**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

The Traffic Sign Recognition and Training

|  |  |
| --- | --- |
| **Group 2** | |
| **Group member** | Mai Văn Tân – Team Leader – SE90061  Bùi Việt Phong – Team Member - SE60747  Hồ Đắc Nghĩa – Team Member - SE60628 (Drop out)  Trần Lê Tuấn – Team Member – 60350 (Drop out) |
| **Supervisor** | Mr. Kiều Trọng Khánh |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | TSRT |

-Ho Chi Minh City, 01/2014-

*This page is intentionally left blank*

***ACKNOWLEDGEMENTS***

We wish to thank various people for their contribution to this project: Our teachers for their advices and participant in the final review, our friend for the valuable technical support.

Special thanks should be given to Mr.Kiều Trọng Khánh, our research supervisor for his professional guidance and the useful, constructive recommendations throughout the course of this project.

# Table of Contents

[Table of Contents 4](#_Toc385451684)

[List of Tables 6](#_Toc385451685)

[List of Figures 7](#_Toc385451686)

[Definitions, Acronyms, and Abbreviations 8](#_Toc385451687)

[A. Project Management Plan 9](#_Toc385451688)

[1. Project Information 9](#_Toc385451689)

[B. Software Project Management Plan 9](#_Toc385451690)

[1. Problem Definition 9](#_Toc385451691)

[1.1 Name of this Capstone Project 9](#_Toc385451692)

[1.2 Problem Abstract 9](#_Toc385451693)

[1.3 Project Overview 9](#_Toc385451694)

[2. Project organization 11](#_Toc385451695)

[2.1 Software Process Model 11](#_Toc385451696)

[2.2 Roles and responsibilities 12](#_Toc385451697)

[2.3 Tools and Techniques 12](#_Toc385451698)

[3. Project Management Plan 13](#_Toc385451699)

[3.1 Iteration 13](#_Toc385451700)

[Table 5: Iteration 13](#_Toc385451701)

[C. Software Requirement Specification 14](#_Toc385451702)

[1. User Requirement Specification 14](#_Toc385451703)

[1.1 Guest Requirement 14](#_Toc385451704)

[1.2 User Requirement 14](#_Toc385451705)

[1.3 Staff Requirement 14](#_Toc385451706)

[1.4 Admin Requirement 14](#_Toc385451707)

[1.5 System Requirement 14](#_Toc385451708)

[2. System Requirement Specification 14](#_Toc385451709)

[2.1 External Interface Requirement 14](#_Toc385451710)

[2.2 System Overview Use Case 15](#_Toc385451711)

[3. Software System Attribute 15](#_Toc385451712)

[3.1 Usability 15](#_Toc385451713)

[3.2 Reliability 16](#_Toc385451714)

[3.3 Availability 16](#_Toc385451715)

[3.4 Security 16](#_Toc385451716)

[3.5 Maintainability 16](#_Toc385451717)

[3.6 Portability 16](#_Toc385451718)

[3.7 Performance 16](#_Toc385451719)

[4. Conceptual Diagram 16](#_Toc385451720)

[**Data Dictionary 17**](#_Toc385451721)

[D. Software Design Description 18](#_Toc385451722)

[1. Design Overview 18](#_Toc385451723)

[2. System Architectural Design 18](#_Toc385451724)

[3. Component Diagram 19](#_Toc385451725)

[**Data Dictionary** 19](#_Toc385451726)

[4. Detailed Description of Components 20](#_Toc385451727)

[4.1 Class Diagram 20](#_Toc385451728)

[4.2 Class Diagram Explanation 20](#_Toc385451729)

[5. User Interface Design 24](#_Toc385451730)

[6. Database Design 24](#_Toc385451731)

[6.1 Entity Relationship Diagram 24](#_Toc385451732)

[6.2 Data Dictionary 25](#_Toc385451733)

[7. Algorithms 26](#_Toc385451734)

[7.1 Traffic sign detection 26](#_Toc385451735)

[7.2 Traffic sign recognition 27](#_Toc385451736)

[E. System Implementation & Test 30](#_Toc385451737)

[1. Introduction 30](#_Toc385451738)

[2. Database Relationship Diagram 30](#_Toc385451739)

[3. Performance measures 31](#_Toc385451740)

[3.1 Traffic Sign Detection Performance 31](#_Toc385451741)

[3.2 Traffic Sign Recognition Performance 31](#_Toc385451742)

[Figure 12: Traffic Sign Recognition Accuracy 31](#_Toc385451743)

[4. Test Plan 31](#_Toc385451744)

[5. System Testing Test Case 32](#_Toc385451745)

# List of Tables

[Table 1: Hardware Requirement for Server 10](#_Toc385451746)

[Table 2: Hardware Requirement for Web User 10](#_Toc385451747)

[Table 3: Hardware Requirement for Mobile User 11](#_Toc385451748)

[Table 4: Roles and Responsibility Details 12](#_Toc385451749)

[Table 5: Iteration 13](#_Toc385451750)

[Table 6: Data Dictionary 25](#_Toc385451751)

# List of Figures

[Figure 1: Agile Development Model 11](#_Toc385451752)

[Figure 2: System Overview Use Case 15](#_Toc385451753)

[Figure 3: Conceptual Diagram 17](#_Toc385451754)

[Figure 4: N-tier Architecture 18](#_Toc385451755)

[Figure 5: Component Diagram 19](#_Toc385451756)

[Figure 6: Class Diagram 20](#_Toc385451757)

[Figure 7: Entity Relationship Diagram 24](#_Toc385451758)

[Figure 8: Viola-Jones flow chart 27](#_Toc385451759)

[Figure 9: Classifier flow chart 29](#_Toc385451760)

[Figure 10: Database Relationship Diagram 30](#_Toc385451761)

[Figure 11: Traffic Sign Detection Accuracy 31](#_Toc385451762)

[Figure 12: Traffic Sign Recognition Accuracy 31](#_Toc385451763)

[Figure 13: Core Workflow 32](#_Toc385451764)

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| TSRT | Traffic Sign Recognition and Training |
| TS | Traffic sign |
| Traffic Sign Detection | Determines the locations and sizes of traffic signs in images. |
| Traffic Sign Recognition | Identify the name of traffic sign from detected area in an image. |
| Train Image | Image use to train to recognition traffic sign. |
| Model | The result after train, use to classify, recognize traffic sign. |

# Project Management Plan

## Project Information

* Project name: **The Traffic Sign Recognition and Training**
* Project Code: **TSRT**
* Product Type: **Website, Phone Application**
* Start Date: **January 6th, 2013**
* End Date: **April 19th, 2013**

# Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

The Traffic Sign Recognition and Training (TSRT)

### Problem Abstract

Now a day, the traffic system is expand more than before. Beside that the traffic sign use to control traffic is developed to help people when join in the traffic. In Vietnam, we have more than 200 traffic signs, this may cause the difficult to people when meet a new traffic sign. On the other hand, internet and smartphone is more popular in Vietnam. Now it’s easy to have a smartphone with internet connection. That is the reason we want to build a system to help people find information about any traffic sign easily. So people can know more about the sign and traffic rule in each traffic sign, help them join in traffic easier and safer.

### Project Overview

#### The Current System

Below are some current system:

* Search using book:

+ Advantages: clearly, accurate information.

+ Disadvantages: slow and not update regularly.

* Current application:

+ Advantage: update regularly, easy to use.

+ Disadvantages: few information (just have basic information about traffic sign), search slow.

* Using internet:

+ Advantages: provide updated information.

+ Disadvantages: search slowly, need internet connection, too many useless information.

#### The Proposed System

* Develop a recognition system that support auto detect and recognize traffic sign using smartphone’s camera.
* Support more information about the traffic sign: penalty fee, view history.
* Support user learn traffic sign.

##### Web

* The admin can manage information about system, staff and user.
* The system provides a method for admin to set permission for staff, select user to set permission.
* The system provides a method for admin configure system.
* The admin can make statistic about register and search.
* Staff can manager traffic sign: add/edit/delete traffic sign.
* Staff can manager report.
* The system will make a traffic sign take note for user. Support sticker to learn traffic sign.
* The system will provide a method for user upload image to website. Then, the system will recognize the image and give information for user.
* The system will provide a method for user view user’s history, user can send report if have any errors happen.

##### Mobile

* The mobile version also allow guest to register a new account.
* The user or guest can take a photo of traffic sign to make auto search about its information.
* In case the result is wrong, user can send a report about this result.
* Allow user to add traffic sign into favorite list for learning purpose.

#### Boundaries of the System

* The system is intended for almost Vietnam’s land traffic sign.
* Using in normal weather condition (sunny, cloudy...) and traffic sign must be in good state (In original shape, isn’t covered by other objects...)
* The system is intended for learning purpose only.
* The language of the system is Vietnamese.
* The complete product includes:

+ The website for admin, staff, user and guest to interactive with the system.

+ Mobile application for user and guest.

+ All the process involved document.

#### Development Environment

##### Hardware requirements

**For server**

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | 4Mbps | 8 Mbps |
| Operating System | Ubuntu 12 | Ubuntu 13.10 |
| Computer Processor | Intel® Core 2 Duo 2GHz | Intel® Core(TM) i5 CPU , M 460 @ 2.53GHz |
| Computer Memory | 512MB RAM | 3GB or more |

Table 1: Hardware Requirement for Server

**For Web User**

|  |  |  |
| --- | --- | --- |
| Web | Minimum Requirements | Recommended |
| Internet Connection | 2Mbps | 4Mbps |
| Web Browser | Firefox 26 | Firefox 28 |

Table 2: Hardware Requirement for Web User

**For Mobile User**

|  |  |  |
| --- | --- | --- |
| Mobile | Minimum Requirements | Recommended |
| Internet Connection | 2Mbps | 4Mbps |
| Operating System | Android 4.0 | Android 4.0 |
| Hardware | Touchscreen, Camera 5.0 MP or above | Touchscreen, Camera 5.0 MP or above |
| Memory | 512 MB or more | 1 GB or more |

Table 3: Hardware Requirement for Mobile User

##### Software requirements

* Microsoft Windows 7 Service Pack 1, Ubuntu 13.10: operating system and platform for development.
* MySql 5.5.20: use for database system.
* StarUML 5.0, Visio 2010: used to create models and diagrams
* Skype: used for communication and meeting
* Eclipse Juno, Android SDK, ADT 22.0.5 & JDK 1.7.0\_51: used to implement mobile application.
* Eclipse Kepler: use to implement web application and web service.
* Netbean: use to implement console application in C++ (detect and recognize module).
* Google Code & TortoiseSVN: used for source control.

## Project organization

### Software Process Model

Project is developed under agile model.

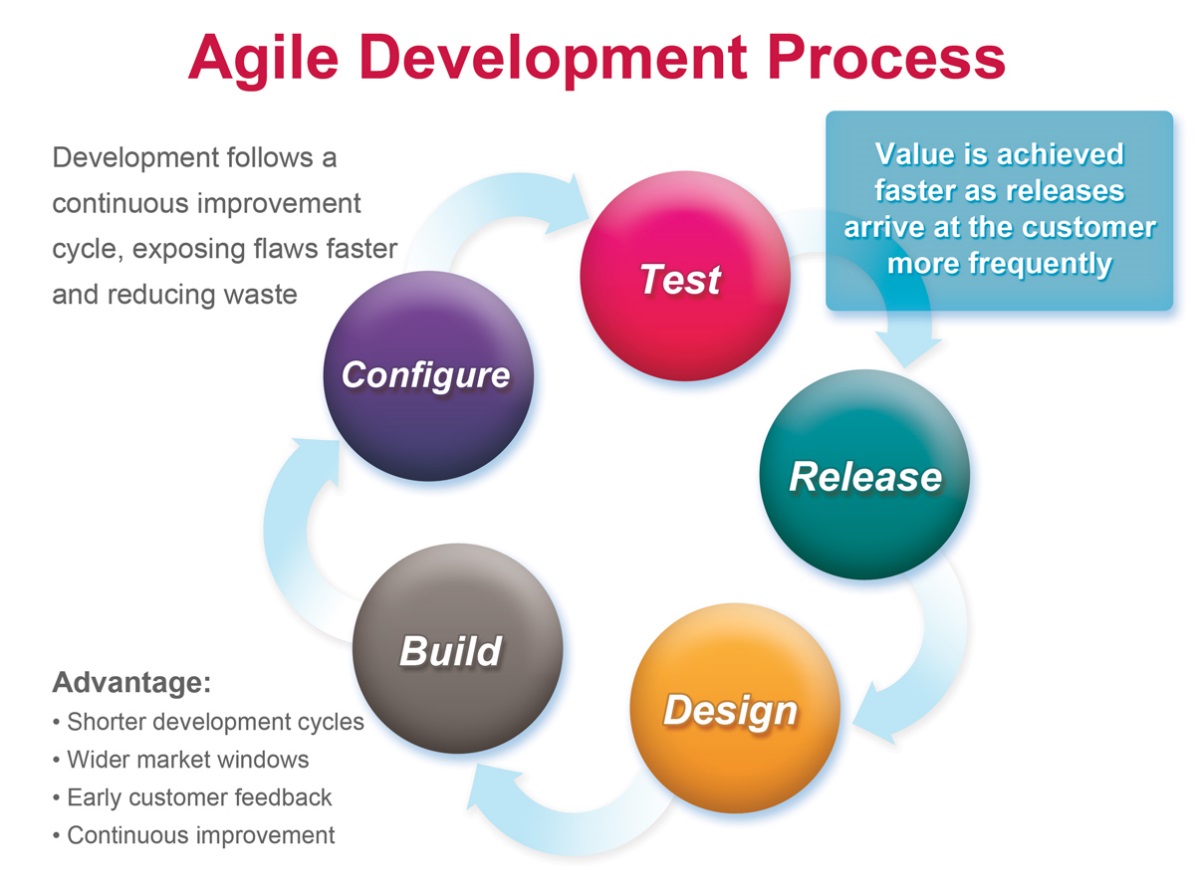


Figure 1: Agile Development Model

[<http://masokotanzania.com/5-steps-to-making-agile-development-work>]

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| **1** | Kiều Trọng Khánh | Project manager | * Specify user requirement * Control the development process * Give out technique and business analysis support |
| **2** | Mai Văn Tân | Team Leader, BA, DEV, Tester | * Managing process * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |
| **3** | Bùi Việt Phong | Team Member, BA, DEV, Tester | * Designing database * Clarifying requirements * Prepare documents * GUI Design * Create test plan * Coding * Testing |

Table 4: Roles and Responsibility Details

### Tools and Techniques

- Front-end technologies: HTML5, CSS3, JavaScript, jQuery, AJAX.

- Web application: Java servlet.

- Web Service: rest full Jersey.

- Mobile App: Android - Java.

- Web Server: Tomcat 7.0.

- Database Management System: MySql 5.5.20

## Project Management Plan

### Iteration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Phase**  **/Iteration** | **Description** | **Deliverables** | **Resource needed** | **Dependencies and Constrains** | **Risks** |
| **Preliminary Investigation or Analysis** | - Study similar existing systems.  -Identify and clarify requirements for the system in general. | -Introduction of proposed system.  -Main functions.  -Project Iteration Plan. | 30 man-days | N/A | Project may  not be feasible  for developing  because lack of technologies  and/or data |
| **Traffic Sign Detect & Recognize** | - Study traffic sign detect and recognize algorithm.  - Implement, testing and statistic result.  - Optimize algorithm. | Traffic sign detect and recognize console app. | 25 man-days | N/A | Lack of experience.  The implemented algorithm is not the best.  Lack of test data |
| **Account Manage** | Implement code for manage account | Account manage functions | 30 man-days | N/A |  |
| **Traffic sign manage** | Implement code for traffic sign manage | Traffic sign manage functions | 30 man-days | N/A |  |
| **Search traffic sign** | Implement code for search traffic sign | Traffic sign search functions (auto search and manually search) | 30 man-days | Traffic sign detect and recognize |  |
| **Favorite, history and report manage** | Implement code for manage favorite, history and report manage | Favorite, history and report manage functions | 30 man-days | Search traffic sign |  |

# Table 5: Iteration

# Software Requirement Specification

## User Requirement Specification

### Guest Requirement

Guest, who is not have account or not login system. Guest can use the basic functions of system like search auto and search manually

### User Requirement

User is normal user of the system. When login system, use can search auto, search manually beside that user have favorite, store favorite traffic sign and history store old requests. When view traffic sign information and search auto use also can report to help system work more accurate.

### Staff Requirement

Staff is the main manager of the system. Staff’s account is set by admin. Staff can use system with these functions:

* Manage user account: Active/Inactive account
* Manage traffic sign: add/delete/edit/import/export traffic sign
* Manage report that user submit: view/delete report
* Make statistic

### Admin Requirement

Admin is highest manage in system. Admin can do functions:

* Manage account: set/unset staff account, active/inactive
* Manage system: configure system
* Make statistic

### System Requirement

System is also an important part. System can do functions:

* Detect traffic sign
* Recognize traffic sign
* Store user history
* Store user favorite

## System Requirement Specification

### External Interface Requirement

#### User Interface

* + - The interface of website must be clear.
    - The interface of mobile app must be clear, compatible with touch screen. The size of controls must be big enough to touch on smartphone.
    - The error, warning messages must be make clear, easy to understand.

#### Software Interface

* Firefox with Resolution (1024x768) or bigger and support JavaScript and HTML5
* Smartphone with Android 4.0. Screen size 4 inch or bigger.

#### Communication Protocol

* Website using HTTP protocol for communication between the web browser and the web server.
* Mobile app using HTTP protocol for communicating between app and web service.

### System Overview Use Case



Figure 2: System Overview Use Case

## Software System Attribute

### Usability

#### Graphic User Interface

* All the text, label and image should be in Vietnamese.

#### Usability

* Website admin, staff should need only two days of training to use the system.
* The guest, user can use the system easily without training.

#### Installation

* The system must be easy to deploy. Customer can deploy successfully and learn to configure, maintain the system within one day of training.
* The mobile app must be easy to install. Compatible with Android OS 4.0.
* The attached manual guide must be clear.

### Reliability

* System have recognition accuracy about 60-80 percent

### Availability

* The user search, view favorite and history offline using mobile in case of network or server unavailable.

### Security

* Privacy: Each role of user has a specific permission to interact with system.
* Only admin can grant permission to staff.

### Maintainability

* The codes is easy to maintain and upgrade.

### Portability

* N/A

### Performance

- The traffic detection and recognition take less than 5 minutes to search traffic in an image.

## Conceptual Diagram

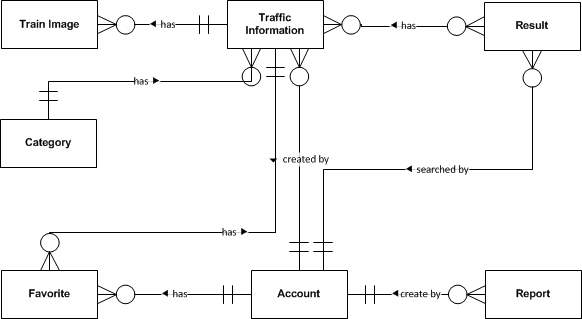


Figure 3: Conceptual Diagram

**Data Dictionary**

|  |  |
| --- | --- |
| **Entity Data dictionary: describe content of all entities** | |
| **Entity Name** | **Description** |
| Traffic Information | Describe information about traffic sign |
| Train Image | Describe information about image to support recognize traffic sign |
| Category | Describe all type of traffic sign |
| Report | Describe information about report of user. Contain content when user though the search result or traffic information wrong. |
| Account | Describe all information about user in the system |
| Result | Describe information about search auto result (user’s history) |
| Favorite | Describe information about favorite traffic sign of user |

# Software Design Description

## Design Overview

## System Architectural Design

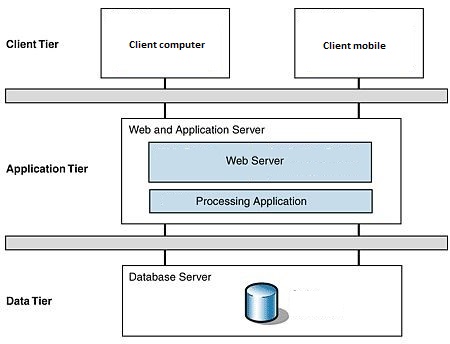


Figure 4: N-tier Architecture

<http://www.cardisoft.gr/frontend/article.php?aid=87&cid=96>

* **Client Tier**: The client tier interacts with the users for the solution. Since the application conforms to a three layered services application it hosts the presentation layer components.
* **Application Tier**: The servers used in the application tier are responsible for host­ing all the application's business components and, in the case of Web applications, the Web servers as well.
* **Data Tier**: The servers in the data tier host the databases that the application requires; it is within this tier that the data layer is hosted.

## Component Diagram

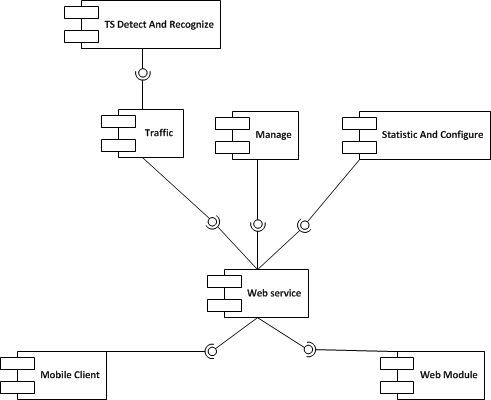


Figure 5: Component Diagram

**Data Dictionary**

|  |  |
| --- | --- |
| **Name** | **Description** |
| TS Detect And Recognize | Provide functions to detect and recognize traffic sign |
| Traffic | Provide functions about traffic sign: manage, search |
| Manage | Provide functions to manage account, favorite |
| Statistic And Configure | Provide functions to statistic and configure server |
| Web service | Provide service interface using http protocol |
| Mobile client | Provide functions in mobile |
| Web Module | Provide functions in web |

## Detailed Description of Components

### Class Diagram

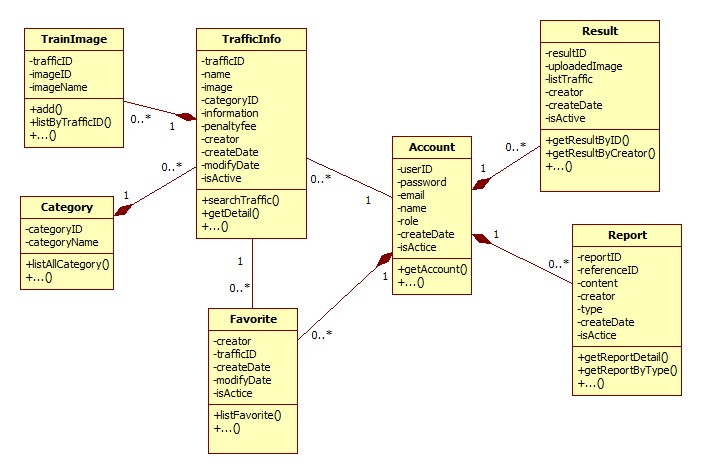
–

Figure 6: Class Diagram

### Class Diagram Explanation

#### TrafficInfo

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| trafficID | String | Private | Traffic sign no |
| name | String | Private | Name of traffic sign |
| image | String | Private | Traffic sign image link |
| categoryID | Int | Private | Category ID of traffic sign |
| information | String | Private | Content of traffic sign |
| penaltyfee | String | Private | Penalty fee of traffic sign |
| creator | String | Private | Creator of traffic sign |
| createDate | Date | Private | Create date |
| modifyDate | Date | Private | Modify date |
| isActive | Boolean | Private | Status of traffic sign |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| searchTraffic() | ArrayList<TrafficInfo> | Public | Search traffic by input parameter: name, cateID, limit |
| getDetail() | TrafficInfo | Public | Get traffic sign detail |

#### Account

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| userID | String | Private | UserID |
| password | String | Private | User’s password in MD5 form |
| email | String | Private | User’s email |
| name | String | Private | User’s name |
| role | String | Private | Role in system |
| createDate | Date | Private | Create date |
| isActive | Boolean | Private | Status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getAccount() | Account | Public | Get user information |

#### Category

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| categoryID | Int | Private | ID of category |
| categoryName | String | Private | Name of category |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| listAllCategory() | ArrayList<Category> | Public | Get list all category |

#### TrainImage

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| trafficID | String | Private | TrafficID of Image |
| imageID | String | Private | ID of image |
| imageName | String | Private | Name of image |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| add() | Boolean | Public | Add a train image |
| listByTrafficID() | ArrayList<TrainImage> | Public | List all train image of a traffic sign |

#### Favorite

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| creator | String | Private | Creator of favorite |
| trafficID | String | Private | Traffic ID of favorite |
| createDate | Date | Private | Create date |
| modifyDate | Date | Private | Modify date |
| isActive | Boolean | Private | Status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| listFavorite() | ArrayList<Favorite> | Public | List all favorite of an user |

#### Result

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| resultID | Int | Private | Unique ID for result |
| uploadedImage | String | Private | Image link of result |
| listTraffic | String | Private | List result contain trafficID and their location in image |
| creator | String | Private | Creator of result |
| createDate | Date | Private | Create date |
| isActive | Boolean | Private | Status of result |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getResultByID() | Result | Public | Get result detail by ID |
| getResultByCreator() | ArrayList<Result> | Public | Get all result of an user |

#### Report

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| reportID | Int | Private | Unique id for report |
| referenceID | String | Private | Reference Id of report, this ID reference to TrafficInfo or Result depend on type of report |
| content | String | Private | Content of report |
| creator | String | Private | Creator of report |
| type | Int | Private | Type of report. 1 for wrong recognize, 2 for wrong traffic information |
| createDate | Date | Private | Create date |
| isActive | Boolean | Private | Status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getReportDetail() | Report | Public | Get detail of report |
| getReportByType | ArrayList<Report> | Public | Get all report by type |

## User Interface Design

## Database Design

### Entity Relationship Diagram

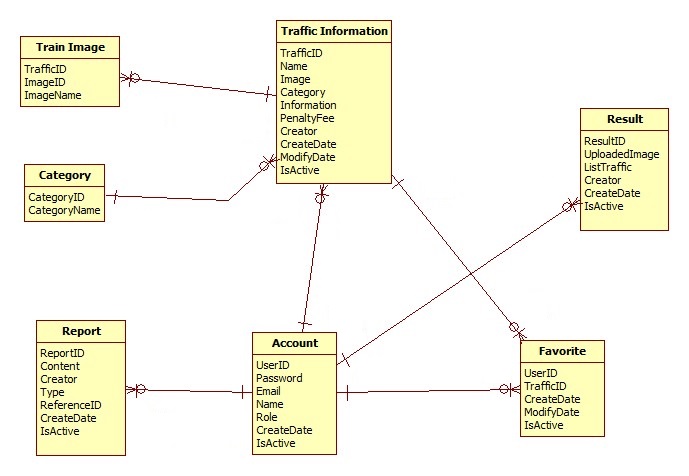


Figure 7: Entity Relationship Diagram

### Data Dictionary

|  |  |
| --- | --- |
| **Data dictionary: describe content of all entities** | |
| Name | Description |
| Traffic Information | Describe information about traffic sign |
| Train Image | Describe information about image use to train/recognize a traffic sign |
| Category | Describe all type of traffic sign |
| Report | Describe information about report of user. Contain wrong information report and wrong recognition report |
| Account | Describe all information about user in the system |
| Result | Describe information about search auto result |
| Favorite | Describe information about favorite traffic sign of user |

Table 6: Data Dictionary

## Algorithms

### Traffic sign detection

#### Definition

#### Different method for traffic sign detection

#### The method we chose

We chose the Viola-Jones method (a classifier cascade method). Reason:

* Already implemented in OpenCV (that support many platforms: Linux, Windows, Android…)
* Easy to extend base on train cascade file.
* Easy to use and handle in code.
* High accurate rate (base on testing with another method)

**Method Introduction**

The Viola-Jones (V-J) object detection framework is the first object detection framework to provide competitive object rates in real-time proposed. It can be trained to detect a variety of object classes. In OpenCV, it’s implemented as cvHaarDetectObjects() and the new method for multithread is cascadeClassifier()

Reference: <http://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework>

Key ideas of Viola-Jones:

* Integral images for fast feature evaluation
* Boosting for feature selection
* Attentional cascade for fast rejection of non-object windows

Viola-Jones Technique:

* Feature extraction and feature evaluation
* Classifier training a feature selection using a method called AdaBoost.
* A degenerate decision tree of classifiers is formed.

**All Viola-Jones flow chart:**

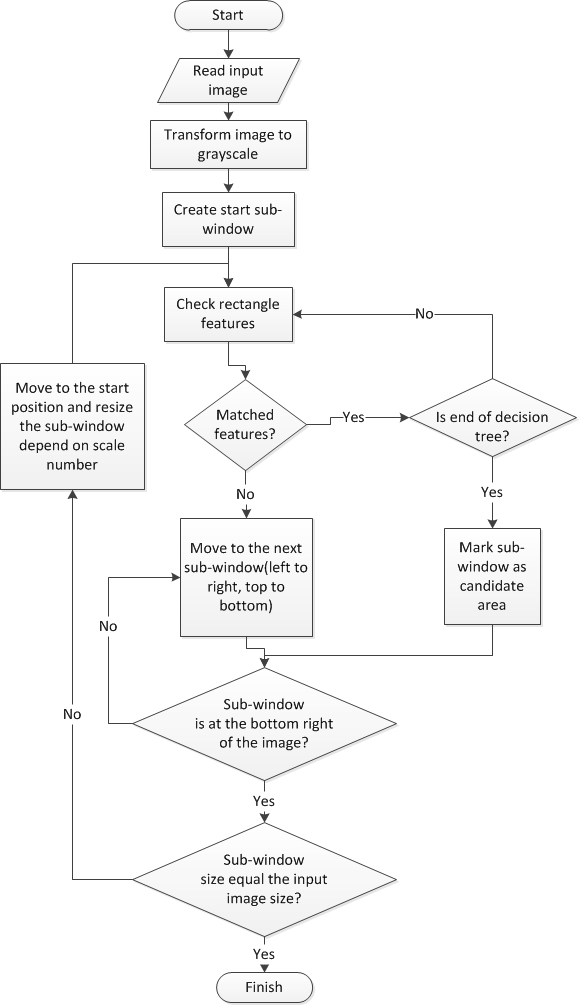


Figure 8: Viola-Jones flow chart

### Traffic sign recognition

#### Definition

#### Method for traffic sign recognition

#### The Method we choose

**We chose SVM multi-class method, reason:**

* Implemented in many library
* Easy to integrate in project

**Method Introduction**

**Support vector machine (SVMs) are supervised learning models with associated learning algorithms what analyze data and recognize pattern used for classification and regression analysis.**

**Basic SVM only support to classify 2 classes.**

**Multiclass SVM: assign labels to instances by using support vector machines, where the labels are drawn from a finite set of several elements.**

**This approach is extend of basic SVM (binary classification):**

**One-versus-all: Using binary classifiers with between one of the labels and the rest. The result calculate by winner-takes-all, in which the classifier with the highest output function assigns the class.**

**One-versus-one: between every pair of classes. The result calculate by max-wins voting strategy, in which every classifier assigns the instance to one of the two classes, then the vote for assigned class is increased by one vote, and finally the class with the most votes determines the instance classification.**

**In this project we use SVM multi-class with One-versus-all method.**

Reference:

<http://en.wikipedia.org/wiki/Support_Vector_Machine>

<http://www.cs.cornell.edu/people/tj/svm_light/svm_multiclass.html>

**Algorithm description:**

* Extract features of image
* Compare these features with the features of the model and get the best matched rate model (SVM multi-class One-versus-all)

**Classifier flow chart**

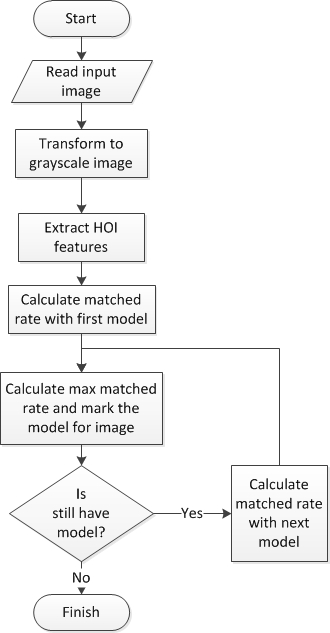


Figure 9: Classifier flow chart

### Sync favorite list

#### Definition

This system allows user to use in both mobile and website. Besides that, application in mobile can run normally even if there is no connection to server. Thus, there should be a solution to sync favorite between mobile and website.

#### Solution

**Algorithm description:**

It is suggested that there is a variable saving the final time being added or deleted for each favorite item. During synchronization process, each favorite in mobile will be sent to server in respond to action of adding or deleting. The server adds or deletes only if modifying time shown in mobile is later than the one in server. The last step is delete favorite list in mobile and replace by the one in server.

**Advantage:**

* + Allow sync favorite list between mobile and server by time order.
  + Avoid losing favorite items during synchronization process.

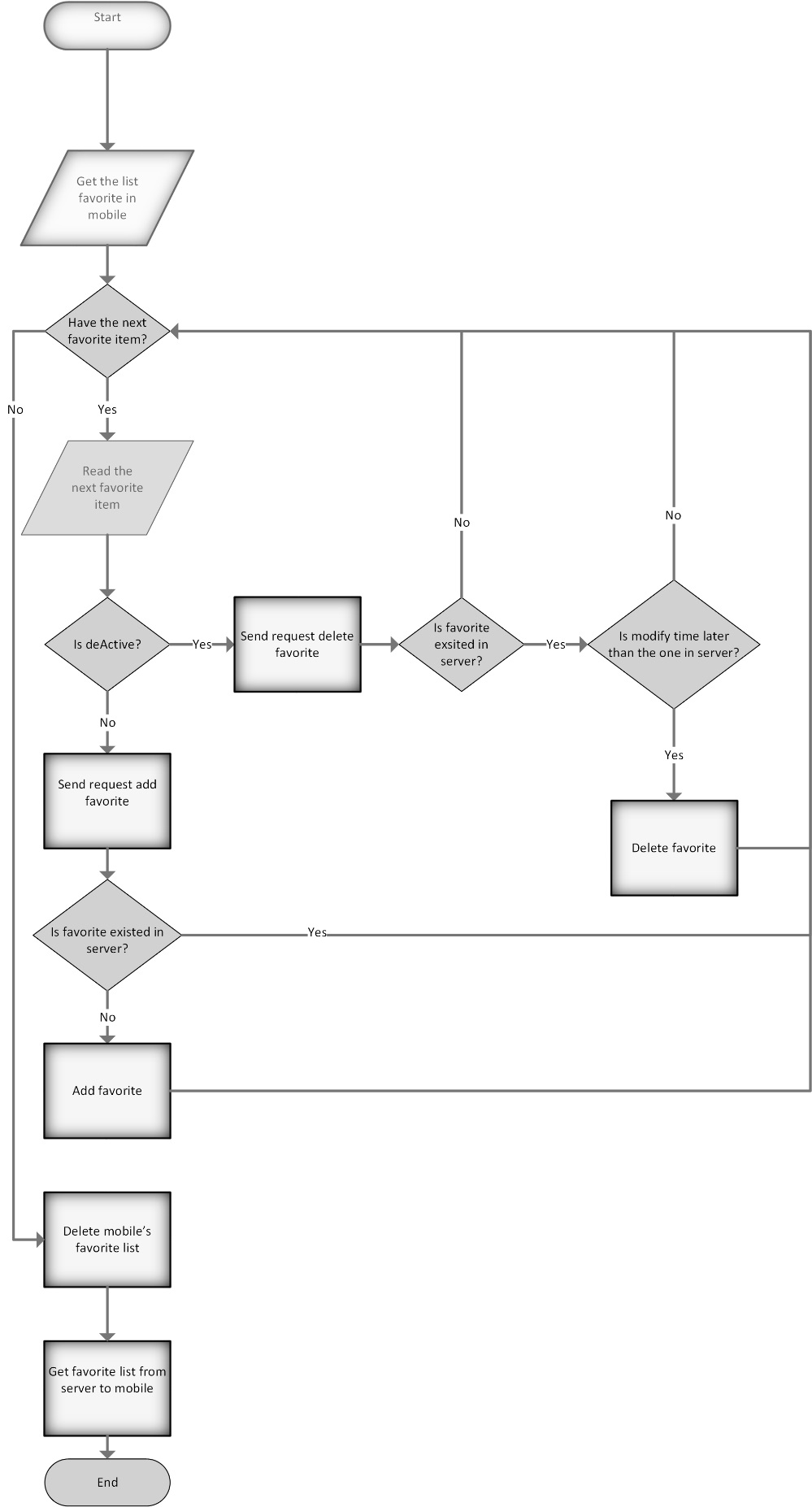


Figure 70: Sync favorite flow chart

# System Implementation & Test

## Introduction

## Database Relationship Diagram

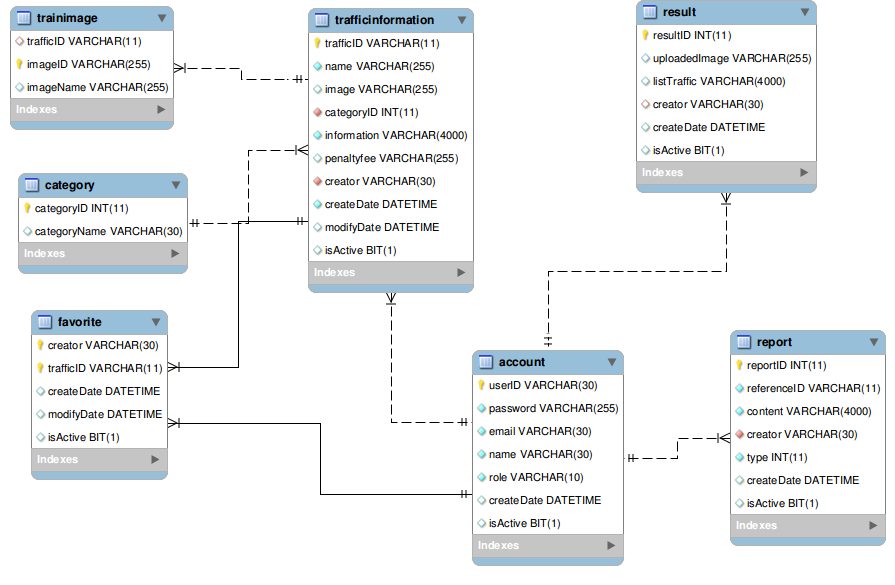


Figure 10: Database Relationship Diagram

## Performance measures

### Traffic Sign Detection Performance

With image resolution: 800x460px

Web module: Ubuntu 13.10 with core i3 M350 2.27GHz, 3GB Ram

* Best traffic sign detected result: 60/84 traffic sign (two types). Percent: 71.4%
* Worst traffic sign detected result: 56/84 traffic sign (two types). Percent 66.7%

Detection rate of type 1 base on different train image data:

Figure 11: Traffic Sign Detection Accuracy

Mobile module: Android 4.0 with 2 cores 1GHz, 512MB Ram

* Average time to detect traffic sign: 0.357s (2.8 FPS)

### Traffic Sign Recognition Performance

With image resolution: 800x460px

Web module: Ubuntu 13.10 with core i3 M350 2.27GHz, 3GB Ram

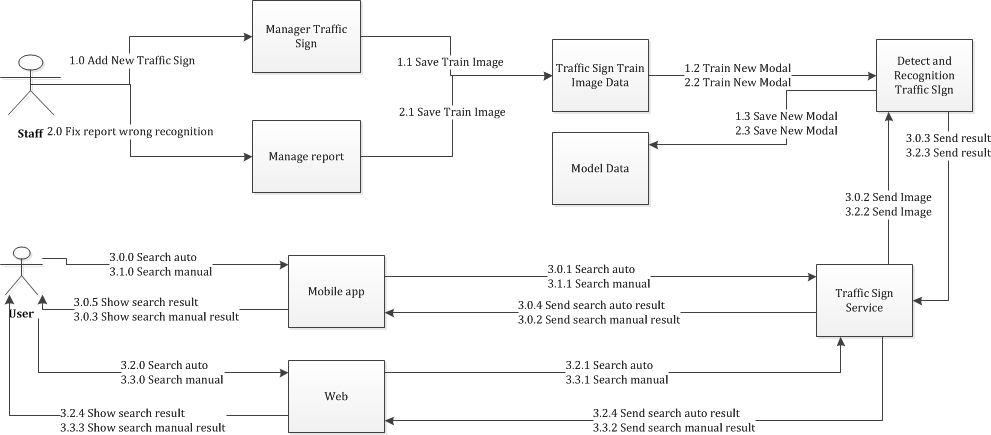
* Best time to run traffic sign recognition: 0.013s
* Worst time to run traffic sign recognition: 0.016s

Recognition accuracy (good condition):

## Figure 12: Traffic Sign Recognition Accuracy

## Test Plan

## System Testing Test Case



**Figure 13: Core Workflow**