total amount owed due to the infraction.

DriverProfile This section generally keeps a log of the drivers information

FirstName: Drivers first name
LastName: Drivers last name
Password: Unique characters to identify account
LicenseNo: Holds the license plate number
Balance: Holds the balance owed in a double.
TicketHistory: Shows past tickets that were offered to this unique driver profile.

OfficerProfile holds the first name, last name, and password information for the officer's profile.

FirstName: Officers first name
LastName: Officers last name
Password: Unique password that the officer uses

AdminProfile This stores the login information for the system admin using this application.

FirstName: Admins first name
LastName: Admins last name
Password: Unique password for the admin.

DriverActivity onCreate: Utilizes the OnCreate function previously defined.
CheckProfile: Obtains the drivers profile information and checks if it is valid.
CheckTickets: Checks for the tickets that the driver has obtained.
PayTicket: Allows the user to pay for the infraction.
AddMoney: Adds money to the balance.
LogOut: Void takes the user to a different interface.

OfficerActivity: Gives the same functions as the Driver activity however is able to give tickets as well
GiveTicket: Boolean
And can also check the profile of the user that paid.

onCreate = utilizes the onCreate method previously mentioned.
CheckProfile: Checks the user profile with the admin view.
CheckDriverProfile: Able to check the database of driver profiles.
CheckTicket: Checks the ticket
LogOut: Void returns to the logout screen